
Article:
Bai, P., Ge, Y., Liu, F. et al. (1 more author) (2019) Joint interaction with context operation for collaborative filtering. Pattern Recognition, 88. pp. 729-738. ISSN 0031-3203

https://doi.org/10.1016/j.patcog.2018.12.003

Joint Interaction with Context Operation for Collaborative Filtering

Peizhen Bai, Yan Ge, Fangling Liu, Haiping Lu*

Department of Computer Science, The University of Sheffield, 211 Portobello, Sheffield S1 4DP, United Kingdom

Abstract

In recommender systems, the classical matrix factorization model for collaborative filtering only considers joint interactions between users and items. In contrast, context-aware recommender systems (CARS) use contexts to improve recommendation performance. Some early CARS models treat user, item and context equally, unable to capture contextual impact accurately. More recent models perform context operations on users and items separately, leading to "double-counting" of contextual information. This paper proposes a new model, Joint Interaction with Context Operation (JICO), to integrate the joint interaction model with the context operation model, via two layers. The joint interaction layer models interactions between users and items via an interaction tensor. The context operation layer captures contextual information via a contextual operating tensor. We evaluate JICO on four datasets and conduct novel studies, including varying contextual influence and time split recommendation. JICO consistently outperforms competing methods, while providing many useful insights to assist further analysis.

Keywords: Recommender system, collaborative filtering, matrix factorization, context aware, joint interaction, tensor

*Corresponding author. Address: The University of Sheffield, 211 Portobello, Sheffield, S1 4DP, United Kingdom. Tel: +44 (0) 114 222 1853. Email: h.lu@sheffield.ac.uk

Email addresses: peizhen_bai@foxmail.com (Peizhen Bai), yge5@sheffield.ac.uk (Yan Ge), lfl7824833@foxmail.com (Fangling Liu), h.lu@sheffield.ac.uk (Haiping Lu)

1. Introduction

Collaborative filtering (CF) is popular for recommender systems in many applications such as social networks, e-commerce, and music or movie websites [1, 2, 3, 4, 5, 6, 7]. Typically, CF recommends *items* to *users* by learning user/item similarities from existing *ratings*. A classical CF model is Matrix Factorization (MF) [8, 9, 10, 11], which is a latent factor model [12]. It decomposes the rating matrix into two factor matrices, representing latent factors for users and items respectively. The predicted rating r_{ij} of user i on item j can be viewed as the *joint interaction* between latent factors of user i and item j. Another formulation of the recommender system is to view CF as a matrix completion problem [13, 14, 15, 16].

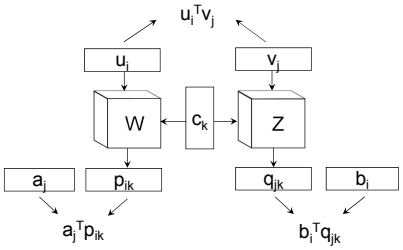
The MF model considers only two *entities* (user and item) involved. In contrast, context-aware recommender systems (CARS) aim to achieve better recommendation performance by taking useful *context* information, such as time, location, user age, gender and occupation, into account [17, 18, 19, 20, 21, 22, 23, 24]. For instance, time is an important factor in an e-commerce recommender system. Supposing a user bought a T-shirt in the summer, recommending a T-shirt in the winter would be inconsequential. The term "context" is broadly defined in this paper. We consider an attribute such as gender, occupation, and age as a particular context, following [25, 26, 27]. E.g. both occupation and age can change over time in real applications, such as from students to engineers, academia to industry, and getting older year by year.

Karatzoglou *et al.* [25] proposed a Multiverse model to generalize the MF model directly to incorporate contexts. It is based on the Higher-Order Singular Value Decomposition (HOSVD) [28], a Tensor Factorization (TF) model called the Tucker Decomposition [29]. Multiverse generalizes the MF model to higher order by extending the 2-D user $\times$ item matrix with additional context dimensions into a higher-order data tensor [30], e.g., a user $\times$ item $\times$ time tensor. This provides good flexibility in incorporating contextual information, but its computational complexity increases exponentially with the number of con-

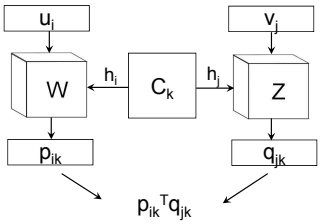
text variables. Instead of extending the rating matrix to a tensor, Rendle [31] developed a Factorization Machine (FM) model for context-aware recommendation. The FM model extends the latent factor model in MF by transforming the user-item-context rating data into real-valued feature vectors via one-hot encoding [32]. Then, it treats user-item, user-context, and item-context interactions as inner products of their latent factors, which can improve the rating computation to linear time. There are several more complex models developed based on FM, such as the Gaussian Process FM [33], random partition FM [27], ensemble-based FM [27] and hierarchical FM [34].

Entities (user and item) and contexts play different roles in recommender systems. However, all the CARS models above treat contexts as dimensions similar to user or item, making it difficult to interpret the relationships between entities and context, and also limiting their recommendation performance. This motivates the CARS^2 model [35] as depicted in Fig. 1(a). It defines the contextual information in its own latent space $\mathbf{c}_k$, and treats it differently from user and item latent spaces. Instead of a direct interaction between entities and contexts, CARS^2 transforms the original latent factors of users/items into a new latent space ($\mathcal{W}/\mathcal{Z}$), which represents user/item under contexts by a 3-D contextual operating tensor. This provides a clear interpretation of the relationship between entities and contexts.

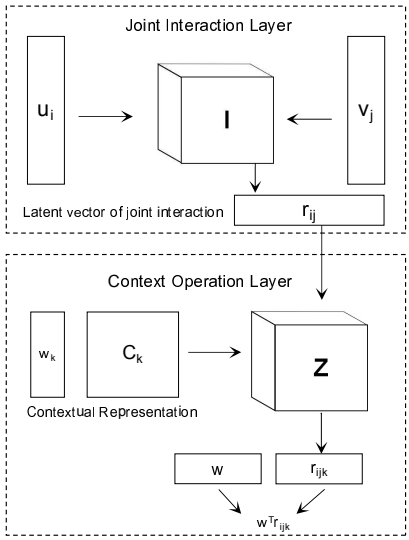
One limitation of CARS^2 is that it uses one vector $\mathbf{c}_k$ to represent the whole context, which is inadequate for abundant contextual information. Liu *et al.* [36] proposed the Contextual Operating Tensor (COT) model to improve CARS^2, as illustrated in Fig. 1(b), motivated by the semantic compositionality and vector representation *word2vec* in Natural Language Processing (NLP) [37]. COT extends the latent space of context representation from a vector $\mathbf{c}_k$ to a matrix $\mathbf{C}_k$. In addition, COT interprets the tensor used in CARS^2 as "contextual operating tensor", drawing analogy between the entity-context relationship with the noun-adjective relationship in NLP [38, 39, 40]. Entity is similar to a noun providing semantic information, while a context is similar to an adjective giving entities the semantic operation. Thus, COT models a context semantic operation on



(a) The CARS^2 model.



(b) The COT model.



(c) The proposed JICO model.

Figure 1: Illustrations: (a) The CARS^2 model, where the cubes denote 3-D contextual operating tensors and the elongated rectangles denote latent factor vectors; (b) The COT model, where the square $\mathbf{C}_k$ is a matrix representing contextual information; (c) The proposed JICO model with two layers, the joint interaction layer and context operation layer, which integrates the virtues of the joint interaction model in MF and the contextual operating tensor in COT.

the original latent vectors of users and items.

