This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2016.2562621, IEEE Transactions on Knowledge and Data Engineering

Contextual Operation for Recommender Systems

Shu Wu, *Member, IEEE,* Qiang Liu, Liang Wang, *Senior Member, IEEE,* and Tieniu Tan, *Fellow, IEEE*

Abstract—With the rapid growth of various applications on the Internet, recommender systems become fundamental for helping users alleviate the problem of information overload. Since contextual information is a significant factor in modeling the user behavior, various context-aware recommendation methods have been proposed recently. The state-of-the-art context modeling methods usually treat contexts as certain dimensions similar to those of users and items, and capture relevances between contexts and users/items. However, such kind of relevance has much difficulty in explanation, e.g., it is not intuitive that a user is more relevant to $weekday$ than $weekend$. Some works on multi-domain relation prediction can also be used for the context-aware recommendation, but they have limitations in generating recommendations under a large amount of contextual information. Motivated by recent works in natural language processing, we represent each context value with a latent vector, and model the contextual information as a semantic operation on the user and item. Besides, we use the contextual operating tensor to capture the common semantic effects of contexts. For the contextual information of each user-item interaction, the contextual operation can be modeled by multiplying the operating tensor with latent vectors of contexts. Experimental results show that the proposed Context Operating Tensor (COT) model yields significant improvements over the competitive compared methods on three typical datasets. From the experimental results of COT, we also obtain some interesting observations which follow our intuition. For example, contexts have different importances in operating characteristics of users/items, and the potential relation among context values can be revealed by representation vectors of these values.

Index Terms—Recommender systems, Context-aware, Context representation, Contextual operation

1 Introduction

With the rapid growth of available information on the Internet, users are getting in trouble with the problem of information overload. Recommender systems have become an important tool which can help users to select the information of interest in many web applications such as social networks, e-commerce, online reading and so on. Nowadays, with the enhanced ability of systems in collecting information, a great amount of contextual information has been collected. The contextual information describes the situation of behavior, such as $location$, $time$, $weather$, $companion$ and so on. The user behavior tends to change significantly under these kinds of contexts. For instance, a man may like to watch cartoons when he is with his children, but he may like to watch romantic movies with his wife.

The survey of [1] indicates that contexts of recommender systems specify the contextual information associated with a recommendation application, and provides two kinds of examples which are attributes associated with users or items and attributes associated with user-item interactions. For instance, contexts of a user, such as $gender$, age and $occupation$, can profile this entity, and contexts of a user-item rating, such as $time$, $location$, $companion$ and $platform$, describe situations of this interaction. The work [2] indicates that context-aware methods are more general than attribute-aware methods [3][4], which only take additional information about users and items into account. As shown in Fig. 1, generally, contextual information includes interaction contexts, which describe the interaction situations, and entity contexts, which can identify user/item characteristics. Here, we focus on modeling the general contextual information associated with not only users/items but also user-item interactions.

Due to the fundamental effect of contextual information in recommender systems, many context modeling methods have been developed. Some works [2][5] incorporate contextual information in a factorization model via treating contexts as one or several dimensions which have similar properties as dimensions of the user and the item. Most of these methods calculate the relevance between contexts and entities, but such kind of relevance is not always reasonable [6]. For example, it is not intuitive that a user is more relevant to $weekday$ than $weedend$. In 2014, Shi et al. propose a novel CARS^2 model [6] which provides each user/item with not only a latent vector but also a context-aware representation. Due to the effectiveness of additional context-aware representations, this approach provides a new way for building context-aware recommender systems. However, using a distinct vector to represent contexts of each interaction,

- *The authors are with the Center for Research on Intelligent Perception and Computing (CRIPAC), National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing, 100190, China.*
E-mail: {shu.wu, qiang.liu, liang.wang, tnt}@nlpr.ia.ac.cn.

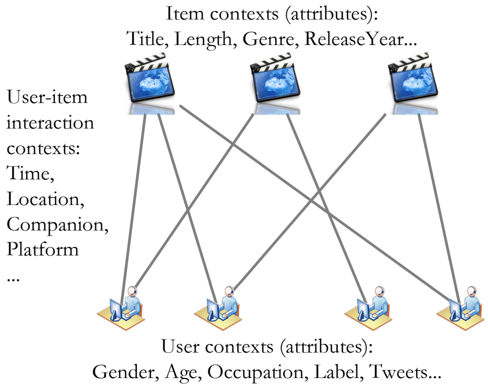

Item contexts (attributes):
Title, Length, Genre, ReleaseYear...
User-item
interaction
contexts:
Time,
Location,
Companion,
Platform
...
User contexts (attributes):
Gender, Age, Occupation, Label, Tweets...

Fig. 1. Contextual information in recommender systems contains user contexts, item contexts and user-item interaction contexts. User contexts or item contexts are attributes associated with the corresponding entity, and interaction contexts describe situations of the user-item interaction.

CARS^2 has the problem in confronting with abundant contextual information in real applications. Besides, since CARS^2 can only model the categorical context, the numerical context should be categorized at first.

Moreover, some works on multi-domain relation prediction [7][8] can also be employed for context-aware recommender systems. These methods incorporate the transfer matrix to map latent vectors of entities from one domain to another. In context-aware recommendation systems, using the transfer matrix, latent vectors of users/items can be mapped from one contextual situation to another. However, similar to the limitation of CARS^2 [6], using a transfer matrix for each specific contextual information, these methods have the difficulty in dealing with a large amount of contextual information.

To overcome the shortages of the existing methods mentioned above, we propose a novel context modeling method Contextual Operating Tensor model, named COT, which is motivated by the recent work of semantic compositionality in Natural Language Processing (NLP). Continuous vector representations of words have a long history in NLP, and become even popular since Mikolov et al. [9] provide an efficient implementation $word2vec$. Inspired by the powerful ability in describing latent properties of words, in recommender systems, using a vector representation of each context value seems a good solution to examine the effect of contexts on user-item interactions. Different from the one-hot representation of contexts in Factorization Machine (FM) [10] and Multiverse Recommendation [5], the distributed representation inferred from all contexts has more powerful ability in illustrating the operation properties of contexts. Moreover, in the research direction of sentence sentiment detection, a noun has semantic information as a latent vector, and an adjective has semantic operation on nouns as an operating matrix [11][12]. For instance, in the phrase "excellent product", the noun "product" is represented by a latent vector, and the adjective "excellent" is associated with a semantic operating matrix which can operate the noun vector of "product". Thus, multiplying the operating matrix with the latent vector, the phrase "excellent product" has a new latent vector which shows not only the latent properties of the "product" but also a positive attitude to the "product". We assume that contexts in recommendation systems have a similar property of adjectives and can operate latent characteristics of users and items. Then, new latent representations of entities can show not only characteristics of original entities but also new proprieties under a specific contextual situation. For instance, a man has his original latent interests. When this man is with children, this $companion$ context operates his latent interests and he may like to watch cartoons with these children. Besides, in real recommendation systems, some contexts have very similar effects. For instance, both $weekend$ and being at $home$ may make you prefer to read novels. Inspired by [13] in simplifying the Matrix-Vector operation, we use contextual operating tensors to capture the common effects of contexts.

The proposed Context Operating Tensor (COT) method learns representation vectors of context values and uses contextual operations to capture the semantic operations of the contextual information. We provide a strategy in embedding each context value into a latent representation, no matter which domain the value belongs to. For each user-item interaction event, we use contextual operating matrices to represent the semantic operations of these contexts, and employ contextual operating tensors to capture common effects of contexts. Then, the operating matrix can be generated by multiplying latent representations of contexts with the operating tensor.

The main contributions are listed as follows:

- To describe the operation ability of contexts, we embed each context value with a latent representation, and model the contextual information as semantic operations on users and items. Context representation and contextual operation present a novel perspective of context modeling.
- We use the contextual operating tensor to capture the common semantic effects of contexts. For each interaction, the contextual operation can be generated from the multiplication of operating tensor and latent vectors of contexts.
- Experimental results on three real datasets show that COT is effective and evidently outperforms the state-of-the-arts. Besides, context representations can reveal the latent relation among these context values, and context weights can indicate different importances of context values in operating latent vectors of users and items.

The rest of the paper is organized as follows: we

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2016.2562621, IEEE Transactions on Knowledge and Data Engineering

review some related works in Section 2. Then, we introduce COT in Section 3, including context representation, contextual operating matrix, contextual operating tensor and the optimization algorithm. In Section 4, we report and analyze experimental results of COT and state-of-the-art methods on real datasets. Finally, we conclude this work and discuss the future research direction.

2 Related Work

In this section, we review some related works on matrix factorization based methods and state-of-the-art context-aware models. In addition, we introduce some recent works on representation learning, which motivate this work.

2.1 Matrix Factorization

Matrix Factorization (MF) based methods [14][15][16] have become a state-of-the-art approach to recommender systems. The basic objective of MF is to factorize a user-item rating matrix into two low rank matrices, each of which represents latent factors of users or items. With the multiplication of two factorized matrices, the original matrix can be reconstructed, and rating predictions are obtained accordingly. Due to the effectiveness of MF, the MF based methods have been studied extensively. SVD++ [17], which combines neighborhood models with latent factor models in one prediction function, is one of the most popular models for recommender systems.

There are some MF based methods which are designed for a specific kind of contexts, such as the time factor and entity attributes. Koren proposes a model named timeSVD++ [18], which is one of the most effective models for time-aware recommendation. Xiong et al. [19] add the time factor as a new dimension to the rating matrix, and factorize a three-dimensional tensor. Attribute-aware MF is another important direction of MF extensions. The attribute-aware recommender systems [3][4][20] extend the conventional MF model to handle the user and item attributes.

2.2 Context-aware Recommender Systems

Contextual information has been proved to be useful for recommender systems [1][21], and various context-aware recommendation methods have been developed. According to the survey of [1], these methods can be categorized into pre-filtering, post-filtering and context modeling. Employing the pre-filtering or post-filtering strategy, conventional methods [22][23][24] utilize the contextual information to drive data selection or adjust the resulting set. Baltrunas et al. use the item-splitting for the contextual pre-filtering process [23]. Li et al. view a context as a dynamic feature of items and filter out the items that do not match a specific context [25]. Some works [26][27] have applied tree-based partition with matrix factorization, which also fall into the pre-filtering category. Zhong et al. propose RPMF [26], which applies tree based random partition to split the user-item-rating matrix by grouping users and items with similar contexts, and then applies matrix factorization on each node of the tree. To deal with contexts and social network in recommender systems, Liu et al. propose SoCo [27], which has the similar idea with RPMF but applies matrix factorization only on the leaf nodes. These pre-filtering and post-filtering methods may work in practice, but they require supervision and fine-tuning in all steps of recommendation [2].

