Incorporating Context Correlation into Context-aware Matrix Factorization

Yong Zheng, Bamshad Mobasher, Robin Burke
Center for Web Intelligence, DePaul University
Chicago, Illinois, USA
{yzheng8, mobasher, rburke}@cs.depaul.edu

Abstract

Context-aware recommender systems (CARS) go beyond traditional recommender systems, that only consider users' profiles, by adapting their recommendations also to users' contextual situations. Several contextual recommendation algorithms have been developed by incorporating context into recommendation algorithms in different ways. The most effective approaches try to model deviations in ratings among contexts, but ignore the correlations that may exist among these contexts. In this paper, we highlight the importance of contextual correlations and propose a correlation-based context-aware matrix factorization algorithm. Through detailed experimental evaluation we demonstrate that adopting contextual correlations leads to improved performance.

1 Introduction

Recommender systems (RS) are an effective ways to alleviate information overload by tailoring recommendations to users' personal preferences. Context-aware recommender systems (CARS) emerged to go beyond user preferences and also take into account the contextual situation of users in generating recommendations.

The standard formulation of the recommendation problem begins with a two-dimens-ional (2D) matrix of ratings, organized by user and item: *Users* $\times$ *Items* $\rightarrow$ *Ratings*. The key insight of context-aware recommender systems is that users' preferences for items may be a function of the context in which those items are encountered. Incorporating contexts requires that we estimate user preferences using a multidimensional rating function – *R*: *Users* $\times$ *Items* $\times$ *Contexts* $\rightarrow$ *Ratings* [Adomavicius *et al.*, 2011].

In the past decade, a number of context-aware recommendation algorithms have been developed that attempt to integrate context with recommendation algorithms. Some of the most effective methods, such as context-aware matrix factorization (CAMF) [Baltrunas *et al.*, 2011c] and contextual sparse linear methods (CSLIM) [Zheng *et al.*, 2014b; 2014c], incorporate a *contextual rating deviation* component into the recommendation algorithms. However, these methods generally ignore *contextual correlations* that may have bearing on how the rating behavior in different contexts is modeled. In this paper, we highlight the importance of contextual correlation, and propose a correlation-based context-aware matrix factorization algorithm.

2 Related Work

In context-aware recommender systems, context is usually defined as, "any information that can be used to characterize the situation of an entity [Abowd *et al.*, 1999]", e.g., *time* and *companion* may be two influential contexts in movie domain.

According to the two-part classification of contextual information by Adomavicius, *et al.* [Adomavicius *et al.*, 2011], we are concerned with *static* and *fully observable* contexts, where we already have a set of known contextual variables at hand which remain stable over time, and try to model users' contextual preferences to provide recommendations.

Several context-aware recommendation algorithms have been developed in the past decade. Typically, context can be taken into account using three basic strategies: pre-filtering, post-filtering and contextual modeling [Adomavicius *et al.*, 2011]. Pre-filtering techniques, such as context-aware splitting approaches [Baltrunas and Ricci, 2014; Zheng *et al.*, 2014a], simply apply contexts as filters beforehand to filter out irrelevant rating profiles. Post-filtering, on the other hand, applies the context as filters to the recommended items after the recommendation process. Or, contexts can be used as filters in the recommendation process, such as differential context modeling [Zheng *et al.*, 2012; 2013]. In contextual modeling, predictive models are learned using the full contextual data, and context information is used in the recommendation process. Most of the recent work, such as CAMF [Baltrunas *et al.*, 2011c], tensor factorization (TF) [Karatzoglou *et al.*, 2010] and CSLIM [Zheng *et al.*, 2014b; 2014c], belong to contextual modeling.

The most effective context-aware recommendation algorithms, such as CAMF and CSLIM, usually incorporate a contextual rating deviation term which is used to estimate users' rating deviations associated with specific contexts. Alternatively, the contextual correlation could be another way to incorporate contextual information. This idea has been introduced in the context of sparse linear method in our previous work [Zheng *et al.*, 2015], but it has not been explored as part of recommendation models based on matrix factorization.