Although CARS^2 and COT separate contexts from the latent space of entities to enhance interpretability and performance, the contexts operate on two entities, user and item separately via two contextual operating tensors $\mathcal{W}$ and $\mathcal{Z}$. Such separate interactions are then added (in CARS^2) or multiplied (in

COT). Because the latent vectors of user $\mathbf{u}_i$ and item $\mathbf{v}_j$ lie in a common latent space, the addition and multiplication in CARS^2 and COT "double-count" the impact of contextual information $\mathbf{c}_k$ and $\mathbf{C}_k$. This motivates us to consider the impact of contexts on users and items jointly as in the MF models, rather than separately.

We propose a new context-aware model named as Joint Interaction with Context Operation (JICO) to integrate the virtues of the joint interaction model in MF and the contextual operating tensor in COT. We model the joint interaction between users and items as the semantic subject, i.e., the contextual information operates on the joint interaction between users and items, rather than separately. As shown in Fig. 1(c), JICO has two layers. In the first layer, an interaction operating tensor $\mathcal{I}$ captures the joint interaction between users $\{\mathbf{u}_i\}$ and items $\{\mathbf{v}_j\}$ as in MF, which is then projected to a latent space to define the joint interaction vector $\mathbf{r}_{ij}$. The second layer performs the context semantic operation as in COT, where the contextual information is represented as a matrix $\mathbf{C}_k$.

The remainder of this paper is structured as follows. The next section introduces notations and related works briefly. In Section 3, we propose the JICO model and derive an algorithm based on stochastic gradient descent (SGD). In Section 4, we evaluate JICO on four popular datasets to demonstrate its superior recommendation performance over existing methods. We also study varying contextual influence, more realistic time split recommendation, result interpretation and analysis, as well as parameter sensitivity. Finally, Section 5 draws conclusions.

2. Preliminaries and Related Works

2.1. Notations

In this paper, we follow the notations in [28, 41, 42]. We denote vectors by lowercase boldface letters such as $\mathbf{a}$, matrices by uppercase boldface letters such as $\mathbf{A}$, and tensors by calligraphic letters such as $\mathcal{A}$. An Tth-order tensor $\mathcal{A} \in$

$\mathbb{R}^{I_1 \times \cdots \times I_T}$ is addressed by T indices $\{i_t\}$. Each i_t addresses the t-mode of $\mathcal{A}$. The mode-t product of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_T}$ by a matrix $\mathbf{U} \in \mathbb{R}^{J_t \times I_t}$ is a tensor $\mathcal{B} \in \mathbb{R}^{I_1 \times \ldots \times I_{t-1} \times J_t \times I_{t+1} \times \ldots \times I_T}$, denoted as

$$\mathcal{B} = \mathcal{A} \times_t \mathbf{U}, \tag{1}$$

where each entry of $\mathcal{B}$ is defined as the sum of products of corresponding entries in $\mathcal{A}$ and $\mathbf{U}$:

$$\mathcal{B}(i_1, ..., i_{t-1}, j_t, i_{t+1}, ..., i_T) = \sum_{i_t} \mathcal{A}(i_1, ..., i_T) \cdot \mathbf{U}(j_t, i_t). \tag{2}$$

The *Kronecker product* (also called the *tensor product*) of two matrices $\mathbf{A} \in \mathbb{R}^{r \times s}$ and $\mathbf{B} \in \mathbb{R}^{c \times d}$ results in a block matrix of size $rc \times sd$, which can also be viewed as a fourth-order tensor of size $r \times s \times c \times d$:

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \dots & a_{1s}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \dots & a_{2s}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{r1}\mathbf{B} & a_{r2}\mathbf{B} & \dots & a_{rs}\mathbf{B} \end{bmatrix}. \tag{3}$$

2.2. Matrix Factorization (MF)

In an MF model [8], the joint interaction between users and items is represented with the inner product of their latent vectors. Let $\mathbf{u}_i$, $\mathbf{v}_j \in \mathbb{R}^d$ represent the latent vectors of user i and item j. The estimated rating $\hat{y}_{ij}$ is:

$$\hat{y}_{ij} = \mathbf{u}_i^\top \mathbf{v}_j. \tag{4}$$

2.3. CARS²

CARS^2 model [35] consists of three parts: the user-item, user-context and item-context interactions. For the interaction between users and items, it follows the formula in MF, which is $\mathbf{u}_i^\top \mathbf{v}_j$.

For the user-context and item-context interactions, two 3rd-order tensors $\mathcal{W} \in \mathbb{R}^{d \times d_p \times d_c}$ and $\mathcal{Z} \in \mathbb{R}^{d \times d_p \times d_c}$ are given. In CARS^2, the specific context k

is denoted by a distinct vector $\mathbf{c}_k \in \mathbb{R}^{d_c}$. The impact of context k on the user i and item j is formulated as:

$$\mathbf{p}_{ik} = \mathcal{W} \times_1 \mathbf{u}_i \times_3 \mathbf{c}_k, \tag{5}$$

$$\mathbf{q}_{jk} = \mathcal{Z} \times_1 \mathbf{v}_j \times_3 \mathbf{c}_k, \tag{6}$$

where $\mathbf{p}_{ik}$ and $\mathbf{q}_{jk}$ are d_p-dimensional vectors that represent the interaction between user/item and contexts. Then CARS^2 introduces $\mathbf{a}_j,\ \mathbf{b}_i \in \mathbb{R}^{d_p}$ as weight vectors for $\mathbf{p}_{ik}$, $\mathbf{q}_{jk}$. The estimated rating $\hat{y}_{ijk}$ is modeled as:

$$\hat{y}_{ijk} = \mathbf{u}_i^\top \mathbf{v}_j + \mathbf{a}_j^\top \mathbf{p}_{ik} + \mathbf{b}_i^\top \mathbf{q}_{jk}. \tag{7}$$

2.4. Contextual Operating Tensor (COT)

COT [36] extends the latent space of context representation from a distinct vector $\mathbf{c}_k$ in CARS^2 to a matrix $\mathbf{C}_k \in \mathbb{R}^{d_c \times n}$, where n is the number of contexts. The latent vectors of users and items under contexts are formulated as:

$$\mathbf{p}_{ik} = \mathcal{W} \times_1 \mathbf{u}_i \times_3 (\mathbf{h}_i \mathbf{C}_k), \tag{8}$$

$$\mathbf{q}_{jk} = \mathcal{Z} \times_1 \mathbf{v}_j \times_3 (\mathbf{h}_j \mathbf{C}_k), \tag{9}$$

where $\mathbf{h}_i$ and $\mathbf{h}_j$ are weight vectors for context $\mathbf{C}_k$ on user i and item j respectively. The estimated rating $\hat{y}_{ijk}$ is defined as:

$$\hat{y}_{ijk} = b_0 + b_i + b_j + \sum_{m=1}^{n} b_{m,k} + \mathbf{p}_{i,k}^\top \mathbf{q}_{j,k}, \tag{10}$$

in which b_0, b_i and b_j represent the global, user and item bias respectively, and $b_{m,k}$ is the bias of the mth context variable. $\mathbf{p}_{i,k}^\top \mathbf{q}_{j,k}$ is the interaction between user i and item j under context k, which is similar to MF.

Finally, the objective function of all three models can be expressed as below:

$$min\ L(\Theta) = \sum_{(i,j,k \in S)} (\hat{y}_{ijk} - y_{ijk})^2 + \Omega(\Theta), \tag{11}$$

where $\Omega(\Theta)$ is the regularization terms for model parameters.

2.5. Double-Counting of Contextual Information

CARS^2 and COT lead to "double-counting" of contextual information since predicted rating is calculated via their product in COT and their sum in CARS^2. These two methods represent the interaction between user/item and contexts by latent vector $\mathbf{p}_{ik}$ or $\mathbf{q}_{jk}$, and the contextual information is "double-counted" via such product/sum. Our method will address this issue.