The context modeling methods, using the contextual information directly in the model, have become popular recently. These methods focus on integrating the contextual information with the user-item rating matrix and construct factorization models. The work of [22] proposes a multidimensional recommendation model based on the cube of multiple dimensions such as the user dimension, the item dimension and dimensions for all contexts. Multiverse recommendation [5] represents the rating matrix with contextual information as a user-item-context tensor, which is factorized with Tucker decomposition [28]. Multiverse recommendation has proved performing better than the conventional contextual pre-filtering and post-filtering models. Rendle et al. [2] apply Factorization Machine (FM) for the context-aware recommendation. This method can handle different kinds of contextual information, and factorizes pairwise context relation through generating feature vectors in a proper way. However, since these methods treat contexts as one or several dimensions as those of the user and item, the relation between an entity and a context value is not intuitive and has difficulty in explanation. Recently, Shi et al. propose a novel CARS^2 [6] model which provides each user/item with a latent factor and a context-aware representation. Similar to HeteroMF [8], CARS^2 provides the contextual information of each interaction with a distinct vector, but is not suitable for numerical contexts and abundant contexts in real-world applications.

2.3 Multi-domain Relation Prediction

Multi-domain relation prediction can also be used for the context-aware recommendation. For relation learning in multiple domains [29][30][31], Collective Matrix Factorization (CMF) factorizes the user-item-rating matrix in each domain, and latent vectors of users/items are shared among these domains. However, sharing the same latent vector among different domains makes the model unable to well formulate the properties of entities under different domains. Then, Zhang et al. [7] treat user attributes as priors for user latent vectors, and employ a transfer matrix to generate latent vectors from the general ones.

Similarly, Jamali et al. propose Heterogeneous Matrix Factorization (HeteroMF), which generates context-specific latent vectors using a general latent vector for the entity and context dependent transfer matrices [8]. However, for the context-aware recommendation, with a transfer matrix for contexts in each interaction event, these methods have to estimate numerous matrices for a large amount of contextual information.

2.4 Representation Learning

Here, we introduce several most significant works in NLP, which motivate this work. For continuous vectors of words, the neural network language model [32] is a popular and classic work, which learns a vector representation of each word and proposes statistic language models using artificial neural networks. Mikolov et al. [9] propose neural net language models for computing continuous vector representations of words and provide the tool $word2vec$ for an efficient implementation. For sentence sentiment detection, the work [11] introduces a presentation of adjective-noun phrase, where a noun has semantic information as a latent vector and an adjective has semantic operation on nouns as an operating matrix, then the adjective-noun composition can be represented by multiplying the adjective matrix with the noun vector. Further, Socher et al. propose a model [12] in which each word or longer phrase has a Matrix-Vector representation. The vector captures the meaning of the constituent and the matrix describes how it modifies the meaning of the other combined word. Since each word has a Matrix-Vector representation, the number of parameters becomes very large with an increasing size of vocabulary. Then, Socher et al. [13] propose a global tensor-based composition function for all combinations, and improve the performance of sentence sentiment detection over the Matrix-Vector representation [12].

3 Context Operating Tensor Model

In this section, we present a novel model, named Contextual Operating Tensor (COT), for the context-aware recommendation. We introduce the notations and the fundamental concept of context representation at first, and then present the COT model thoroughly. Finally we describe the process of parameter inference, and the optimization algorithm based on stochastic gradient decent.

3.1 Notations

In typical recommender systems, there is a user set U and an item set V. $\mathbf{u} \in \mathbb{R}^d$ and $\mathbf{v} \in \mathbb{R}^d$ are latent vectors of user u and item v, where d is the dimensionality. There are multiple contexts associated with users, items and user-item interactions, such as age, $gender$, $occupation$, $releaseYear$, $director$, $genre$, $theater$, $time$, $companion$, etc.

In this work, we divide these multiple contexts into user contexts $\mathcal{C}_1^U, \mathcal{C}_2^U, ...,$, item contexts $\mathcal{C}_1^V, \mathcal{C}_1^V, ...$ and interaction contexts $\mathcal{C}_1^I, \mathcal{C}_1^I, ...$ User contexts and item contexts indicate the attribute information associated with the user and item, while interaction contexts describe the situations of user-item interaction. For instance, in the scenario of a movie recommendation system, item contexts contain $title$, $length$, $releaseYear$, $director$ and $genre$, and interaction contexts contain $theater$, $time$ and $companion$, etc.

A specific context value c_m^i is a variable of context $\mathcal{C}_m^I$. Context values of a specific user u, $c^u = \{c_1^u, c_2^u, ...\}$, are called user context combination, and context values of a specific item v, $c^v = \{c_1^v, c_2^v, ...\}$, are named item context combination. The interaction contexts of a user-item rating are named interaction context combination and can be represented as $c^i = \{c_1^i, c_2^i, ...\}$. The rating that user u provides to item v under contextual information c can be written as $r_{u,v,c}$. The general contextual information c associated with rating $r_{u,v,c}$ is composed of user context combination c^u, item context combination c^v and interaction context combination c^i.

The representation vector of a context value c_m^i is denoted as $\mathbf{h}_m^i \in \mathbb{R}^{d_c}$. Each context combination can be illustrated by a latent matrix which consists of latent vectors of context values. Then, user context combination c^u can be represented as a matrix $\mathbf{H}^u = [\mathbf{h}_1^u, \mathbf{h}_2^u, ...] \in \mathbb{R}^{d_c \times |c^u|}$, item context combination c^v as $\mathbf{H}^v = [\mathbf{h}_1^v, \mathbf{h}_2^v, ...] \in \mathbb{R}^{d_c \times |c^v|}$, and interaction context combination c^i as $\mathbf{H}^i = [\mathbf{h}_1^i, \mathbf{h}_2^i, ...] \in \mathbb{R}^{d_c \times |c^i|}$, where $|c^u|$, $|c^v|$ and $\left|c^i\right|$ are the numbers of user contexts, item contexts and interaction contexts respectively.

3.2 Context Representation

There are various types of context values in practical recommender systems, such as categorical value, categorical set value and numerical value. Here, we show how different types of context values can be transformed into the corresponding latent representations.

Categorical domain: If a user watches a movie in a $theater$, $theater$ is the categorical context value. Each context value c_m^i in the categorical domain should be represented by a distinct vector $\mathbf{h}_m^i \in \mathbb{R}^{d_c}$. Fig. 2 shows $male$ and $famale$ in the categorical domain $gender$ are represented by two distinct vector.

Numerical domain: Numerical context values widely exist, e.g., the age of a user, the $length$ of a movie and the $time$ when a user watches a movie. To match with the representations of context values in other domains, we use a vector $\mathbf{h}_m^I \in \mathbb{R}^{d_c}$ to represent a numerical domain. To alleviate the dominant effect of large context values and the negligible effect of small ones, we employ a logistic function in normalization. Assume context values of the numerical domain falls a normal distribution, we can calculate the mean μ and variance σ^2 of this numerical domain. Then a context value c_m^i in this domain is represented as a logistic function $\mathbf{h}_m^i = \left(1 + exp\left(-\left(c_m^i - \mu\right)/\sigma^2\right)\right)^{-1} \cdot \mathbf{h}_m^I$, where μ and σ^2 are the corresponding mean and variance of this context domain. For example, in Fig. 2, the numerical domain $length$ is represented as $\mathbf{h}_m^I$, then 18 and 97 are normalized by a logistic function as $\left(1 + exp\left(-(18-\mu)/\sigma^2\right)\right)^{-1} \cdot \mathbf{h}_m^I$ and $\left(1 + exp\left(-(97-\mu)/\sigma^2\right)\right)^{-1} \cdot \mathbf{h}_m^I$.

Categorical set domain: When a user watches a movie with parents and children, this companion $c_m^i = \{parents, children\}$ is a context value in the categorical set domain $\mathcal{C}_m^I = \{friends, parents, children\}$. For this context value, we construct an indicator vector, where we normalize this vector for non-empty context values such that all values in the vector sum up to 1. Then we estimate a latent vector $\mathbf{h}_{m,*}^i$ for each categorical element $c_{m,*}^i$ in the domain $\mathcal{C}_m^I$. For example, in Fig. 2, watching a movie with $parents$ and $children$, the indicator vector is $\mathbf{z}^T = (0, 0.5, 0.5)$ and the categorical set domain $\{friends, parents, children\}$ is represented by $(\mathbf{h}_{m,1}^I; \mathbf{h}_{m,2}^I; \mathbf{h}_{m,3}^I)$. Then, the context value $\{parents, children\}$ can be computed as $\mathbf{h}_m^i = \mathbf{z}^T \cdot (\mathbf{h}_{m,1}^I; \mathbf{h}_{m,2}^I; \mathbf{h}_{m,3}^I)$.

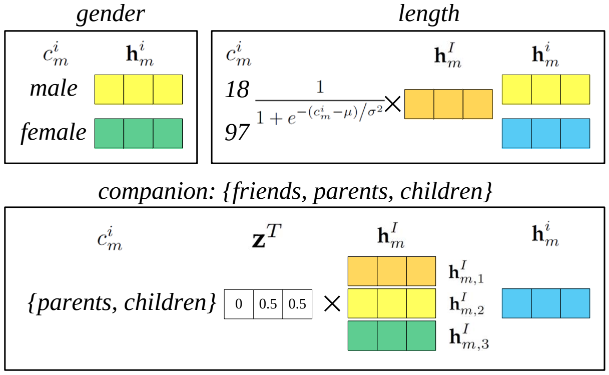


Fig. 2. Latent representations of context values in different domains. The context value in a categorical domain can be converted into a latent vector. Context values in one numerical domain have a shared latent vector and each value can be represented by a scalar multiplication. Each element in a categorical set is represented by a latent vector, and each context value can be represented by averaging latent vectors of elements in this context value.

In practical applications, various types of contexts fall into one of these domains mentioned above. For instance, the geographical location can be denoted by a numerical set $\{latitude, longitude\}$, where each element has its numerical domain. Some kinds of entity contexts, for instance, low-level features of image/text and social relations, can be transformed into feature vectors using machine learning techniques. Each value in the obtained feature vector can be treated as the value in a numerical domain.