3 Preliminary: Matrix Factorization and Context-aware Matrix Factorization

In this section, we introduce matrix factorization used in recommender systems, as well as the existing research on Context-Aware Matrix Factorization which utilizes the contextual rating deviations.

3.1 Matrix Factorization

Matrix factorization (MF) [Koren *et al.*, 2009] is one of the most effective recommendation algorithm in the traditional recommender systems. Simply, both users and items are represented by vectors. For example, $\overrightarrow{p_u}$ is used to denote a user vector, and $\overrightarrow{q_i}$ as an item vector. The values in those vectors indicate the weights on K (e.g., K = 5) latent factors. As a result, the rating prediction can be described by Equation 1.

$$\hat{r}_{ui} = \overrightarrow{p_u} \cdot \overrightarrow{q_i} \tag{1}$$

More specifically, the weights in $\overrightarrow{p_u}$ can be viewed as how much users like those latent factors, and the weights in $\overrightarrow{q_i}$ represent how this specific item obtains those latent factors. Therefore, the dot product of those two vectors can be used to indicate how much the user likes this item, where users' preferences on items are captured by the latent factors.

In addition, user and item rating biases can be added to the prediction function, as shown in Equation 2, where μ denotes the global average rating in the data set, b_u and b_i represent the user bias and item bias respectively.

$$\hat{r}_{ui} = \mu + b_u + b_i + \overrightarrow{p_u} \cdot \overrightarrow{q_i} \tag{2}$$

3.2 Context-aware Matrix Factorization

Consider the movie recommendation example in Table 1. There is one user $U1$, one item $T1$, and three contextual dimensions – Time (weekend or weekday), Location (at home or cinema) and Companion (alone, girlfriend, family). In the following discussion, we use *contextual dimension* to denote the contextual variable, e.g. "Location". The term *contextual condition* refers to a specific value in a dimension, e.g. "home" and "cinema" are two contextual conditions for "Location". A *context* or *contextual situation* is, therefore, a set of contextual conditions, e.g. {*weekend, home, family*}.

Table 1: Contextual Ratings on Movies

User	Item	Rating	Time	Location	Companion
U1	T1	3	weekend	home	alone
U1	T1	5	weekend	cinema	girlfriend
U1	T1	?	weekday	home	family

More specifically, let c_k and c_m denote two different contextual situations each of which is composed of a set of contextual conditions. We use $c_{k,l}$ to denote the l^{th} contextual condition in the context c_k. For example, assume c_k = {weekend, home, alone}, then $c_{k,2}$ is the contextual condition "home". In the following discussion, we continue to use those terms and symbols to describe corresponding contextual dimensions and conditions in the algorithms or equations.

The CAMF algorithm was proposed by Baltrunas et al. [Baltrunas *et al.*, 2011c]. The CAMF rating prediction function is shown in Equation 3.

$$\hat{r}_{uic_{k,1}c_{k,2}...c_{k,L}} = \mu + b_u + \sum_{j=1}^{L} B_{ijc_{k,j}} + \overrightarrow{p_u} \cdot \overrightarrow{q_i} \tag{3}$$

Assume there are L contextual dimensions in total, then $c_k = \{c_{k,1}c_{k,2}...c_{k,L}\}$ describes a contextual situation, where $c_{k,j}$ denotes the contextual condition in the j^{th} context dimension. Therefore, $B_{ijc_{k,j}}$ indicates the contextual rating deviation associated with item i and the contextual condition in the j^{th} dimension.

A comparison between Equation 2 and Equation 3 reveals that CAMF simply replaces the item bias b_i by a contextual rating deviation term $\sum\limits_{j=1}^{L} B_{ijc_{k,j}}$, where it assumes that the contextual rating deviation is dependent on items. Therefore, this approach is named as CAMF_CI. Alternatively, this deviation can also be viewed as being dependent on users, which replaces b_u by the contextual rating deviation term resulting in the CAMF_CU variant. In addition, CAMF_C algorithm assumes that the contextual rating deviation is independent of both users and items.