3. Proposed JICO Model

3.1. Joint Interaction between Users and Items

We propose JICO to model joint interaction as in MF and context operation as in CARS^2/COT so that the contextual information can be more properly accounted for. The JICO model has two layers. We name the first layer as the "joint interaction layer". In this layer, we capture the joint interaction between user i and item j. In a typical MF model, the interaction is represented by the inner product of the latent vectors $\mathbf{u}_i \in \mathbb{R}^d$ and $\mathbf{v}_j \in \mathbb{R}^d$, which is an effective way to estimate ratings. However, this way of joint interaction modelling does not allow introducing the influence of contextual information. To overcome this problem, we propose to form an *interaction operating tensor* $\mathcal{I} \in \mathbb{R}^{d \times d_p \times d}$ to capture the joint interaction and enable further context operation. We then define the *joint interaction vector* $\mathbf{r}_{ij}$ as:

$$\mathbf{r}_{ij} = \mathcal{I} \times_1 \mathbf{u}_i \times_3 \mathbf{v}_j. \tag{12}$$

In the joint interaction layer, the interaction operating tensor $\mathcal{I}$ describes the common latent interaction space between users and items. With the introduction of $\mathcal{I}$, the interaction can now be represented as a latent *vector*, rather than a *scalar* value as in the MF model. It is intuitive that the interaction between two entities (user and item) can be evaluated from many different aspects rather than a single value. The latent interaction vector $\mathbf{r}_{ij}$ can reflect these latent aspects. Moreover, a vector $\mathbf{r}_{ij}$ enables further combination with the influence of context, which a single scalar in MF cannot achieve.

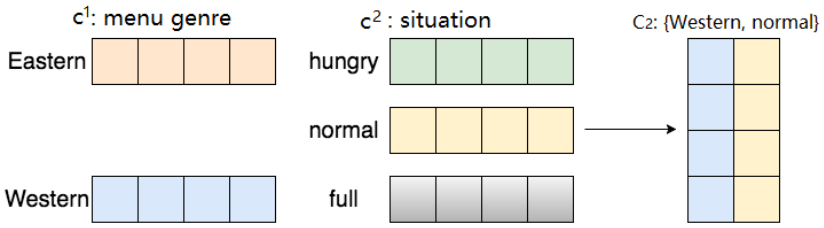


Figure 2: An example of combining two types of latent contextual information in a food rating dataset.

3.2. Contextual Information Representation

We denote contextual information set by $C = \{C^1, C^2, ..., C^n\}$, where n is the number of contextual variables and each element C^i represents a specific context such as age, location, time, etc. Each contextual variable C^i can take different values $\{c_1^i, c_2^i, ...\}$. Similar to the representation in COT, each specific value c_m^i is represented by a latent vector $\mathbf{c}_m^i \in \mathbb{R}^{d_c}$ in JICO. The vector representation is an efficient method to describe the latent properties of contexts. As different contextual values are not always independent, they can have similar or opposite effects on the user preferences. For example, doctors and nurses may have a similar preference on certain books, but the preference of programmers may be quite different. If we just use a real value to represent the occupation, it is difficult to describe such similarity properties properly. Therefore, the latent vector provides a rich representation that can better reflect the relevance of different contextual information than a single scalar value. In this way, the whole contextual information of each rating could be individually denoted by a context combination matrix $\mathbf{C}_k = [\mathbf{c}_{m_1}^1, \mathbf{c}_{m_2}^2, ..., \mathbf{c}_{m_n}^n] \in \mathbb{R}^{d_c \times n}$, where each column vector $\mathbf{c}_{m_j}^j$ denotes a latent vector for a specific value m_j taken on contextual variable C^j. Figure 2 gives a simple example of $\mathbf{C}_k$.

With the above observation, we form the *context combination vector* $\mathbf{c}_k \in \mathbb{R}^{d_c}$ as:

$$\mathbf{c}_k = \mathbf{C}_k \mathbf{w}_k, \tag{13}$$

where $\mathbf{C}_k$ is the *context combination matrix* and $\mathbf{w}_k \in \mathbb{R}^n$ is a vector denoting

the weights of each context.

3.3. Context Operation Layer

We name the second layer of JICO as "context operation layer". In this layer, we introduce the *contextual operating tensor* $\mathcal{Z} \in \mathbb{R}^{d_p \times d \times d_c}$ as in COT to take the influence of contexts into account. In the first layer, we have obtained the joint interaction vector $\mathbf{r}_{ij}$ of users and items. Now the tensor $\mathcal{Z}$ denotes the context operation by $\mathbf{C}_k$ on $\mathbf{r}_{ij}$. Therefore, the joint interaction after the context operation can be defined as:

$$\mathbf{r}_{ijk} = \mathcal{Z} \times_1 \mathbf{r}_{ij} \times_3 \mathbf{c}_k, \tag{14}$$

where $\mathbf{r}_{ijk} \in \mathbb{R}^d$ represents the joint interaction vector under contextual information $\mathbf{c}_k$. The estimated rating $\hat{y}_{ijk}$ is then formulated as:

$$\hat{y}_{ijk} = \mathbf{w}^\top \mathbf{r}_{ijk}, \tag{15}$$

where $\mathbf{w} \in \mathbb{R}^d$ is a weight vector for $\mathbf{r}_{ijk}$. The JICO model has been illustrated earlier in Fig. 1(c).

Next, we do not expect the interaction $\mathbf{r}_{ijk}$ can fully explain the ratings due to the existence of bias. We assume that each user or item has its own bias. For example, if a movie has a good reputation, the average rating of this movie tends to be higher and users could have different standards for rating scales. Some may be strict in evaluating a movie and tend to give low ratings but some may give high average ratings. This means that each movie or user has its own bias. To take this into accounts, we add the bias to get the following:

$$\hat{y}_{ijk} = b_k + b_i + b_j + \mathbf{w}^\top \mathbf{r}_{ijk}, \tag{16}$$

where b_k is a constant indicating the overall average rating under contexts combination k, b_i and b_j are the bias of user i and item j respectively. In this way, the interaction part $\mathbf{w}^\top \mathbf{r}_{ijk}$ can capture information from the dataset more accurately.

Algorithm 1 Joint Interaction with Context Operation

Input: Training dataset S, where each real rating y_{ijk} is associated with user i, item j, and context combination k. Parameters η, λ_1 and λ_2.

Output: Updated parameters $\{b_i\}$, $\{b_j\}$, $\mathbf{U}$, $\mathbf{V}$, $\mathbf{w}$, $\{\mathbf{C}_k\}$, $\{\mathbf{w}_k\}$, $\mathcal{I}$, $\mathcal{Z}$

Initialize $\{b_i\}$, $\{b_j\}$, $\{\mathbf{u}_i\}$, $\{\mathbf{v}_j\}$, $\mathbf{w}$, $\{\mathbf{C}_k\}$, $\{\mathbf{w}_k\}$, $\mathcal{I}$, $\mathcal{Z}$ with random values
Set $t = 0$
while not convergent **do**
    $\eta \leftarrow \frac{1}{\sqrt{t}}$ and $t = t + 1$
    Randomly select an instance y_{ijk} from S
    $\hat{y}_{ijk} = b_k + b_i + b_j + \mathbf{w}^\top \mathbf{r}_{ijk}$
    Update $b_i \leftarrow b_i - \eta \frac{\partial L}{\partial b_i}$
    Update $b_j \leftarrow b_j - \eta \frac{\partial L}{\partial b_j}$
    Update $\mathbf{u}_i \leftarrow \mathbf{u}_i - \eta \frac{\partial L}{\partial \mathbf{u}_i}$
    Update $\mathbf{v}_j \leftarrow \mathbf{v}_j - \eta \frac{\partial L}{\partial \mathbf{v}_j}$
    Update $\mathbf{w} \leftarrow \mathbf{w} - \eta \frac{\partial L}{\partial \mathbf{w}}$
    Update $\mathbf{C}_k \leftarrow \mathbf{C}_k - \eta \frac{\partial L}{\partial \mathbf{C}_k}$
    Update $\mathbf{w}_k \leftarrow \mathbf{w}_k - \eta \frac{\partial L}{\partial \mathbf{w}_k}$
    Update $\mathcal{I} \leftarrow \mathcal{I} - \eta \frac{\partial L}{\partial \mathcal{I}}$
    Update $\mathcal{Z} \leftarrow \mathcal{Z} - \eta \frac{\partial L}{\partial \mathcal{Z}}$
end while