3.3 Contextual Operating Matrix

In typical matrix factorization methods, latent vectors of users and items are constant with varying contexts. But in real-world applications, user interests and item properties are changed with varying contexts. Here, under different contexts, we provide context-specific latent vectors for users and items. Then, the rating prediction of matrix factorization based methods can be rewritten as:

$$\hat{r}_{u,v,c} = b_0 + b_u + b_v + \sum_{m=1}^{|c|} b_{c,m} + \mathbf{u}_c{}^T \mathbf{v}_c \ , \qquad (1)$$

where $|c| = |c^u| + |c^v| + \left|c^i\right|$ is the number of contexts, b_0 is the mean rating in training data, b_u and b_v denote the biases of user u and item v, $b_{c,m}$ is the bias of a context value. $\mathbf{u}_c$ and $\mathbf{v}_c$ are latent vectors of user u and item v under the contextual information c.

In a phrase of noun and adjective, the noun has semantic information and the adjective has semantic operation on the noun. In recommender systems, entities have rich semantic information and contexts act like adjectives which have the semantic operation on entities. For example, companion with children can change interests of a user, and he/she may tend to watch cartoons with children. During Valentine's Day, the latent characteristics of a romantic film may be changed and this film may become popular. We use contextual operating matrices to reveal how the contextual information c affects the properties of user/item. The context-specific latent vectors of users and items can be generated from their original ones.

$$\mathbf{u}_c = \mathbf{M}_c^U \mathbf{u} \ , \qquad (2)$$

$$\mathbf{v}_c = \mathbf{M}_c^V \mathbf{v} \ , \qquad (3)$$

where $\mathbf{u}$ and $\mathbf{v}$ are the original representations of the user and item, $\mathbf{M}_c^U$ and $\mathbf{M}_c^V$ are $d \times d$ contextual operating matrices of the contextual information c on users and items. These context-specific vectors can also be generated from other nonlinear function, e.g., the sigmoid function in Eq. 4 and 5, to assess the effectiveness of the COT framework. In the experimental section, we will show the results of these two computations.

$$\mathbf{u}_c = \left(1 + exp\left(-\mathbf{M}_c^U \mathbf{u}\ \right)\right)^{-1} - 0.5 \ , \qquad (4)$$

$$\mathbf{v}_c = \left(1 + exp\left(-\mathbf{M}_c^V \mathbf{v}\right)\right)^{-1} - 0.5 \ , \qquad (5)$$

Comparing with the interaction context combination which can operate the latent properties of users and items simultaneously, the user or item context combination does not have very similar operation on both users and items. For instance, a user context

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2016.2562621, IEEE Transactions on Knowledge and Data Engineering

$occupation$ can indicate the potential characteristics of users which may not have been revealed by the observed user-item interactions, but has slight influence in changing item properties. In this work, confronting with three kinds of contextual information, we plan to separate theirs effects.

Treating three context combinations separately, we can rewrite the contextual operating matrix as an operation combination. Serial multiplication may enlarge the defect of one context combination. Here, we resort to the linear operation combination, contextual operating matrices of users and items are denoted as

$$\mathbf{M}_c^U = \mathbf{M}_{c_u}^U + \mathbf{M}_{c_v}^U + \mathbf{M}_{c_i}^U, \tag{6}$$

$$\mathbf{M}_c^V = \mathbf{M}_{c_u}^V + \mathbf{M}_{c_v}^V + \mathbf{M}_{c_i}^V, \tag{7}$$

where $\mathbf{M}_{c_u}^U$, $\mathbf{M}_{c_v}^U$ and $\mathbf{M}_{c_i}^U$ are user-wise operation matrices of user context combination c^u, item context combination c^v and interaction context combination c^i, and $\mathbf{M}_{c_u}^V$, $\mathbf{M}_{c_v}^V$ and $\mathbf{M}_{c_i}^V$ are item-wise operation matrices of these context combinations. This linear operation combination not only can separate the effect of three context combinations but also can learn different weights of them implicitly. This combination can be replaced by other nonlinear combination. For example, using a sigmoid function, these two operation matrices can be denoted as follows.

$$\mathbf{M}_c^U = \left(1 + exp\left(-\mathbf{M}_{c_u}^U - \mathbf{M}_{c_v}^U - \mathbf{M}_{c_i}^U\right)\right)^{-1} - 0.5\ , \tag{8}$$

$$\mathbf{M}_c^V = \left(1 + exp\left(-\mathbf{M}_{c_u}^V - \mathbf{M}_{c_v}^V - \mathbf{M}_{c_i}^V\right)\right)^{-1} - 0.5, \tag{9}$$

3.4 Contextual Operating Tensor (COT)

We need two weighting matrices to map the latent matrix of a specific context combination into the operation matrices. For example, we need to estimate two matrices for each $\mathbf{H}^i$ and obtain operation matrices $\mathbf{M}_{c_i}^U$ and $\mathbf{M}_{c_i}^V$. The number of parameters will increase rapidly as the number of context combinations grows. Besides, since different contexts share similar semantic effects, for example, both $weekend$ and being at $home$ may make you would like to read novels. It will be wonderful and plausible if we can generate contextual operating matrices from several basic matrices (named operation tensor) which represent some common semantic effects of contexts. In this way, we can not only reduce the number of parameters to be estimated, but also model the underlying characteristics of contexts.

To employ the contextual operating tensor, we first should convert the latent matrix of context combination into a vector, and then the operating matrix can be generated from the multiplication of this vector with the operating tensor. Here, we show how to transform latent matrices of three context combinations into latent vectors as follows:

$$\mathbf{a}_{c_u}^U = \mathbf{H}^u \mathbf{w}_{C_U}^U,\ \mathbf{a}_{c_v}^U = \mathbf{H}^v \mathbf{w}_{C_V}^U,\ \mathbf{a}_{c_i}^U = \mathbf{H}^i \mathbf{w}_{C_I}^U,$$

$$\mathbf{a}_{c_u}^V = \mathbf{H}^u \mathbf{w}_{C_U}^V,\ \mathbf{a}_{c_v}^V = \mathbf{H}^v \mathbf{w}_{C_V}^V,\ \mathbf{a}_{c_i}^V = \mathbf{H}^i \mathbf{w}_{C_I}^V,$$

where each column of $\mathbf{H}^i$ denotes the latent vector of a context value, and $\mathbf{w}_{C_I}^U$ indicates the user-wise context weights on $\mathbf{H}^i$. The context combination vector $\mathbf{a}$ is a d_c dimensional latent vector, which is a weighted combination of context vectors.

We demonstrate the process of mapping the latent matrix of a context combination to vectors in Fig. 3. For each latent matrix $\mathbf{H}^i$, we estimate two weighting vectors $\mathbf{w}_{C_I}^U$ and $\mathbf{w}_{C_I}^V$ which indicate user-wise and item-wise weights on this context combination. Multiplying the latent matrix with the weighting vectors, we can obtain the user-specific and item-specific latent vectors $\mathbf{a}_{C_I}^U$ and $\mathbf{a}_{C_I}^V$ respectively. For example, we predict the rating value of Tom towards $Titanic$ under the contextual information of $theater$, $time$, $weather$, $companion$. The matrix $\mathbf{H}^i$ of these contexts is shown in the middle part of Fig. 3, and each context value is denoted as a column. Context weights in the left indicate the influences of four contexts on Tom and $Titanic$. The right part is context combination vectors for Tom and $Titanic$.

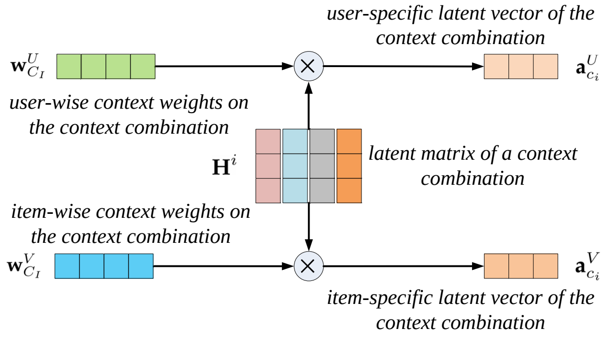

user-specific latent vector of the context combination
$\mathbf{w}_{C_I}^U$
$\mathbf{a}_{c_i}^U$
user-wise context weights on the context combination
$\mathbf{H}^i$
latent matrix of a context combination
item-wise context weights on the context combination
$\mathbf{w}_{C_I}^V$
$\mathbf{a}_{c_i}^V$
item-specific latent vector of the context combination

Fig. 3. The process of generating latent vectors of a context combination. The middle part is the latent matrix $\mathbf{H}^i$ of the context combination c^i. The left part is user-wise and item-wise context weights which indicate different influences of context values on users and items. The right part denotes the user-specific and item-specific latent vectors of this context combination.

After obtaining latent vectors of context combinations, operation matrices can be generated by multiplying the latent vectors of context combinations with the operating tensors. We use $\mathbf{T}^{U,[1:d]}$ and $\mathbf{T}^{V,[1:d]}$ to denote the operating tensors for users and items, and briefly write as $\mathbf{T}^U$ and $\mathbf{T}^V$ for simplicity. Here, incorporating contextual operating tensors for users and items, the operation matrices for users and items are

$$\mathbf{M}_c^U = \left(\mathbf{a}_{c_u}^U + \mathbf{a}_{c_v}^U + \mathbf{a}_{c_i}^U\right)^T \mathbf{T}^U$$

$$\mathbf{M}_c^V = \left(\mathbf{a}_{c_u}^V + \mathbf{a}_{c_v}^V + \mathbf{a}_{c_i}^V\right)^T \mathbf{T}^V$$

Since entity contexts have significantly different properties from interaction contexts, we would like