The parameters, such as the user and item vectors, user biases and rating deviations, can be learned by the stochastic gradient descent (SGD) method to minimize the rating prediction errors. In early work [Baltrunas *et al.*, 2011c], CAMF was demonstrated to outperform other contextual recommendation algorithms, such as the tensor factorization [Karatzoglou *et al.*, 2010].

4 Correlation-Based CAMF

Introducing the contextual rating deviation term is an effective way to build context-aware recommendation algorithms. Our earlier work [Zheng *et al.*, 2014b; 2014c] has successfully incorporated the contextual rating deviations into the sparse linear method (SLIM) to develop contextual SLIM (CSLIM) which was demonstrated to outperform the state-of-the-art contextual recommendation algorithms, including CAMF and tensor factorization.

As mentioned before, contextual correlation is an alternative way to build context-aware recommendation algorithms, other than modeling the contextual deviations. Our recent work [Zheng *et al.*, 2015] has made the first attempt to introduce the contextual correlation (or context similarity) into SLIM. In this paper, we further explore ways to develop correlation-based CAMF.

The underlying assumption behind the notion of "contextual correlation (or context similarity)" is that the more similar or correlated two contexts are, the more similar two recommendation lists for the same user within those two contextual situations should be. When integrated into matrix

Table 2: Example of a Correlation Matrix

	Time=Weekend	Time=Weekday	Location=Home	Location=Cinema
Time=Weekend	1	0.54	N/A	N/A
Time=Weekday	0.54	1	N/A	N/A
Location=Home	N/A	N/A	1	0.82
Location=Cinema	N/A	N/A	0.82	1

factorization, the prediction function can be described with Equation 4.

$$\hat{r}_{uic_k} = \overrightarrow{p_u} \cdot \overrightarrow{q_i} \cdot Corr(c_k, c_E) \tag{4}$$

where c_E denotes the empty contextual situation – the value in each contextual dimension is empty or "NA"; that is, $c_{E,1}$ = $c_{E,2}$ = ... = $c_{E,L}$ = N/A. Therefore, the function $Corr(c_k, c_E)$ estimates the correlation between the c_E and the contextual situation c_k where at least one contextual condition is not empty or "NA". Note that in Equation 3, the contextual rating deviation can be viewed as the deviation from the empty contextual situation to a non-empty contextual situation.

Accordingly, the user and item vectors, as well as the contextual correlations can be learned based using stochastic gradient decsent by minimizing the rating prediction errors. The loss function is described in Equation 5. Note that this is the general formulation for the loss function, where the term "$Corr^2$" should be specified and adjusted accordingly when the similarity of context is modeled in different ways. We will introduce three context similarity models in the next section.

$$\underset{p,q,Corr}{Minimize} \frac{1}{2}(r_{uic_k} - \widehat{r}_{uic_k})^2 + \frac{\lambda}{2}(\|\overrightarrow{p_u}\|^2 + \|\overrightarrow{q_i}\|^2 + Corr^2) \tag{5}$$

The remaining challenge is how to represent or model the correlation function in Equation 4. Different representations may directly influence the recommendation performance. In our recent work [Zheng *et al.*, 2015], we considered four strategies: Independent Context Similarity (ICS), Latent Context Similarity (LCS), Multidimensional Context Similarity (MCS) and Weighted Jaccard Context Similarity (WJCS) [1]. And those strategies can also be reused for the correlation-based CAMF. The prediction function in Equation 4 and loss function in Equation 5 can be updated accordingly when the correction is represented by different ways.