3.4. Objective Function and Parameters Inference

After introducing the JICO model, we formulate the JICO objective function to minimize using the squared error loss function:

$$\begin{aligned} \min_{\Theta} L = & \sum_{(i,j,k \in S)} (\hat{y}_{ijk} - y_{ijk})^2 \\ & + \frac{\lambda_1}{2}(\|\mathbf{u}_i\|^2 + \|\mathbf{v}_j\|^2 + \|\mathbf{C}_k\|^2 + \|\mathbf{w}_k\|^2 + \|\mathcal{I}\|^2 + \|\mathcal{Z}\|^2 + \|\mathbf{w}\|^2) \\ & + \frac{\lambda_2}{2}(\|b_i\|^2 + \|b_j\|^2), \end{aligned} \tag{17}$$

where $\Theta = \{\mathbf{u}_i, \mathbf{v}_j, \mathbf{C}_k, \mathbf{w}_k, \mathcal{I}, \mathcal{Z}, \mathbf{w}\}$ represents the set of parameters in the objective function L, S is the training dataset, λ_1 and λ_2 denote the regularization

parameters. We denote the Frobenius norm of a tensor $\mathcal{Z} \in \mathbb{R}^{d_p \times d \times d_c}$ as $\|\mathcal{Z}\|$, which is defined as follows:

$$||\mathcal{Z}|| = \sqrt{\sum_{i=1}^{d_p} \sum_{j=1}^{d} \sum_{k=1}^{d_c} z_{ijk}^2}, \tag{18}$$

where z_{ijk} is the element at (i, j, k) of $\mathcal{Z}$.

The regularization terms control the magnitude of parameters to avoid overfitting. In this paper, we use a common optimization scheme, stochastic gradient descent (SGD), to update parameters. The gradients with respect to parameters in $L(\Theta)$ can be calculated as:

$$\frac{\partial L}{\partial b_i} = 2l_{ijk} + \lambda_2 b_i, \tag{19}$$

$$\frac{\partial L}{\partial b_j} = 2l_{ijk} + \lambda_2 b_j, \tag{20}$$

$$\frac{\partial L}{\partial \mathbf{C}_k} = 2l_{ijk}(\mathcal{Z} \times_1 (\mathcal{I} \times_1 \mathbf{u}_i^\top \times_3 \mathbf{v}_j^\top)^\top \times_2 \mathbf{w}^\top)^\top \mathbf{w}_k + \lambda_1 \mathbf{C}_k, \tag{21}$$

$$\frac{\partial L}{\partial \mathbf{w}_k} = 2l_{ijk}(\mathcal{Z} \times_1 (\mathcal{I} \times_1 \mathbf{u}_i^\top \times_3 \mathbf{v}_j^\top)^\top \times_2 \mathbf{w}^\top)\mathbf{C}_k + \lambda_1 \mathbf{w}_k, \tag{22}$$

$$\frac{\partial L}{\partial \mathbf{w}} = 2l_{ijk}\mathcal{Z} \times_1 (\mathcal{I} \times_1 \mathbf{u}_i^\top \times_3 \mathbf{v}_j^\top)^\top \times_3 (\mathbf{C}_k \mathbf{w}_k)^\top + \lambda_1 \mathbf{w}, \tag{23}$$

$$\frac{\partial L}{\partial \mathbf{u}_i} = 2l_{ijk}\mathcal{I} \times_3 \mathbf{v}_j^\top \times_2 (\mathcal{Z} \times_2 \mathbf{w}^\top \times_3 (\mathbf{C}_k \mathbf{w}_k)^\top) + \lambda_1 \mathbf{u}_i, \tag{24}$$

$$\frac{\partial L}{\partial \mathbf{v}_j} = 2l_{ijk}\mathcal{I} \times_1 \mathbf{u}_i^\top \times_2 (\mathcal{Z} \times_2 \mathbf{w}^\top \times_3 (\mathbf{C}_k \mathbf{w}_k)^\top) + \lambda_1 \mathbf{v}_j, \tag{25}$$

$$\frac{\partial L}{\partial \mathcal{I}} = 2l_{ijk}\mathbf{u}_i^\top \otimes (\mathcal{Z} \times_2 \mathbf{w}^\top \times_3 (\mathbf{C}_k \mathbf{w}_k)^\top) \otimes \mathbf{v}_j + \lambda_1 \mathcal{I}, \tag{26}$$

$$\frac{\partial L}{\partial \mathcal{Z}} = 2l_{ijk}(\mathcal{I} \times_1 \mathbf{u}_i^\top \times_3 \mathbf{v}_j^\top) \otimes \mathbf{w}^\top \otimes (\mathbf{C}_k \mathbf{w}_k)^\top + \lambda_1 \mathcal{Z}, \tag{27}$$

where $l_{ijk} = \hat{y}_{ijk} - y_{ijk}$.

Algorithm 1 summarizes the pseudocode for JICO. The matrices $\mathbf{U} \in \mathbb{R}^{I \times d}$ and $\mathbf{V} \in \mathbb{R}^{J \times d}$ are the user and item latent matrices (I/J is the number of users/items). The ith row of $\mathbf{U}$ denotes the user latent vector $\mathbf{u}_i$ and the jth row of $\mathbf{V}$ denotes the item latent vector $\mathbf{v}_j$. η is the learning rate. It decreases with the number of iterations increases. In each iteration, the algorithm will evaluate and update only one relevant $\mathbf{u}_i$ and $\mathbf{v}_j$. To prevent gradient explosion, the elements of initial matrices and tensors are drawn from a zero-mean uniform distribution in the range [-$\frac{1}{2}$, $\frac{1}{2}$]. When the iteration converges, we obtain the final parameters for estimating the rating in test data.

3.5. Computational Complexity and Model Complexity

We follow CARS^2 [35] and COT [36] to analyse the computational complexity of JICO. To simplify the analysis, we set $d_p = d_c = d$. The computational complexity of JICO is dominated by the computation of $\mathbf{U}$, $\mathbf{V}$, $\mathcal{I}$ and $\mathcal{Z}$, which have complexity $O(d^3 \times |S|)$. Therefore, the total computational complexity is $O(d^3 \times |S|)$, which is similar to that for CARS^2 [35] and COT [36].

The model complexity of JICO (Γ_J) and COT (Γ_C) are presented as follows:

$$\Gamma_J = \underbrace{d \times d_p \times d}_{\mathcal{I}} + \underbrace{d_p \times d \times d_c}_{\mathcal{Z}} + \underbrace{d_c \times n^2}_{\{\mathbf{C}_k\}} + \underbrace{d \times I}_{\{\mathbf{u}_i\}} + \underbrace{d \times J}_{\{\mathbf{v}_j\}} + \underset{\mathbf{w}}{\underbrace{d}} + \underset{\{\mathbf{w}_k\}}{\underbrace{n}}^2 + \underset{\{b_i\}}{\underbrace{I}} + \underset{\{b_j\}}{\underbrace{J}}, \tag{28}$$

$$\Gamma_C = \underbrace{d \times d_p \times d_c}_{\mathcal{W}} + \underbrace{d \times d_p \times d_c}_{\mathcal{Z}} + \underbrace{d_c \times n}_{\{\mathbf{C}_k\}} + \underbrace{d \times I}_{\{\mathbf{u}_i\}} + \underbrace{d \times J}_{\{\mathbf{v}_j\}} + \underbrace{d_c \times I}_{\{\mathbf{h}_i\}} + \underbrace{d_c \times J}_{\{\mathbf{h}_j\}} + \underset{\{b_{m,k}\}}{\underbrace{n}}^2 + \underset{\{b_i\}}{\underbrace{I}} + \underset{\{b_j\}}{\underbrace{J}}. \tag{29}$$

The difference between Γ_C and Γ_J is:

$$\Gamma_C - \Gamma_J = d_c \times I + d_c \times J + d \times d_p \times d_c - d - d \times d_p \times d. \tag{30}$$

In practice I and J are much larger than n, d, d_p and d_c. Thus, Γ_C will be greater than Γ_J, and model complexity of JICO is smaller than that of COT.