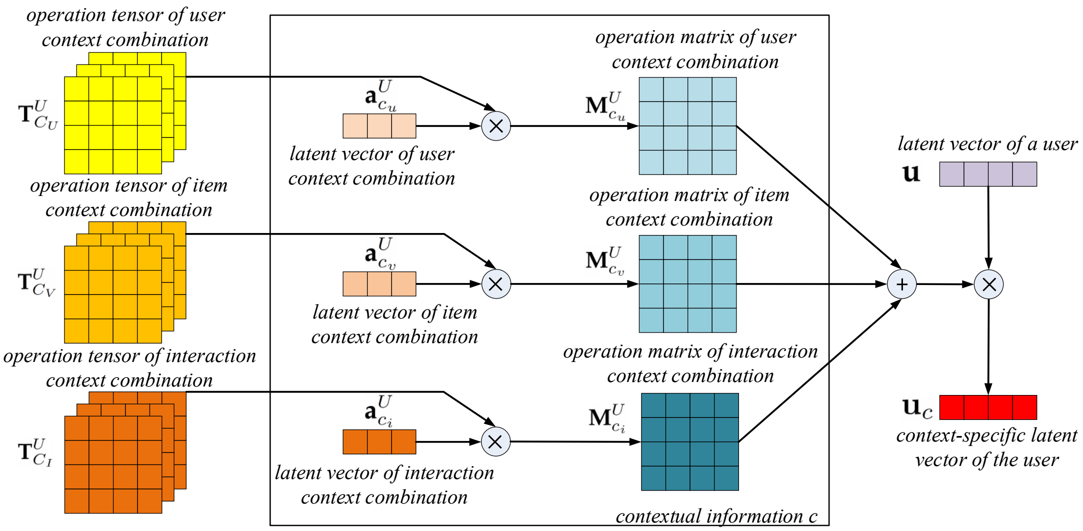

operation tensor of user context combination
$\mathbf{T}_{C_U}^U$
operation tensor of item context combination
$\mathbf{T}_{C_V}^U$
operation tensor of interaction context combination
$\mathbf{T}_{C_I}^U$
$\mathbf{a}_{c_u}^U$
latent vector of user context combination
$\mathbf{a}_{c_v}^U$
latent vector of item context combination
$\mathbf{a}_{c_i}^U$
latent vector of interaction context combination
operation matrix of user context combination
$\mathbf{M}_{c_u}^U$
operation matrix of item context combination
$\mathbf{M}_{c_v}^U$
operation matrix of interaction context combination
$\mathbf{M}_{c_i}^U$
contextual information c
latent vector of a user
$\mathbf{u}$
$\mathbf{u}_c$
context-specific latent vector of the user

Fig. 4. Overview of constructing the context-specific latent vector for a user. Contextual operating tensors are shown on the left side, and the process of generating the operating matrix is illustrated in the square.

to employ different operating tensors for three context combinations. We can rewrite the contextual operation matrices as follows.

$$\mathbf{M}_c^U = \left(\mathbf{a}_{c_u}^U\right)^T \mathbf{T}_{C_U}^U + \left(\mathbf{a}_{c_v}^U\right)^T \mathbf{T}_{C_V}^U + \left(\mathbf{a}_{c_i}^U\right)^T \mathbf{T}_{C_I}^U \ , \quad (10)$$

$$\mathbf{M}_c^V = \left(\mathbf{a}_{c_u}^V\right)^T \mathbf{T}_{C_U}^V + \left(\mathbf{a}_{c_v}^V\right)^T \mathbf{T}_{C_V}^V + \left(\mathbf{a}_{c_i}^V\right)^T \mathbf{T}_{C_I}^V \ , \quad (11)$$

where $\mathbf{T}_{C_U}^U$, $\mathbf{T}_{C_V}^U$ and $\mathbf{T}_{C_I}^U$ are $d_c \times d \times d$ tensors, denoting the operating tensors of three context combinations for users, $\mathbf{T}_{C_U}^V$, $\mathbf{T}_{C_V}^V$ and $\mathbf{T}_{C_I}^V$ are operating tensors for items. Substituting Equations (6-7) in Equations (2-3), we have:

$$\mathbf{u}_c = \left[\left(\mathbf{a}_{c_u}^U\right)^T \mathbf{T}_{C_U}^U + \left(\mathbf{a}_{c_v}^U\right)^T \mathbf{T}_{C_V}^U + \left(\mathbf{a}_{c_i}^U\right)^T \mathbf{T}_{C_I}^U\right] \mathbf{u} \ ,$$

$$\mathbf{v}_c = \left[\left(\mathbf{a}_{c_u}^V\right)^T \mathbf{T}_{C_U}^V + \left(\mathbf{a}_{c_v}^V\right)^T \mathbf{T}_{C_V}^V + \left(\mathbf{a}_{c_i}^V\right)^T \mathbf{T}_{C_I}^V\right] \mathbf{v} \ .$$

The operating tensor is composed of d slices, and each slice should be multiplied with the vector of context combination. Further, we write detailed equations of context-specific latent vectors of users and items.

$$\mathbf{u}_c = \begin{bmatrix} \left(\mathbf{a}_{c_u}^U\right)^T \mathbf{T}_{C_U,1}^U \mathbf{u} + \left(\mathbf{a}_{c_v}^U\right)^T \mathbf{T}_{C_V,1}^U \mathbf{u} + \left(\mathbf{a}_{c_i}^U\right)^T \mathbf{T}_{C_I,1}^U \mathbf{u} \\ \cdots \\ \left(\mathbf{a}_{c_u}^U\right)^T \mathbf{T}_{C_U,d}^U \mathbf{u} + \left(\mathbf{a}_{c_v}^U\right)^T \mathbf{T}_{C_V,d}^U \mathbf{u} + \left(\mathbf{a}_{c_i}^U\right)^T \mathbf{T}_{C_I,d}^U \mathbf{u} \end{bmatrix}$$

$$\mathbf{v}_c = \begin{bmatrix} \left(\mathbf{a}_{c_u}^V\right)^T \mathbf{T}_{C_U,1}^V \mathbf{v} + \left(\mathbf{a}_{c_v}^V\right)^T \mathbf{T}_{C_V,1}^V \mathbf{v} + \left(\mathbf{a}_{c_i}^V\right)^T \mathbf{T}_{C_I,1}^V \mathbf{v} \\ \cdots \\ \left(\mathbf{a}_{c_u}^V\right)^T \mathbf{T}_{C_U,d}^V \mathbf{v} + \left(\mathbf{a}_{c_v}^V\right)^T \mathbf{T}_{C_V,d}^V \mathbf{v} + \left(\mathbf{a}_{c_i}^V\right)^T \mathbf{T}_{C_I,d}^V \mathbf{v} \end{bmatrix}$$

where $\mathbf{T}_{C_I,m}^U$ and $\mathbf{T}_{C_I,m}^V$ are $d_c \times d$ matrices, denoting the mth slice of $\mathbf{T}_{C_I}^U$ and $\mathbf{T}_{C_I}^V$. Each slice captures a specific type of common semantic operation on users/items.

The process of generating the context-specific latent vector of a user is illustrated in Fig. 4. After generating the latent vector of contexts in Fig. 3, we can obtain the operation of these contexts by multiplying with a tensor. The operation matrix of these contexts should change the latent vector of user or item under these contexts. There are three operating tensors $\mathbf{T}_{C_U}^U$, $\mathbf{T}_{C_V}^U$ and $\mathbf{T}_{C_I}^U$. For a specific user-item interaction $r_{u,v,c}$, we use $\mathbf{a}_{c_u}^U$, $\mathbf{a}_{c_v}^U$ and $\mathbf{a}_{c_i}^U$to represent three context combinations. Multiplying these vectors of context combinations with the tensors, the corresponding operation matrices $\mathbf{M}_{c_u}^U$, $\mathbf{M}_{c_v}^U$ and $\mathbf{M}_{c_i}^U$ can be obtained, which are certain combinations of semantic operations in respective contextual operating tensors. The linear combination of three operation matrices $\mathbf{M}_{c_u}^U + \mathbf{M}_{c_v}^U + \mathbf{M}_{c_i}^U$ is used to change the original latent vector of user u, and then the context-specific latent vector $\mathbf{u}_c$ under the contextual information c can be yielded.

After discussing the generating process of context-specific vectors of users and items, the overall prediction function of COT can be written as:

$$\hat{r}_{u,v,c} = b_0 + b_u + b_v + \sum_{m=1}^{|c|} b_{c,m} +$$
$$\left\{\left[\left(\mathbf{a}_{c_u}^U\right)^T \mathbf{T}_{C_U}^U + \left(\mathbf{a}_{c_v}^U\right)^T \mathbf{T}_{C_V}^U + \left(\mathbf{a}_{c_i}^U\right)^T \mathbf{T}_{C_I}^U\right] \mathbf{u}\right\}^T$$
$$\left\{\left[\left(\mathbf{a}_{c_u}^V\right)^T \mathbf{T}_{C_U}^V + \left(\mathbf{a}_{c_v}^V\right)^T \mathbf{T}_{C_V}^V + \left(\mathbf{a}_{c_i}^V\right)^T \mathbf{T}_{C_I}^V\right] \mathbf{v}\right\}$$

Here, we give an example to describe the framework of COT illustrated in Fig. 4. Given Tom, $Titanic$ and contexts $theater$, $time$, $weather$, $companion$, we showed how to generate vectors of four interaction contexts for Tom and $Titanic$ in Fig. 3. In Fig. 4, we can compute the operation matrix of these vectors by multiplying with operation tensor in the left. Then, the latent vector of Tom under these contexts can be calculated by using operation matrix and their original latent vectors.