In this paper, we will not consider WJCS since it only uses the contextual dimensions with the same values, and the correlation in Equation 4 is measured between a context and the empty context. WJCS is therefore not applicable to the correlation-based CAMF, since it only works when the contextual conditions are the same. In the following, we introduce and compare the ICS, LCS and MCS approaches based on CAMF. Note that context correlation or similarity is assumed to be measured between any two contextual situations c_k and c_m. However, in the correlation-based CAMF, the correlation is actually measured between c_k and c_E, where c_E is the empty context.

[1]Here, *context similarity* is identical to *context correlation*

4.1 Independent Context Similarity (ICS)

An example of a correlation matrix can be seen in Table 2. With Independent Context Similarity, we only measure the context correlation or similarity between two contextual conditions when they lie on the same contextual dimension, e.g., we never measure the correlation between "Time = Weekend" and "Location = Home", since they are from two different dimensions. Each pair of contextual dimensions are assumed to be independent. In this case, the correlation between two contexts can be represented by the product of the correlations among different dimensions. For example, assume c_k is $\{$Time = Weekend, Location = Home$\}$, and c_m is $\{$Time = Weekday, Location = Cinema$\}$, the correlation between c_k and c_m can be represented by the correlation of <Time = Weekend, Time = Weekday> multiplied by the correlation of <Location = Home, Location = Cinema>, since those two dimensions are assumed as independent.

Assuming there are L contextual dimensions in total, the correlations can be depicted by Equation 6, where $c_{k,l}$ is used to denote the value of contextual condition in the l^{th} dimension in context c_k, and the "correlation" function is used to represent the correlation between two contextual conditions, which is also what to be learnt in the optimization. In other words, the correlation between two contexts is represented by the multiplication of the individual correlations between contextual conditions on each dimension.

$$Corr(c_k, c_m) = \prod_{l=1}^{L} correlation(c_{k,l}, c_{m,l}) \tag{6}$$

These correlation values (i.e., $correlation(c_{k,l}, c_{m,l})$) can be learned by the optimization process accordingly. The risk of this representation is that some information may be lost, if correlations are not in fact independent in different dimensions. For example, if users usually go to cinema to see romantic movies with their partners, the "Location" (e.g. at cinema) and "Companion" (e.g. partners) may be significantly correlated as a result.

4.2 Latent Context Similarity (LCS)

As noted earlier, contextual rating data is often sparse, since it is somewhat unusual to have users rate items repeatedly within multiple contextual situations. This poses a difficulty when new contexts are encountered. For example, the correlation between a *new pair* of contextual conditions <"Time=Weekend", "Time=Holiday"> may be required in the testing set, but it may not have been learned from the training data due to the sparsity problem. But, the correlation for two *existing pairs*, <"Time=Weekend", "Time=Weekday"> and <"Time=Weekday", "Time=Holiday">, may have been