The reason is that the double-counting in COT requires more parameters than JICO. Therefore, JICO outperforms COT with a new architecture of lower capacity, i.e., more compact size by removing "double-counting".

3.6. JICO with Double-Counting (JICO^{DC})

To explore the effect of "double-counting" on the performance of JICO, we consider JICO with "double-counting" (JICO$^{\mathrm{DC}}$) via directly interacting latent vector of user $\mathbf{u}_i$ and item $\mathbf{v}_j$ with contextual operating tensor $\mathcal{Z}$. This leads to:

$$\hat{\mathbf{p}}_{ik} = \mathcal{Z} \times_1 \mathbf{u}_i \times_3 \mathbf{c}_k, \tag{31}$$

$$\hat{\mathbf{q}}_{jk} = \mathcal{Z} \times_1 \mathbf{v}_j \times_3 \mathbf{c}_k. \tag{32}$$

The estimated rating $\hat{y}_{ijk}$ is:

$$\hat{y}_{ijk} = b_k + b_i + b_j + \hat{\mathbf{p}}_{ik}^{\top} \hat{\mathbf{q}}_{jk}. \tag{33}$$

After introducing the JICO$^{\mathrm{DC}}$ model, we formulate the objective function to minimize using the squared error loss function as follows:

$$\begin{aligned} \min_{\Theta} L = & \sum_{(i,j,k \in S)} (\hat{y}_{ijk} - y_{ijk})^2 \\ & + \frac{\lambda_1}{2}(\|\mathbf{u}_i\|^2 + \|\mathbf{v}_j\|^2 + \|\mathbf{C}_k\|^2 + \|\mathbf{w}_k\|^2 + \|\mathcal{Z}\|^2 + \|\mathbf{w}\|^2) \\ & + \frac{\lambda_2}{2}(\|b_i\|^2 + \|b_j\|^2). \end{aligned} \tag{34}$$

JICO$^{\mathrm{DC}}$ is a variant of JICO by considering "double-counting" procedure. We will experimentally explore the effect of "double-counting" on JICO.

4. Experiments

4.1. Experimental Design

Compared methods. In this section, we perform experiments on context-aware recommender systems to evaluate JICO against the following methods:

- SVD++ [43] is an improved matrix factorization model, via Singular Value Decomposition (SVD), for recommender systems with L2 regularization.

  It is not context aware.

- Multiverse [25] is developed based on Tucker decomposition [29] on the user $\times$ item $\times$ context rating tensor. This method has been shown to outperform pre-filtering and multidimensional approach.

- Factorization Machine (FM) [26] can model contextual information and provide context-awareness rating predictions by specifying input data.

- CARS^2 [35] associates a latent vector and a context-aware representation to each user and item. This representation can efficiently capture latent properties of users and items.

- COT [36] extends the latent space of context representation from a distinct vector to a matrix. In addition, COT performs a context semantic operation on the original latent vectors of users and items to model contexts.

We set the parameters of all methods via 5$\times$5-fold cross-validation (CV). We use grid search CV to find the best parameters for each method. The grid for $d_u/d_v/d_c/d_p/d$ is [1, 2, $\cdots$, 20], and the grid for $\eta/\lambda_1/\lambda_2$ is [0.1, 0.05, 0.01, 0.005, 0.001].

Datasets. We conduct experiments on three real datasets and one semi-synthetic dataset below, with Table 1 summarizing the basic information.

- **Food dataset**: This is a popular small-scale dataset obtained from the coauthor (H. Asoh) of [44]. It studies how much one subject wants to order a dish. It contains 6,360 ratings given by 212 users for 20 food menus. The data includes ratings in real situations and virtual situations (really hungry or just imagine). Following the advice of the data provider (H. Asoh), we only take ratings in real situations to study. Two popular context variables are selected: situation (hungry, normal, full) and menu

Table 1: Experiment dataset summary.

Dataset	Users	Items	Context Dim.	Ratings	Scale
Food	212	20	2	2120	1-5
Movielens-100K	943	1682	3	100K	1-5
Yahoo Webscope	7620	11916	2	221K	1-5
Movielens-1M	6040	3706	3	1M	1-5

genre (Japanese, Chinese, Western), which are user and item contexts respectively.

- **Movielens-100K**[1]: This is a popular dataset collected by the GroupLens Research Project at the University of Minnesota. It consists of 100K ratings in a {1, 2, 3, 4, 5} scale from around 943 users on 1,682 movies. Each user rated at least 20 movies. We pick three types of user contextual information that are most interesting to us: the gender, occupation, and age. There are 20 occupations such as lawyer, doctor, and artist, and the ages are divided into 5 groups: $\leq$18, 18-25, 25-35, 35-50 and $\geq$50.

- **Movielens-1M**[1]: This is a larger version of Movielens-100K. It contains 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 users. Other information is the same as that of Movielens-100K, such as rating scale and occupation category. The timestamp is separated to hour and day contextual information.

- **Yahoo Webscope Movies**[2]: This dataset is provided as part of the Yahoo! Research Alliance Webscope program. It contains 221K ratings in a {1, 2, 3, 4, 5} scale for about 11,916 movies evaluated by 7,620 users, as well as various user contexts such as gender and age, and item contexts including the timestamp. In order to explore the extent that

[1] http://movielens.org/
[2] http://research.yahoo.com/

contexts affect user ratings, we follow the data preprocessing steps in [25] to generate six semi-synthetic datasets from the original data using two user-context features "age" and "gender" in the raw Yahoo dataset by the following steps.

1. We convert the age feature into a three-class categorical variable according to three groups: above 50, between 18 and 50 and below 18.
2. We generate a random feature $x \in \{0, 1\}$ for each rating to replace the original gender feature to artificially affect the ratings.
3. We randomly pick $a \times 100\%$ movies from the dataset.
4. We randomly choose $b \times 100\%$ ratings from these movies to increase (decrease) the rating value by one if the corresponding $x = 1$ ($x = 0$) unless the rating value is already 5 (1). E.g., if 50% of the dataset's movies are randomly picked and 10% of the selected movies' rating would be modified, the synthetic dataset with $a = 0.9$ and $b = 0.5$ has 90% of its items altered with profiles that have half of their ratings changed. Six semi-synthetic datasets are generated with $a = 0.1, 0.5, 0.9$ and $b = 0.1, 0.5, 0.9$. In this way, a large $a \times b$ has a greater contextual influence on the ratings.

Evaluation metrics. We use two popular metrics in evaluation, the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE):

$$RMSE = \sqrt{\frac{\sum_{y \in \Omega_{test}} (y - \hat{y})^2}{|\Omega_{test}|}} \tag{35}$$

$$MAE = \frac{\sum_{y \in \Omega_{test}} |y - \hat{y}|}{|\Omega_{test}|} \tag{36}$$

where y denotes the true rating, $\hat{y}$ is the estimated rating, and Ω_{test} represents the test set. The smaller the RMSE and MAE, the better the recommendation performance. 5×5-fold cross-validation results are reported.