3.5 Parameter Inference

We have already introduced our model mathematically in the previous section. Now, to accomplish the parameter inference, we need to minimize the following objective function:

$$\begin{aligned}\min_{\mathbf{u},\mathbf{v},\mathbf{H},\mathbf{T},\mathbf{w}} J = & \sum_{\langle u,v,c\rangle\in\Omega} (r_{u,v,c} - \hat{r}_{u,v,c})^2 \\ & + \frac{\lambda}{2}({b_u}^2 + {b_v}^2 + \sum_{m=1}^{|c|} {b_{c,m}}^2 + \\ & ||\mathbf{u}||^2 + ||\mathbf{v}||^2 + ||\mathbf{H}||^2 + ||\mathbf{T}||^2 + ||\mathbf{w}||^2),\end{aligned} \tag{12}$$

where Ω denotes the training set, and λ is a parameter to control the regularizations, which can be determined using cross validation. The derivations of J with respect to all parameters can be calculated as:

$$\frac{\partial J}{\partial b_*} = -2l_{u,v,c} + \lambda b_* \ ,$$

$$\frac{\partial J}{\partial \mathbf{u}} = -2l_{u,v,c}\left(\mathbf{M}_c^U\right)^T\left(\mathbf{M}_c^V\mathbf{v}\right) + \lambda\mathbf{u} \ ,$$

$$\frac{\partial J}{\partial \mathbf{v}} = -2l_{u,v,c}\left(\mathbf{M}_c^U\mathbf{u}\right)\mathbf{M}_c^V + \lambda\mathbf{v} \ ,$$

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{H}_*} = & -2l_{u,v,c}\left(\mathbf{T}_{C_*}^U\mathbf{u}\right)\left(\mathbf{M}_c^V\mathbf{v}\right)\left(\mathbf{w}_{C_*}^U\right)^T \\ & + l_{u,v,c}\left(\mathbf{T}_{C_*}^V\mathbf{v}\right)\left(\mathbf{M}_c^U\mathbf{u}\right)\left(\mathbf{w}_{C_*}^V\right)^T + \lambda\mathbf{H}_* \ ,\end{aligned}$$

$$\frac{\partial J}{\partial \mathbf{w}_{C_U}^U} = -2l_{u,v,c}\mathbf{H}_u^T\left(\mathbf{T}_{C_U}^U\mathbf{u}\right)\left(\mathbf{M}_c^V\mathbf{v}\right) + \lambda\mathbf{w}_{C_U}^U \ ,$$

$$\frac{\partial J}{\partial \mathbf{w}_{C_V}^U} = -2l_{u,v,c}\mathbf{H}_v^T\left(\mathbf{T}_{C_V}^U\mathbf{u}\right)\left(\mathbf{M}_c^V\mathbf{v}\right) + \lambda\mathbf{w}_{C_V}^U \ ,$$

$$\frac{\partial J}{\partial \mathbf{w}_{C_I}^U} = -2l_{u,v,c}\mathbf{H}_i^T\left(\mathbf{T}_{C_I}^U\mathbf{u}\right)\left(\mathbf{M}_c^V\mathbf{v}\right) + \lambda\mathbf{w}_{C_I}^U \ ,$$

$$\frac{\partial J}{\partial \mathbf{w}_{C_U}^V} = -2l_{u,v,c}\left(\mathbf{M}_c^U\mathbf{u}\right)\mathbf{H}_u^T\left(\mathbf{T}_{C_U}^V\mathbf{u}\right) + \lambda\mathbf{w}_{C_U}^V \ ,$$

$$\frac{\partial J}{\partial \mathbf{w}_{C_V}^V} = -2l_{u,v,c}\left(\mathbf{M}_c^U\mathbf{u}\right)\mathbf{H}_v^T\left(\mathbf{T}_{C_V}^V\mathbf{u}\right) + \lambda\mathbf{w}_{C_V}^V \ ,$$

$$\frac{\partial J}{\partial \mathbf{w}_{C_I}^V} = -2l_{u,v,c}\left(\mathbf{M}_c^U\mathbf{u}\right)\mathbf{H}_i^T\left(\mathbf{T}_{C_I}^V\mathbf{u}\right) + \lambda\mathbf{w}_{C_I}^V \ ,$$

$$\frac{\partial J}{\partial \mathbf{T}_{*,m}^U} = -2l_{i,j,k}\mathbf{H}_u\mathbf{w}_*^U\mathbf{u}^T v_{c,m} + \lambda\mathbf{T}_{*,m}^U \ ,$$

$$\frac{\partial J}{\partial \mathbf{T}_{*,m}^V} = -2l_{i,j,k}\mathbf{H}_v\mathbf{w}_*^V\mathbf{v}^T u_{c,m} + \lambda\mathbf{T}_{*,m}^V \ ,$$

where b_* is a specific bias, $\mathbf{H}_*$ describes the latent matrix of a specific context combination, $\mathbf{T}_*^U$ is an operating tensor of a specific context combination for the user, and $\mathbf{T}_{*,m}^U$ is the mth slide of the operating tensor $\mathbf{T}_*^U$. $u_{c,m}$ and $v_{c,m}$ denotes the mth component of latent vector $\mathbf{u}_c$ and $\mathbf{v}_c$ respectively, and $l_{u,v,c} = r_{u,v,c} - \hat{r}_{u,v,c}$.

3.6 Optimization

After calculating all the derivations, a minimum solution of J in Equation 12 can be obtained by using stochastic gradient decent, which has been widely used in recommender systems [15][16]. Here, in Algorithm 1, we describe the optimization algorithm of our proposed COT model.

We propose an efficient learning algorithm (Algorithm 1) to optimize the objective function with the contextual operation. At first, all the parameters are initialized randomly in the range [-0.5, 0.5]. Then, we randomly choose a rating $r_{u,v,c}$ from the training set, and update all parameters using the derivations in the section of parameter inference. After the algorithm is convergent, the model parameters b, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{H}$, $\mathbf{T}$ and $\mathbf{w}$ are obtained, and the rating prediction $\hat{r}_{u,v,c}$ can be calculated using Equation 3.4. Note that γ is the learning rate, which can be determined through the cross validation. This optimization algorithm can be implemented without requiring significant change to conventional matrix factorization models.

Algorithm 1 Optimization Algorithm of COT

```
Input: The training set, each $r_{u,v,c}$ is associated with a user $u$, an item $v$ and contextual information $c$.
Output: Model parameters $b$, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{H}$, $\mathbf{T}$ and $\mathbf{w}$.
Initialize $b$, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{H}$, $\mathbf{T}$ and $\mathbf{w}$ randomly.
while not convergent do
    Select an instance $r_{u,v,c}$ from the training set.
    Calculate $\frac{\partial J}{\partial b}$, $\frac{\partial J}{\partial \mathbf{u}}$, $\frac{\partial J}{\partial \mathbf{v}}$, $\frac{\partial J}{\partial \mathbf{H}}$, $\frac{\partial J}{\partial \mathbf{T}}$, $\frac{\partial J}{\partial \mathbf{w}}$.
    Update $b \leftarrow b - \gamma\frac{\partial J}{\partial b}$.
    Update $\mathbf{u} \leftarrow \mathbf{u} - \gamma\frac{\partial J}{\partial \mathbf{u}}$.
    Update $\mathbf{v} \leftarrow \mathbf{v} - \gamma\frac{\partial J}{\partial \mathbf{v}}$.
    Update $\mathbf{H} \leftarrow \mathbf{H} - \gamma\frac{\partial J}{\partial \mathbf{H}}$.
    Update $\mathbf{T} \leftarrow \mathbf{T} - \gamma\frac{\partial J}{\partial \mathbf{T}}$.
    Update $\mathbf{w} \leftarrow \mathbf{w} - \gamma\frac{\partial J}{\partial \mathbf{w}}$.
end while
```

Based on the optimization algorithm, now we analyze the time complexity of training process. In each iteration, the time complexity of updating $\mathbf{u}$ and $\mathbf{v}$ are $O(d^2 \times |\Omega|)$, where $|\Omega|$ is the size of training dataset. The time complexity of updating $\mathbf{H}$ and $\mathbf{w}$ are $O(d_c \times d^2 \times |\Omega|)$, and the complexity of updating $\mathbf{T}$ is $O(d_c \times d \times |\Omega|)$. Therefore, the total time complexity of training process is $O(d_c \times d^2 \times |\Omega|)$. Since $|\Omega|$ is much larger than $d_c \times d^2$, the time complexity can be viewed as growing linearly with respect to the size of training dataset. Therefore, the time complexity of COT is very similar to that of the state-of-the-art CARS^2 and FM models, which both can be treated as linear with size

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2016.2562621, IEEE Transactions on Knowledge and Data Engineering

of training set. This time complexity also shows that COT has potential to scale up to large-scale data sets.

4 Experiment

In this section, we investigate the performance of the COT model. First, we describe the datasets, the comparison methods and experimental settings. Then we report and analyze experimental results, the convergence performance, the scalability analysis and impact of parameters. Last but not the least, we find some interesting observations on context representations and context weights.

4.1 Evaluation Datasets

Although the context-aware recommendation is a practical problem, there are only a few publicly available datasets. We investigate the performance of our proposed model on three benchmark datasets: the Food dataset [33], the Adom dataset [22] and the MovieLens-1M dataset[1].

- **Food Dataset** [33] is collected from a restaurant. There are two interaction contexts: $virtuality$ describes if the situation in which the user rates is $virtual$ or $real$, and $hunger$ captures how hungry the user is.
- **Adom Dataset** [22] is collected from a movie website and has rich contextual information. There are five interaction contexts: $companion$ captures whom the user watches the movie with, $when$ shows whether the user watches the movie at weekend, $release$ indicates whether the user watches the movie on the release weekend, rec captures how the user will recommend the movie, and $where$ indicates whether the user watches the movie in a theater.
- **Movielens-1M Dataset** is collected from a personalized movie recommender system[2]. There is no explicit contextual information, but the timestamp can be split into two interaction contexts: $hour$ and day. Besides, this dataset contains user and item contexts, i.e., $gender$, age and $occupation$ of the user and $title$ and $genre$ of the item.

With rich interaction contextual information, the Food and Adom datasets are widely used for the context-aware recommendation. The main difference between these two datasets lies on the amount of contextual information, which gives us an opportunity to estimate the relation between the model performance and the scale of contexts. The MovieLens-1M dataset is another widely used dataset [27], where the timestamp can be used as interaction contexts and attributes of users and items can be treated as entity contexts. We employ this dataset to examine the performance of methods in dealing with general contextual information.

4.2 Compared Methods

In this work, we compare the COT model with five state-of-the-art models:

- **SVD++** [17] is an advanced matrix factorization model for recommendation systems, but is not designed for the context-aware recommendation. We implement it as the baseline in our experiments.
- **Multiverse Recommendation** [5] is a state-of-the-art model which employs Tucker decomposition on the user-item-context rating tensor. This model outperforms conventional context-aware recommendation models, such as the pre-filtering and multidimensional approach [22].
- **FM** [2] is applicable for different kinds of contextual information by specifying the input data. We use LibFM[3] to implement this general method.
- **HeteroMF** [8] uses transfer matrices to model the interaction contexts. Each specific context combination has a transfer matrix.
- **CARS**^{2} [6] provides each user and item with a latent vector and a context-aware representation. The context-aware representation captures latent properties of the user and item manipulated by the contextual information.

Moreover, among these methods, FM and COT can handle general contextual information, and other methods only address the interaction contexts.

4.3 Evaluation Metrics

To measure the performance of rating prediction, we use the most popular metrics, Root Mean Square Error (RMSE) and Mean Average Precision (MAE):

$$RMSE = \sqrt{\frac{\sum\limits_{r_{u,v,c} \in \Omega_{test}} (r_{u,v,c} - \hat{r}_{u,v,c})^2}{n_{test}}} \, , \tag{13}$$

$$MAE = \frac{\sum\limits_{r_{u,v,c} \in \Omega_{test}} |r_{u,v,c} - \hat{r}_{u,v,c}|}{n_{test}} \, , \tag{14}$$

where Ω_{test} denotes the test set and n_{test} denotes the number of ratings in the test set. For these two metrics, the smaller the value, the better the performance.