Table 2: Recommendation performance comparison on four datasets, with seven scenarios in total. In each scenario, the best results (achieved by JICO) are in bold and the second best (achieved by COT) ones are underlined. We report the standard deviation of 5×5 cross validation results for COT and JICO except the time split results. For the time split results (time), we indicate the performance degradation of each method in a pair of parentheses.

Dataset		SVD++	Multiverse	FM	CARS^2	COT	JICO	JICO$^{\mathrm{DC}}$
Food	RMSE	1.114	1.068	1.066	1.062	1.058±0.024	**1.040**±0.019	1.064±0.013
	MAE	0.895	0.844	0.840	0.843	0.838±0.015	**0.827**±0.014	0.841±0.008
Movielens-100K	RMSE	0.991	0.953	0.952	0.948	0.947±0.006	**0.941**±0.005	0.948±0.006
	MAE	0.772	0.753	0.750	0.748	0.745±0.003	**0.739**±0.003	0.745±0.003
Movielens-100K (time)	RMSE	1.256(0.265)	1.113(0.160)	1.105(0.153)	1.082(0.134)	1.068(0.121)	**1.055**(0.114)	1.071(0.123)
	MAE	1.022 (0.250)	0.895(0.142)	0.887(0.137)	0.866(0.118)	0.859(0.114)	**0.834**(0.095)	0.860(0.115)
Yahoo (a=0.9, b=0.5)	RMSE	1.049	1.009	1.012	1.008	0.994±0.007	**0.988**±0.005	0.997±0.005
	MAE	0.778	0.726	0.735	0.724	0.720±0.005	**0.714**±0.003	0.722±0.004
Yahoo (a=0.9, b=0.9)	RMSE	1.029	0.928	0.939	0.911	0.899±0.005	**0.896**±0.007	0.912±0.008
	MAE	0.786	0.653	0.668	0.651	0.643±0.004	**0.637**±0.005	0.651±0.006
Movielens-1M	RMSE	0.885	0.876	0.866	0.868	0.864±0.004	**0.859**±0.003	0.867±0.003
	MAE	0.691	0.668	0.664	0.665	0.661±0.004	**0.657**±0.003	0.662±0.004
Movielens-1M (time)	RMSE	1.092(0.207)	1.024(0.048)	1.012(0.146)	0.998(0.130)	0.992(0.128)	**0.983**(0.124)	0.994(0.127)
	MAE	0.866(0.175)	0.778(0.110)	0.773(0.109)	0.767(0.102)	0.764(0.103)	**0.756**(0.099)	0.764(0.102)

4.2. Recommendation Performance Comparison

Table 2 compares the performance of each model on the four datasets. The best and second-best results in each case, achieved by JICO and COT, are highlighted in bold and underline, respectively. Table 3 reports the p values of t-tests performed for the results of JICO and COT, and JICO and JICO$^{\mathrm{DC}}$, to confirm the statistical significance. We have the following observations:

- The context-aware models all significantly outperform the context-free model SVD++, suggesting that rich contextual information is beneficial to improve recommendation performance. JICO improves over SVD++ by 6.6%, 5.0%, 5.8%, 12.9%, 2.9% in RMSE and 7.6%, 4.2%, 8.2%, 19.0%, 4.9% in MAE on the Food, Movielens-100K, Yahoo (0.9 × 0.5), Yahoo (0.9 × 0.9) and Movielens-1M datasets respectively. There is a clear gap between the improvements of different datasets, since the effect of contextual information is not always the same in different situations. The more

Table 3: p-value for the results of JICO and COT, and JICO and JICO$^{\mathrm{DC}}$ reported in Table 2.

	JICO vs COT		JICO vs JICO$^{\mathrm{DC}}$	
	RMSE	MAE	RMSE	MAE
Food	0.005	0.006	8.611e-06	6.936e-05
Moivelens-100K	0.001	5.892e-07	2.517e-4	2.268e-07
Yahoo (0.9×0.5)	0.001	2.002e-04	1.083e-5	2.232e-08
Yahoo (0.9×0.9)	0.158	1.208e-04	2.015e-05	4.612e-10
Movielens-1M	8.126e-05	4.185e-05	3.746e-10	1.571e-06

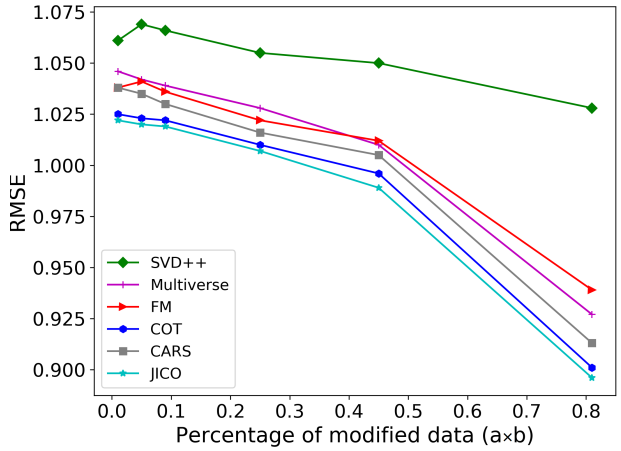
relevant the contexts and entities, the greater the improvement in general.

- Among context-aware methods, Multiverse and FM have similar performance on all datasets, while inferior to CARS^2, COT, and JICO. E.g., JICO improves over the best of Multiverse and FM by 2.4%, 1.1%, 2.0%, 3.4%, 0.81% in RMSE and by 1.5%, 1.5%, 1.7%, 2.5%, 1.1% in MAE on Food, Movielens-100K, Yahoo (0.9×0.5) and Yahoo (0.9×0.9) and Movielens-1M datasets. The limited performance of Multiverse and FM is likely limited due to treating contexts as a dimension similar to users and items. However, contexts play different roles as users and items in a recommendation. Therefore, modelling contexts as semantic operators, as in CARS^2, COT, and JICO, can further improve the performance of context-aware recommendation.

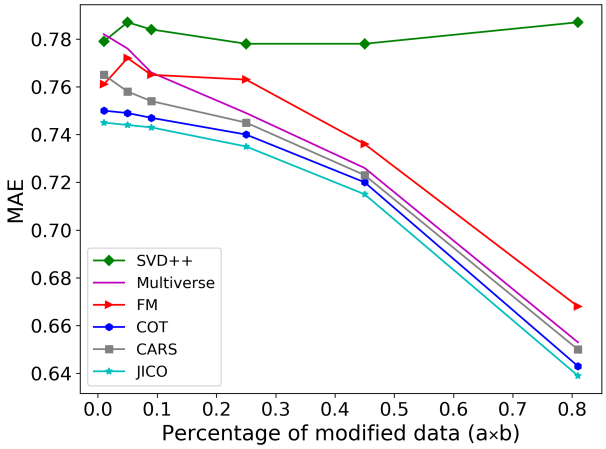
- JICO achieves the best performance consistently in all cases studied. Although CARS^2, COT, and JICO all model contexts as semantic operators, the interaction between contexts and user/item is separately modelled in CARS^2 and COT, leading to double-counting.

  Hence, JICO combines joint interaction in MF and context operation in CARS^2 and COT to achieve superior performance over all other methods.

- JICO significantly outperforms JICO$^{\mathrm{DC}}$ in all datasets, which suggests



(a) RMSE Evolution



(b) MAE Evolution

Figure 3: Recommendation performance variation with respect to the amount of contextual influence on Yahoo semi-synthetic dataset.

that eliminating double-counting is beneficial to improve the performance.

4.3. Varying Contextual Influence

The six semi-synthetic Yahoo datasets enable us to study the various degrees of contextual influence, as controlled by $a \times b$, as in [25]. Figure 3 depicts the recommendation performances with respect to the growth of contextual influence. We can see the following:

- Over all degrees of contextual influence, JICO consistently gives the best performance, with COT the second, and CARS^2 the third. JICO improves over FM, Multiverse, CARS^2, and COT by 4.3%, 2.3%, 1.7%, 1.3% and 0.9% on high context dataset with $a \times b = 0.81$. It means that JICO can make better use of context in a high context influence situation. This again demonstrates the benefits of combining joint interaction in MF and context operation in COT and CARS^2.