4.4 Experimental Methodology

In the experiment, to assess the performance of comparison methods on all users and cold start users, we adopt two different ways in splitting the datasets.

All Users: We randomly sample about 10% ratings from the dataset to create the test set, and the remaining 90% ratings are used as the training set.

Cold Start: We randomly sample some users from the original dataset, then select less than three of

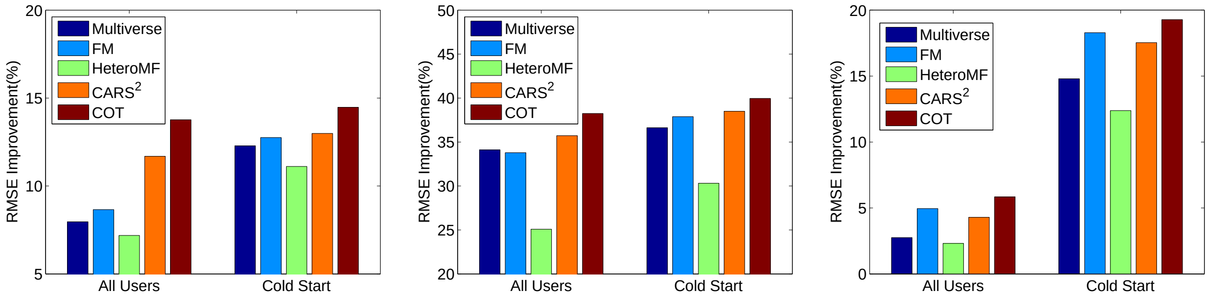
[1] http://grouplens.org/datasets/

[2] http://movielens.org/

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

TABLE 1
Performance comparison with RMSE and MAE on three datasets and two kinds of splitting ($d = 8$, $d_c = 4$).

	Food				Adom				Movielens-1M			
	All Users		Cold Start		All Users		Cold Start		All Users		Cold Start	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
SVD++	1.155	0.948	1.278	1.086	2.782	2.093	3.421	2.436	0.908	0.693	1.203	0.921
Multiverse	1.063	0.841	1.121	0.921	1.833	1.383	2.168	1.556	0.883	0.669	1.025	0.771
FM	1.055	0.845	1.115	0.918	1.842	1.426	2.125	1.563	0.863	0.661	0.983	0.752
HeteroMF	1.072	0.862	1.136	0.932	2.084	1.552	2.384	1.782	0.887	0.677	1.054	0.806
CARS^2	1.020	0.807	1.112	0.911	1.788	1.372	2.104	1.538	0.869	0.664	0.992	0.758
COT	**0.996**	**0.791**	**1.093**	**0.896**	**1.718**	**1.364**	**2.054**	**1.515**	**0.855**	**0.654**	**0.971**	**0.743**



(a) Food (b) Adom (c) Movielens-1M

Fig. 5. RMSE improvements (%) of corresponding context-aware methods over the context-unaware method SVD++ on three datasets.

their ratings as the training set, and use all remaining ratings as the test set. The numbers of ratings of each user in the test set are randomly decided. Also, the test set covers about 10% of the original dataset, and the training set covers about 90%.

Moreover, in all experiments, the training set is further split into five parts, and the model parameters can be better determined by using 5-fold cross validation.

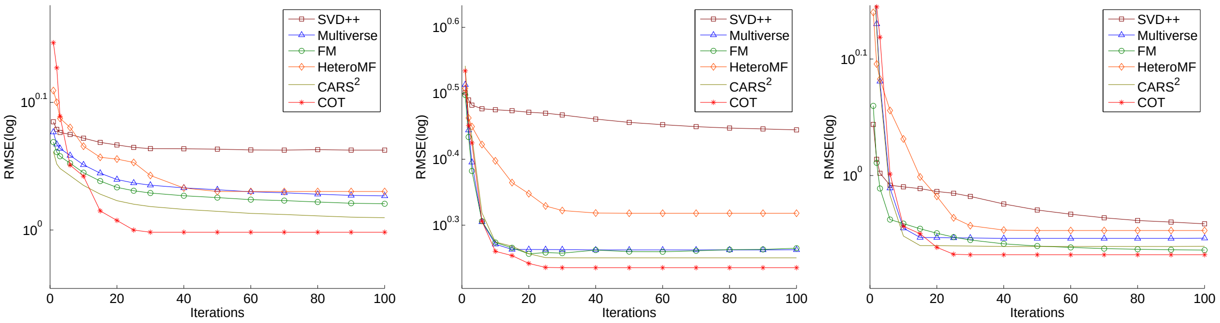
4.5 Performance Comparison

Table 1 illustrates experimental results measured by RMSE and MAE on three datasets and two kinds of splitting. We identify that through all the experiments, context-aware models outperform the context-unaware model SVD++. It demonstrates the importance of utilizing the contextual information in recommender systems. This table also shows that COT achieves the best results consistently. It is because that using context representation and contextual operating tensor to model contextual information is very effective. Comparing with the results of CARS^2, the better performance of COT is due to the powerful representative ability of distributed representations of context values. Moreover, on Movielens-1M with entity contexts, FM gets slightly better results than CARS^2. This evidence shows that user/item contexts can provide additional information which cannot be revealed by interactions and interaction contexts. Since CARS^2 only can deal with interaction contexts, FM utilizing interaction contexts and entity contexts simultaneously can outperform CARS^2 on this dataset.

In terms of all users splitting, comparing with the best performance of other models, COT improves the RMSE values by 2.4%, 3.9% and 1.0% on the Food, Adom and Movelens-1M datasets respectively. In terms of cold start splitting, the improvements become 1.7%, 2.4% and 1.2% accordingly. The improvement of MAE has very similar trend as that of RMSE. Among three datasets, COT has the greatest improvement on Adom, which shows COT to be particularly helpful for the dataset with rich contextual information. FM and Multiverse Recommendation achieve very similar performance on Food and Adom, but on Movielens FM has great improvement over the Multiverse recommendation. This improvement indicates that user/item contexts are important in the contextual modeling. HeteroMF performs close to Multiverse Recommendation on Food and Movielens-1M, but fails on Adom. This may be because Adom has richer contextual information than the other datasets, and HeteroMF needs to estimate too many transfer matrices for great amount of contextual information.

We illustrate the RMSE improvements of corresponding context-aware models over the context-unaware method SVD++ in Fig. 5. On all three datasets, the RMSE improvements on cold start splitting are larger than those on all users splitting. It shows that the contextual information is more important in the cold start situation and can be used to compensate for the lack of history information. Since Adom has the greatest amount of contextual information among three datasets, RMSE improvements on Adom (more than 30% in average) are greater than

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2016.2562621, IEEE Transactions on Knowledge and Data Engineering



(a) Food (b) Adom (c) Movielens-1M

Fig. 6. Convergence curves of comparison models on three datasets. The x-axis is the number of iterations, and y-axis shows the values of RMSE(log).

those on the others (about 10% in average). These abundant contexts are helpful for model construction and significantly enhance the recommendation performance. Moreover, on Movielens, the improvement on cold start splitting is about three times more than the improvement on all users splitting. As the rating number of Movielens is larger than those of other datasets, the advanced context-unaware method SVD++ can be trained more sufficiently and obtain better results. When the rating number decreases significantly, contextual modeling becomes important on the cold start users. This observation reveals the context modeling is really helpful for the cold start users in real applications.

Here, we compare and analyze the performance difference between COT and two variants introduced in Section 3.3. The variant with a nonlinear transformation of context-specific latent vector is named as COT_V1, and the variant employing a nonlinear combination of different contexts is denoted as COT_V2. The experimental performance on all users splitting of three datasets is list in Table 2. The results of COT and two variants are very similar. It may be because that the contextual operation can well reveal the underlying properties of contextual information. Then, under this framework, different strategies of transformation or combination just have a slight effect on the final performance.

TABLE 2
Performance comparsion of COT and two variants.

	Food		Adom		Movielens-1M	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
COT	0.996	0.791	1.718	1.364	0.855	0.654
COT_V1	0.999	0.793	1.726	1.365	0.857	0.655
COT_V2	0.997	0.792	1.723	1.365	0.856	0.655

4.6 Convergence Analysis

The convergence curves of comparison methods on three datasets are illustrated in Fig. 6. It shows that RMSE of COT becomes stable after about 30 iterations. These evidences indicate that COT has a satisfying convergence rate and can be trained rapidly and efficiently in practical applications. HeteroMF also shows its ability in convergence and the performance becomes stable after convergence. SVD++ converges slowly, and convergence curves become stable on Adom and Movielens-1M until the number of iterations reaches about 100. The performance of FM using SGD is very effective, but we also observe that FM needs numerous iterations to obtain convergent situations on Food and Movielens-1M. On Food, CARS^2 and Multiverse need more iterations to obtain a stable result. It may be because that CARS^2 and Multiverse recommendation have many parameters to be estimated and are more likely to overfit this kind of sparse dataset.

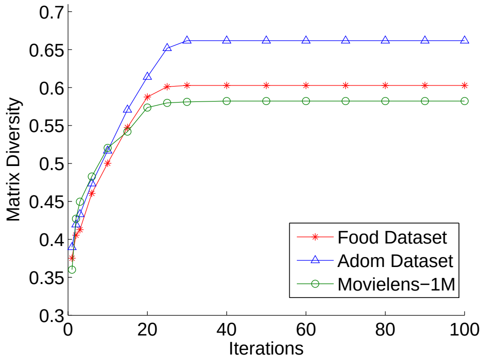


Fig. 7. Matrix diversities of operating tensors with the increasing number of iterations on three datasets.

As we discussed above, each slice of the contextual operating tensor represents one kind of common operation. With the larger difference among these slices, the contextual operating tensor is more powerful in modeling the contextual operation. Similar to the content diversity measuring the difference among contents of movies [34], we use a metric *matrix diversity* to measure the difference among all slices. Matrix diversity is calculated as average RMSE of all slice pairs in each tensor. Fig. 7 illustrates how matrix diversities of contextual operating tensors change with the increasing number of iterations. It indicates that matrix diversities are increasing when the number of iterations grows from 1 to about 30. After 30 iterations, matrix diversities become stable. We find that COT achieves convergence in Fig. 6 at the same time as the matrix diversity converged in Fig. 7. These evidences indicate that when the matrix diversity achieves stable results, COT can obtain the best performance. Moreover, the stable value of matrix diversity on Adom is the largest one. This may be because rich contextual information on Adom has the powerful operating ability in changing the properties of users and items. This clue also confirms the evidence in Fig. 5, context-aware methods have the greatest RMSE improvement over the context-unaware SVD++ on Adom.

4.7 Scalability Analysis

Besides the analysis of convergence rate, we also investigate the scalability of the COT method with varying portion of training data. Here, we implement COT on three datasets and for each dataset we measure the corresponding time cost of one iteration in the training process Fig. 8 shows that on three datasets, time consumptions of our method is linear with respect to the size of training data. This result empirically confirms the analysis of time complexity in Section 3.6. Comparing with the state-of-the-arts, i.e., the CARS^2 and FM methods, time consumptions of these three methods have very similar trend, they all can be viewed as linearly increasing with the size of training set. These empirical results and analysis of time complexity show that our proposed method is very efficient and can provide a reasonable scalability for real applications.

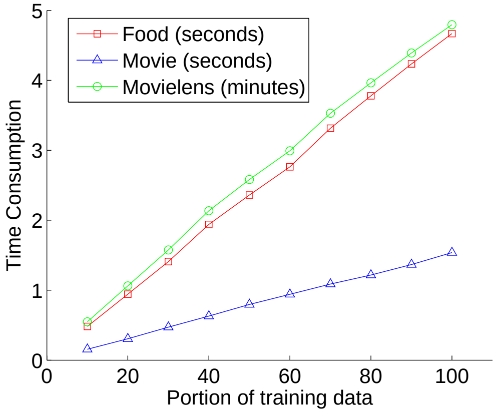


Fig. 8. Time consumption of COT with varying portion of training data on three datasets.

4.8 Distribution of Context Representation

In this section, we demonstrate distributed representations of context values, and observe the potential relation among these context values.

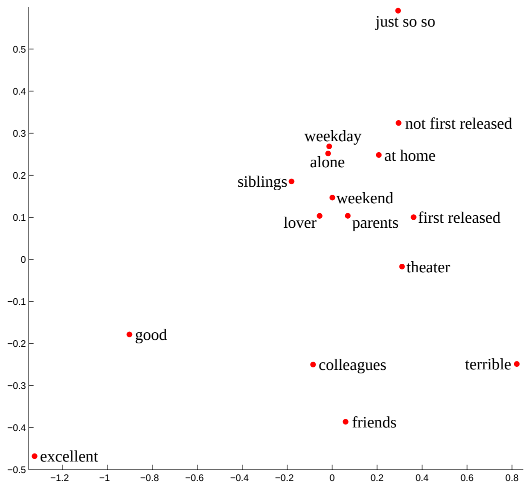


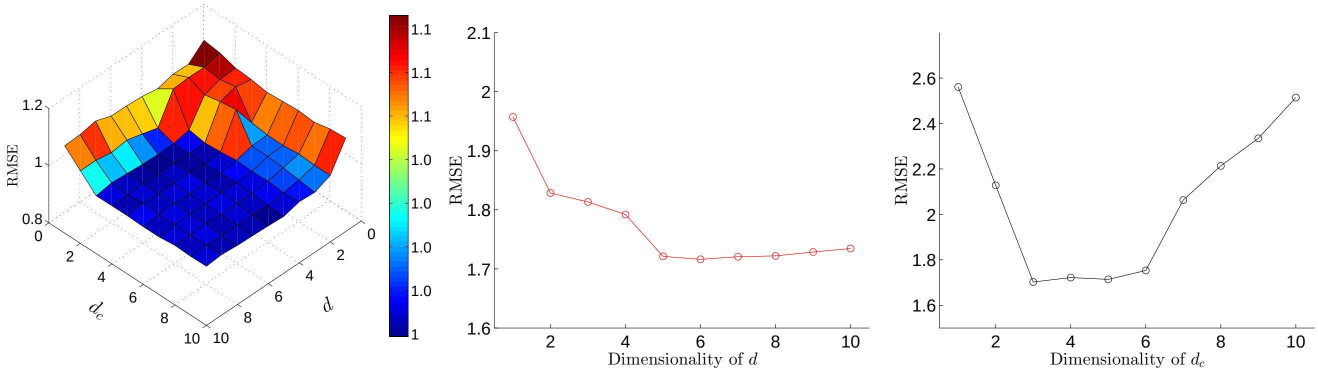
Fig. 9. Demonstration of distributed representations of context values in a two dimensional space using PCA.

We use Principal Component Analysis (PCA) to project representations of context values on Adom into two dimensional vectors, and these context values are illustrated in Fig. 9. The distance reveals the potential relation among these context values. On $weekend$s, we may watch a movie in companion with $lover$ and $parents$, and on $weekday$s we are more likely to watch $alone$. This observation follows our intuition. If a person is $alone$, he is more likely to watch a movie at $home$. If a person is with $lover$ or $parents$, he tends to watch a movie in a $theater$. When a movie is on its $released$ weekend, we can watch it in a $theater$, and if a movie has been released for a long time, we may watch it at $home$. Besides, we see that the $colleagues$ is close to $friends$, $lover$ and $parents$ are close to $siblings$. $colleagues$ and $friends$ seem far away from other context values, which reveals that a person rarely watches a movie with $colleagues$ or $friends$. Moreover, we find that $excellent$ is an outlier, and the majority of movies are $good$, $just\ so\ so$ or $terrible$. $excellent$ is in the same direction of $good$. These observations are interesting and follow our intuition, and the context representation of COT provides us an opportunity to examine the potential relation of these context values.

4.9 Context Weights

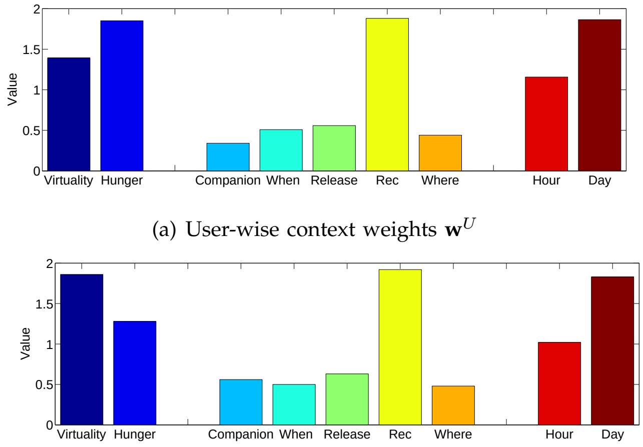
On Food, Adom and Movielens, the user-wise and item-wise context weights $\mathbf{w}^U$ and $\mathbf{w}^V$ are demonstrated in Fig. 9. Fig. 9 (a) is the user-wise context weights and Fig. 9 (b) is the item-wise context weights. The major difference between these two figures is the context weights on the Food dataset. When a user rates one kind of food, for the user, how $hungry$ is more relevant to the rating results than whether the situation is $virtual$ or $real$. For the item, $virtual$ or $real$ is more significant than the hunger degree of the user. These are because the degree of $hungry$ is

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2016.2562621, IEEE Transactions on Knowledge and Data Engineering



(a) Dimensionality d and d_c on Food (b) Dimensionality of entity vector d (c) Dimensionality of context vector d_c

Fig. 11. Performance of COT measured by RMSE on Food and Adom with varying dimensionalities of entity vector d and context vector d_c. (a) shows RMSE values on Food with varying dimensionalities of d and d_c, (b) illustrates RMSE values on Adom with the dimensionality of d, and (c) indicates RMSE values on Adom with the dimensionality of d_c.



(a) User-wise context weights $\mathbf{w}^U$

(b) Item-wise context weights $\mathbf{w}^V$

Fig. 10. User-wise and item-wise context weights on three experimental datasets, i.e., Food, Adom and Movielens-1M. Fig. (a) shows user-wise context weights $\mathbf{w}^U$ and Fig. (b) illustrates item-wise context weights $\mathbf{w}^V$.

describing situations of users and has more important effect on users, while $virtuality$ describes different situations of rating events, and $real$ or $virtual$ is important in revealing the properties of items. On the Adom dataset, the context rec has the highest weight and becomes the dominant context. This may be because the rec context, indicating how the user will recommend the movie, has high relevance with the final rating. On Adom, we also observe that a slight difference between user-wise and item-wise weights is the $companion$ context. It may be because with different $companion$ a person can choose different kinds of movies to watch, but some movies are pictured for specific kinds of audiences. For example, the cartoon movie is for parents with children and the romance movie is for a person with his/her lover. The figure also demonstrates that on Movielens-1M day is more important than $hour$. It shows that the context day has more discrimination ability than the context $hour$ on the user behavior of watching movie.

4.10 Impact of Parameters

Here, we first assess the COT model with respect to varying dimensionalities of entity vector d and context vector d_c on the Food dataset. Fig. 10 (a) shows that with increasing d and d_c, the RMSE value decreases at first, then stays nearly stable after $d = 5$ and $d_c = 3$. These observations indicate that the parameter d and d_c can be selected in a large range on Food, and the performance of COT does not rely on the parameter selection very much.

In Fig. 10 (b) and (c) we further illustrate the RMSE values on Adom w.r.t. d and d_c respectively. We set $d_c = 4$ and calculate the RMSE values with the varying dimensionality of d, and the results are demonstrated in Fig. 10 (b). Then we fix $d = 8$ and evaluate the performance of COT with d_c in Fig. 11 (c). Fig. 11 (b) indicates that the performance of COT is improving with the increase of dimensionality d generally. When the dimensionality d is larger than 5, the COT method can obtain decent results. This observation of the entity vector dimensionality on Adom is very similar to the performance on Food in Fig. 10 (a). On the other hand, the performance of COT is changed greatly with the dimensionality of context vector d_c. The best performance is obtained when the dimensionality d_c is in the range $[3, 6]$. When we increase the dimensionality d_c, the performance of COT is decreasing. Since the Adom dataset has abundant contextual information but a small number of training ratings, the context representation with high dimensionality is prone to overfitting.

From the experimental results in Fig. 10, we observe that the dimensionality of context vector d_c should be selected in a range with small values, and the performance of COT will be very stable. Since the performances of COT with parameter values selected from these ranges are very similar, we only illustrate the results with $d = 8$ and $d_c = 4$ on three datasets for simplicity.