- The performance gain of context-aware methods over the context-free method, SVD++, continues increasing with the greater degree of contextual influence. This indicates that greater contextual influence can lead to greater performance improvement. All the context-aware methods are benefiting from such influence growth.

- Relative to context-aware methods, the influence of contexts on SVD++ is quite limited. It is interesting to note that on the left side of the MAE subfigure, SVD++ slightly outperformed Multiverse, indicating that when the contextual information is weak, Multiverse may not model it properly, leading to a negative impact.

4.4. Realistic Evaluation on Time Split

Although we report the cross-validation results popularly done in performance evaluation, it is actually not a realistic assessment in practice. This is because rating data come in sequence and we can only use past data to predict future data, not vice versa. Considering this situation, we present experiments about time split that is a realistic case embodying two challenges: one is *cold*

start, and the other is *imbalanced* training/testing data. We carry out this realistic study by splitting the training/test data at a time point, rather than random splitting in cross-validation.

For this time split study, we use Movielens-100K and Movielens-1M and split each dataset into training (80%) and test (20%) data by its attribute "timestamp". The results are reported in the rows labelled as "Movielens-100K (time)"and "Movielens-1M (time)" in Table 2. Compared to the "Movielens-100K"and "Movielens-1M" rows, we can see that all results get worse (larger errors) with time split. In Table 2, we indicate the increased amount of error in a pair of parentheses in the "Movielens-100K (time)" and "Movielens-1M (time)" rows. SVD++ suffers the most degradation (e.g., 0.265/0.250 in RMSE/MAE on Movielens-100K, and 0.207/0.175 in RMSE/MAE on Movielens-1M). JICO has the least degradation (only 0.114/0.095 in RMSE/MAE on Movielens-100K, and 0.124/0.099 in RMSE/MAE on Movielens-1M).

Due to the time constraint, although the amount of training data are the same for time split and cross-validation (random split), there are actually less entity (user/item) information available for the time split case. For example, new users and movies after the time split point have no information available for training in the time split case while the cross-validation case could have such information due to random sampling. In other words, the effective contextual information is less in the time split case than in the cross-validation case. The least degradation achieved by JICO again shows the superiority of our two-layer approach.

4.5. Context Analysis and Interpretation

As a context-aware model, JICO enables interpretation of context importance, which is captured by the context weight vector $\mathbf{w}_k$. Figure 4(a) depicts different context weight percentage on Movielens-100K and Food datasets. The greater the percentage, the higher the importance. This figure shows that context variable "Situation" has more influence on rating than context "Menu genre" on the Food dataset. On Movielens-100K, the most important context

is occupation (50%), followed by age (41%), and genders impact is relatively small (<10%). This allows an intuitive understanding of the effects of different contexts on ratings.

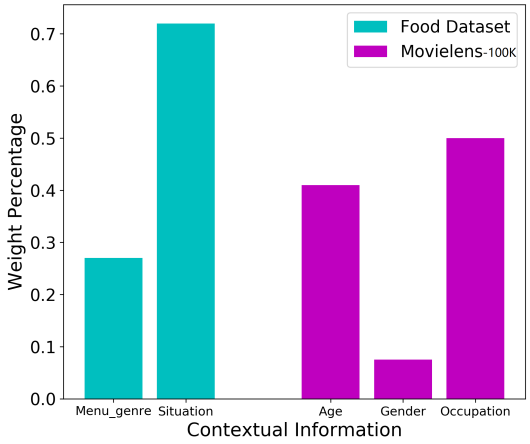
As a latent factor model, JICO also learns latent representations of contexts to reveal the relationships among them. Since the context latent vector's dimension can vary, we follow Wu *et al.* [45] to use Principal Component Analysis (PCA) to reduce the context vector's dimension to 2 to visualize the relationships among different context values in a 2-D space. In the context importance analysis above, we found that "Occupation" is a dominate context on Movielens-100K dataset so we visualize this context in 2-D space via PCA in Fig. 4(b). The figure mainly reflects two facts. We can observe:

- The location of each occupation reflects the occupation's rating bias. The top right tends to give higher ratings while the lower left tends to give lower ratings. For instance, the average ratings for the engineer, programmer, educator, and artist are 3.541, 3.568, 3.670, and 3.441, respectively. We can see that the "artist" tends to give lower ratings, due to their unique artistic taste. In contrast, the "educator" occupation tends to give a higher rating to movies than "artist", which is expected.

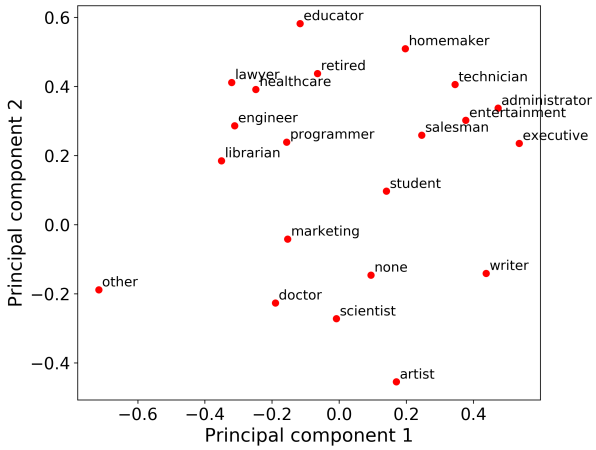
- The distance between different points reflects their preference similarities and potential relationships. E.g., programmer and engineer are close, indicating they have a similar preference on movies, while doctor and educator are far apart, showing that they tend to have different tastes. The context latent vector representations in JICO, with PCA, can help reveal relationships among context values. Such analysis can help us gain important insights and use such information in subsequent applications.

4.6. Impact of Latent Vector Dimensions

Finally, we show the sensitivity of JICO (in RMSE) to the latent vector dimensions of user/item and contexts on the Food dataset in Figs. 5(a) and 5(b), respectively. The user/item or context dimension is fixed to 4 when testing



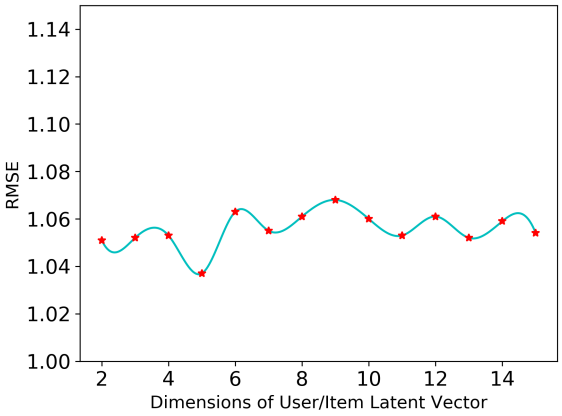
(a)



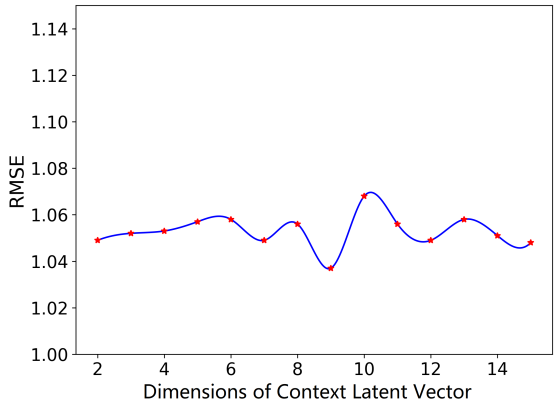
(b)

Figure 4: Contextual information analysis: (a) Different context weight percentages on the Movielens-100K and Food datasets. (b) Occupation representations in 2-D space, with dimension reduced by PCA.

the other. We can see that the best dimensionality of user/item vector d_e is 5 while that for context d_c is 9. Both curves change smoothly with only small



(a) Entity vector dimension.



(b) Context vector dimensions.

Figure 5: Recommendation performance sensitivity with respect to JICO latent vector dimensions on the Food dataset.

variations, which are much smaller than those observed in similar curves in COT [27]. This indicates that JICO is much less sensitive to the latent vector dimension settings than COT.

5. Conclusion

This paper proposed JICO, a novel context-aware recommendation model combining joint interaction with the context operation. It models the relationships between user/item and contexts more properly by addressing limitations of previous models while leveraging their virtues. We performed experiments on four datasets to show the superior recommendation performance and various interesting studies, such as varying contextual influence and time split recommendation. The results show that JICO consistently outperforms other state-of-the-art models in all cases studied.

In future, we can further develop JICO in two directions. Firstly, since JICO can be treated as a generic supervised learning model, we can potentially add deep models with multiple layers to further improve the model performance [46, 47]. Secondly, Lawrence and Urtasun [48] proposed a non-linear matrix factorization for collaborative filtering, which can be extended to tensor factorization. This can further extend JICO and other recommendation models based on the tensor operation to capture non-linear interactions in data.

Acknowledgment

The authors would like to thank the anonymous reviewers for their insightful comments, which have helped to improve the quality of this paper.

References

[1] D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, Recommender systems: an introduction, Cambridge University Press, 2010.

[2] G. Linden, B. Smith, J. York, Amazon. com recommendations: Item-to-item collaborative filtering, IEEE Internet Computing 7 (1) (2003) 76–80.

[3] J. L. Herlocker, J. A. Konstan, L. G. Terveen, J. T. Riedl, Evaluating collaborative filtering recommender systems, ACM Transactions on Information Systems (TOIS) 22 (1) (2004) 5–53.

[4] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web, ACM, 2001, pp. 285–295.

[5] M. D. Ekstrand, J. T. Riedl, J. A. Konstan, et al., Collaborative filtering recommender systems, Foundations and Trends® in Human–Computer Interaction 4 (2) (2011) 81–173.

[6] X. Su, T. M. Khoshgoftaar, A survey of collaborative filtering techniques, Advances in artificial intelligence 2009.

[7] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowledge-based systems 46 (2013) 109–132.

[8] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8).

[9] L. Baltrunas, B. Ludwig, F. Ricci, Matrix factorization techniques for context aware recommendation, in: Proceedings of the fifth ACM conference on Recommender systems, ACM, 2011, pp. 301–304.

[10] A. Mnih, R. R. Salakhutdinov, Probabilistic matrix factorization, in: Advances in neural information processing systems, 2008, pp. 1257–1264.

[11] X. Ma, P. Sun, G. Qin, Nonnegative matrix factorization algorithms for link prediction in temporal networks using graph communicability, Pattern Recognition 71 (2017) 361–374.

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

[13] J. Fan, T. W. Chow, Non-linear matrix completion, Pattern Recognition 77 (2018) 378–394.

[14] J. Fan, T. W. Chow, Matrix completion by least-square, low-rank, and sparse self-representations, Pattern Recognition 71 (2017) 290–305.

[15] F. O. de França, G. P. Coelho, F. J. Von Zuben, Predicting missing values with biclustering: A coherence-based approach, Pattern Recognition 46 (5) (2013) 1255–1266.

[16] J. Sublime, B. Matei, G. Cabanes, N. Grozavu, Y. Bennani, A. Cornuéjols, Entropy based probabilistic collaborative clustering, Pattern Recognition 72 (2017) 144–157.

[17] G. Adomavicius, A. Tuzhilin, Context-aware recommender systems, in: Recommender systems handbook, Springer, 2011, pp. 217–253.

[18] C. Palmisano, A. Tuzhilin, M. Gorgoglione, Using context to improve predictive modeling of customers in personalization applications, IEEE transactions on knowledge and data engineering 20 (11) (2008) 1535–1549.

[19] G. Adomavicius, R. Sankaranarayanan, S. Sen, A. Tuzhilin, Incorporating contextual information in recommender systems using a multidimensional approach, ACM Transactions on Information Systems (TOIS) 23 (1) (2005) 103–145.

[20] L. Baltrunas, 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.

[21] U. Panniello, A. Tuzhilin, M. Gorgoglione, C. Palmisano, 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.

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

[23] E. Zhong, W. Fan, 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.

[24] X. Liu, K. Aberer, Soco: a social network aided context-aware recommender system, in: Proceedings of the 22nd international conference on World Wide Web, ACM, 2013, pp. 781–802.

[25] A. Karatzoglou, X. Amatriain, L. Baltrunas, 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.

[26] S. Rendle, Z. Gantner, C. Freudenthaler, 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.

[27] S. Wang, C. Li, K. Zhao, H. Chen, Context-aware recommendations with random partition factorization machines, Data Science and Engineering 2 (2) (2017) 125–135.

[28] L. De Lathauwer, B. De Moor, J. Vandewalle, A multilinear singular value decomposition, SIAM journal on Matrix Analysis and Applications 21 (4) (2000) 1253–1278.

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

[30] H. Lu, K. N. Plataniotis, A. N. Venetsanopoulos, A survey of multilinear subspace learning for tensor data, Pattern Recognition 44 (7) (2011) 1540–1551.

[31] S. Rendle, Factorization machines, in: Data Mining (ICDM), 2010 IEEE 10th International Conference on, IEEE, 2010, pp. 995–1000.

[32] A. Coates, A. Y. Ng, The importance of encoding versus training with sparse coding and vector quantization, in: Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 921–928.

[33] T. V. Nguyen, A. Karatzoglou, L. Baltrunas, Gaussian process factorization machines for context-aware recommendations, in: Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval, ACM, 2014, pp. 63–72.

[34] S. Wang, C. Li, K. Zhao, H. Chen, Learning to context-aware recommend with hierarchical factorization machines, Information Sciences 409 (2017) 121–138.

[35] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, 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.

[36] Q. Liu, S. Wu, L. Wang, Cot: Contextual operating tensor for context-aware recommender systems., in: AAAI, 2015, pp. 203–209.

[37] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in neural information processing systems, 2013, pp. 3111–3119.

[38] M. Baroni, 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.

[39] R. Socher, B. Huval, C. D. Manning, 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.

[40] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, in: Proceedings of the 2013 conference on empirical methods in natural language processing, 2013, pp. 1631–1642.

[41] T. G. Kolda, B. W. Bader, Tensor decompositions and applications, SIAM review 51 (3) (2009) 455–500.

[42] H. Lu, K. N. Plataniotis, A. N. Venetsanopoulos, Multilinear Subspace Learning: Dimensionality Reduction of Multidimensional Data, CRC Press, 2013.

[43] 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.

[44] C. Ono, Y. Takishima, Y. Motomura, H. Asoh, Context-aware preference model based on a study of difference between real and supposed situation data, User Modeling, Adaptation, and Personalization (2009) 102–113.

[45] S. Wu, Q. Liu, L. Wang, T. Tan, Contextual operation for recommender systems, IEEE Transactions on Knowledge and Data Engineering 28 (8) (2016) 2000–2012.

[46] G. E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural computation 18 (7) (2006) 1527–1554.

[47] Q. Liu, S. Wu, L. Wang, Collaborative prediction for multi-entity interaction with hierarchical representation, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, ACM, 2015, pp. 613–622.

[48] N. D. Lawrence, R. Urtasun, Non-linear matrix factorization with gaussian processes, in: Proceedings of the 26th Annual International Conference on Machine Learning, ACM, 2009, pp. 601–608.