5 Conclusion

In this work, a novel context-aware recommendation method, i.e., COT, has been proposed. We provide each context value with a continuous vector, which is a distributed representation different from the one-hot representation in FM and other methods. Such distributed representations have a powerful ability in describing the semantic operation of context values. Similar to the semantic composition in NLP where the adjective has an operation on the noun, we provide the contextual information of each rating event with a semantic operation matrix, which can be used to generate new vectors of users and items under this contextual situation. At the same time, the common semantic effects of contexts can be captured by contextual operating tensors. Then the contextual operating matrix can be calculated from the contextual operating tensor and context representations. The experimental results on three real datasets show that COT outperforms state-of-the-art context-aware models. From experimental results, we observe that the potential relation among the context values is interesting and follows our intuition. And context weights of COT can be used to explain the importance of context values in changing vectors of users and items.

In the future, we would like to introduce a pairwise ranking constraint on the contextual information. A user-item interaction can be generated under specific contextual information but cannot be yielded under other contextual situations. This kind of pairwise ranking constraint reveals the relative information among different contextual situations and can be used to further enhance context modeling. Moreover, s-ince the top-n recommendation is another significant measurement of recommender systems, analyzing the ranking performance of the COT framework will be a very interesting issue in future.

References

[1] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender Systems Handbook*. Springer, 2011, pp. 217–253.

[2] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme, "Fast context-aware recommendations with factorization machines," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2011, pp. 635–644.

[3] D. H. Stern, R. Herbrich, and T. Graepel, "Matchbox: Large scale online bayesian recommendations," in *Proceedings of the 18th International Conference on World Wide Web*. , 2009, pp. 111–120.

[4] D. Agarwal and B.-C. Chen, "Regression-based latent factor models," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2009, pp. 19–28.

[5] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering," in *Proceedings of the Fourth ACM Conference on Recommender Systems*. ACM, 2010, pp. 79–86.

[6] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, and A. Hanjalic, "Cars2: Learning context-aware representations for context-aware recommendations," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 2014, pp. 291–300.

[7] L. Zhang, D. Agarwal, and B.-C. Chen, "Generalizing matrix factorization through flexible regression priors," in *Proceedings of the Fifth ACM Conference on Recommender Systems*. ACM, 2011, pp. 13–20.

[8] M. Jamali and L. Lakshmanan, "Heteromf: recommendation in heterogeneous information networks using context dependent factor models," in *Proceedings of the 22nd International Conference on World Wide Web*. 2013, pp. 643–654.

[9] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings on Neural Information Processing Systems*, 2013.

[10] S. Rendle, "Factorization machines with libfm," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 3, no. 3, p. 57, 2012.

[11] M. Baroni and R. Zamparelli, "Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2010, pp. 1183–1193.

[12] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "Semantic compositionality through recursive matrix-vector spaces," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 2012, pp. 1201–1211.

[13] 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 *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2013, pp. 1631–1642.

[14] A. Mnih and R. Salakhutdinov, "Probabilistic matrix factorization," in *Proceedings on Neural Information Processing systems*, 2007.

[15] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[16] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender Systems Handbook*. Springer, 2011, pp. 145–186.

[17] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2008, pp. 426–434.

[18] Y. Koren, "Collaborative filtering with temporal dynamics," *Communications of the ACM*, vol. 53, no. 4, pp. 89–97, 2010.

[19] L. Xiong, X. Chen, T.-K. Huang, J. G. Schneider, and J. G. Carbonell, "Temporal collaborative filtering with bayesian probabilistic tensor factorization." in *Proceedings of the 2010 SIAM International Conference on Data Mining*. SIAM, 2010, pp. 211–222.

[20] T. Chen, Z. Zheng, Q. Lu, W. Zhang, and Y. Yu, "Feature-Based Matrix Factorization," *ArXiv e-prints*, Sep. 2011.

[21] C. Palmisano, A. Tuzhilin, and M. Gorgoglione, "Using context to improve predictive modeling of customers in personalization applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 11, pp. 1535–1549, 2008.

[22] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, "Incorporating contextual information in recommender systems using a multidimensional approach," *ACM Transactions on Information Systems (TOIS)*, vol. 23, no. 1, pp. 103–145, 2005.

[23] L. Baltrunas and F. Ricci, "Context-based splitting of item ratings in collaborative filtering," in *Proceedings of the Third ACM Conference on Recommender Systems*. ACM, 2009, pp. 245–248.

[24] U. Panniello, A. Tuzhilin, M. Gorgoglione, C. Palmisano, and A. Pedone, "Experimental comparison of pre-vs. post-filtering

approaches in context-aware recommender systems," in *Proceedings of the Third ACM Conference on Recommender Systems*. ACM, 2009, pp. 265–268.

[25] Y. Li, J. Nie, Y. Zhang, B. Wang, B. Yan, and F. Weng, "Contextual recommendation based on text mining," in *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 692–700.

[26] E. Zhong, W. Fan, and Q. Yang, "Contextual collaborative filtering via hierarchical matrix factorization," in *Proceedings of the 2012 SIAM International Conference on Data Mining*. SIAM, 2012, pp. 744–755.

[27] X. Liu and K. Aberer, "Soco: a social network aided context-aware recommender system," in *Proceedings of the 22nd International Conference on World Wide Web*. 2013, pp. 781–802.

[28] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[29] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2008, pp. 650–658.

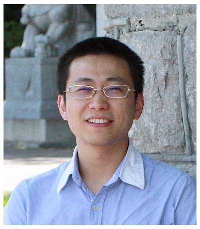
[30] S.-H. Yang, B. Long, A. Smola, N. Sadagopan, Z. Zheng, and H. Zha, "Like like alike: joint friendship and interest propagation in social networks," in *Proceedings of the 20th International Conference on World Wide Web*. 2011, pp. 537–546.

[31] C. Lippert, S. H. Weber, Y. Huang, V. Tresp, M. Schubert, and H.-P. Kriegel, "Relation prediction in multi-relational domains using matrix factorization," in *Workshops of Neural Information Processing Systems: Structured Input-Structured Output*, 2008.

[32] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[33] 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 *User Modeling, Adaptation, and Personalization*. Springer, 2009, pp. 102–113.

[34] T. T. Nguyen, P.-M. Hui, F. M. Harper, L. Terveen, and J. A. Konstan, "Exploring the filter bubble: the effect of using recommender systems on content diversity," in *Proceedings of the 23rd international conference on World Wide Web*. 2014, pp. 677–686.



Shu Wu received his B.S. degree from Hunan University, China, in 2004, M.S. degree from Xiamen University, China, in 2007, and his Ph.D. degree from Department of Computer Science, University of Sherbrooke, Quebec, Canada, all in computer science. He is an Assistant Professor in Center for Research on Intelligent Perception and Computing (CRIPAC) at National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA). He has published more than 20 papers in the areas of data mining and information retrieval at international journals and conferences, such as IEEE TKDE, AAAI, SIGIR, CIKM. His research interests include data mining, information retrieval and recommendation systems.

Qiang Liu received his B.S. degree in electronic science from Yanshan University, China, in 2013. He is currently working toward the Ph.D. degree in National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include machine learning, data mining and recommendation systems.

Liang Wang received both the B. Eng. and M. Eng. degrees from Anhui University in 1997 and 2000 respectively, and the PhD degree from the Institute of Automation, Chinese Academy of Sciences (CAS) in 2004. From 2004 to 2010, he has been working as a Research Assistant at Imperial College London, United Kingdom and Monash University, Australia, a Research Fellow at the University of Melbourne, Australia, and a lecturer at the University of Bath, United Kingdom, respectively. Currently, he is a full Professor of Hundred Talents Program at the National Lab of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, P. R. China.

His major research interests include machine learning, pattern recognition and computer vision. He has widely published at highly-ranked international journals such as IEEE TPAMI and IEEE TIP, and leading international conferences such as CVPR, ICCV and ICDM. He has obtained several honors and awards such as the Special Prize of the Presidential Scholarship of Chinese Academy of Sciences. He is currently a Senior Member of IEEE, as well as a member of BMVA. He is an associate editor of IEEE Transactions on Cybernetics, International Journal of Image and Graphics, Neurocomputing and so on. He is a guest editor of 9 special issues, a co-editor of 6 edited books, co-program chairs of 3 conferences, and a co-chair of 12 international workshops.

Tieniu Tan He received his BSc degree in electronic engineering from Xi'an Jiaotong University, China, in 1984, and his MSc and PhD degrees in electronic engineering from Imperial College London, U.K., in 1986 and 1989, respectively. In October 1989, he joined the Computational Vision Group at the Department of Computer Science, The University of Reading, Reading, U.K., where he worked as a Research Fellow, Senior Research Fellow and Lecturer. In January 1998, he returned to China to join the National Laboratory of Pattern Recognition (NLPR), Institute of Automation of the Chinese Academy of Sciences (CAS), Beijing, China, where he is currently Professor and former director (1998-2013) of the NLPR and Center for Research on Intelligent Perception and Computing (CRIPAC), and was Director General of the Institute (2000-2007).He is currently also Deputy Secretary-General of the Chinese Academy of Sciences. He has published 11 edited books or monographs and more than 400 research papers in refereed international journals and conferences in the areas of image processing, computer vision and pattern recognition. His H-index is 61 (as of December 2014). His current research interests include biometrics, image and video understanding, and information content security.

Dr Tan is a Fellow of CAS, TWAS (The World Academy of Sciences for the advancement of science in developing countries), IEEE and IAPR (the International Association of Pattern Recognition), and an International Fellow of the UK Royal Academy of Engineering. He has served as chair or program committee member for many major national and international conferences. He is or has served as Associate Editor or member of editorial boards of many leading international journals including IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on Automation Science and Engineering, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Circuits and Systems for Video Technology, Pattern Recognition, Pattern Recognition Letters, Image and Vision Computing, etc. He is Editor-in-Chief of the International Journal of Automation and Computing. He was founding chair of the IAPR Technical Committee on Biometrics, the IAPR-IEEE International Conference on Biometrics, the IEEE International Workshop on Visual Surveillance and Asian Conference on Pattern Recognition (ACPR). He currently serves as President of the IEEE Biometrics Council and Deputy President of Chinese Artificial Intelligence Association. He has given invited talks and keynotes at many universities and international conferences, and has received many national and international awards and recognitions.