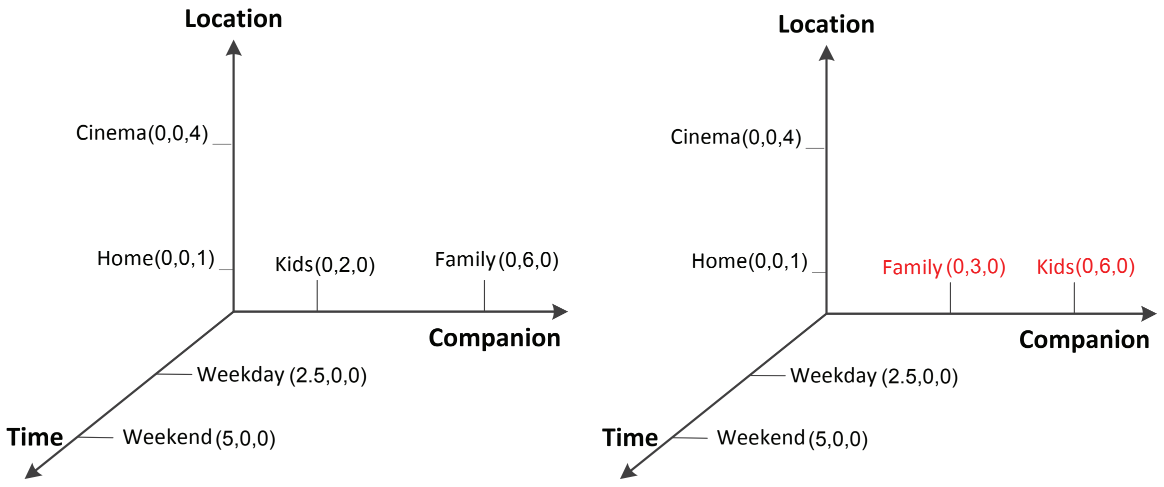

Figure 1: Example of Multidimensional Coordinate System

learned. In this case, this representation suffers from the contextual rating sparsity problem. Treating each dimension independently prevents the algorithm from taking advantage of comparisons that might be made across dimensions.

To alleviate this situation, we represent each contextual condition by a vector of weights over a set of latent factors (we use 5 latent factors in our experiments), where the weights are initialized at the beginning and learnt by the optimization process. The dot product between two vectors can be used to denote the correlation between each pair of contextual conditions. As a result, even if the newly observed pair does not exist in the training data, the weights in the vectors representing the two conditions (i.e., "Time=Weekend" and "Time=Holiday") will be learned and updated by the learning process over *existing pairs*, and the correlation for the *new pair* can be easily computed using the dot product. The correlation is given by

$$correlation(c_{k,l}, c_{m,l}) = V_{c_{k,l}} \bullet V_{c_{m,l}} \qquad (7)$$

where $V_{c_{k,l}}$ and $V_{c_{m,l}}$ denote the vector representation for the contextual condition $c_{k,l}$ and $c_{m,l}$, respectively, over the space of latent factors. We then use the same correlation calculation as shown in Equation 6. We call this approach the *Latent Context Similarity* (LCS) model. This approach was able to improve the performance of deviation-based CSLIM algorithms too [Zheng, 2015]. In contrast to the independent context similarity approach, LCS provides more flexibility, but it also has the added computational costs associated with learning the latent factors. In LCS, what has to be learnt in the optimization process are the vectors of weights representing each contextual condition.

4.3 Multidimensional Context Similarity (MCS)

In the multidimensional context similarity model, we assume that contextual dimensions form a multidimensional coordinate system. An example is depicted in Figure 1.

Let us assume that there are three contextual dimensions: time, location and companion. We assign a real value to each contextual condition in those dimensions, so that each condition can locate a position in the corresponding axis. In this case, a context (as a set of contextual conditions) can be viewed as a point in the multidimensional space. Accordingly, the distance between two such points can be used as the basis for a correlation measure. In this approach, the real values for each contextual condition are the parameters to be learned in the optimization process. For example, the values for "family" and "kids" are updated in the right-hand side of the figure. Thus, the position of the data points associated to those two contextual conditions will be changed, as well as the distance between the corresponding two contexts. Therefore, the correlation can be measured as the inverse of the distance between two data points. In our experiments, we use Euclidean distance to measure the distances, though other distance measures can also be used. The computational cost is directly associated with the number of contextual conditions in the data set, which may make this approach the highest-cost model. Again, the number of contextual conditions can be reduced by context selection.

5 Experimental Evaluation

In this section, we present our experimental evaluation and discuss the results.

5.1 Data Sets

We select three context-aware data sets with different numbers of contextual dimensions and conditions. *Restaurant* data [Ramirez-Garcia and Garca-Valdez, 2014] is comprised of users' ratings on restaurants in city of Tijuana, Mexico. *Music* data [Baltrunas *et al.*, 2011a] captures users' ratings on music tracks in different driving and traffic conditions. The *Tourism* data [Baltrunas *et al.*, 2011b] collects users' places of interest (POIs) from mobile applications. The characteristics of these data sets are summarized in Table 3. For more specific information about the contextual dimensions and conditions, please refer to the original papers using those data sets.

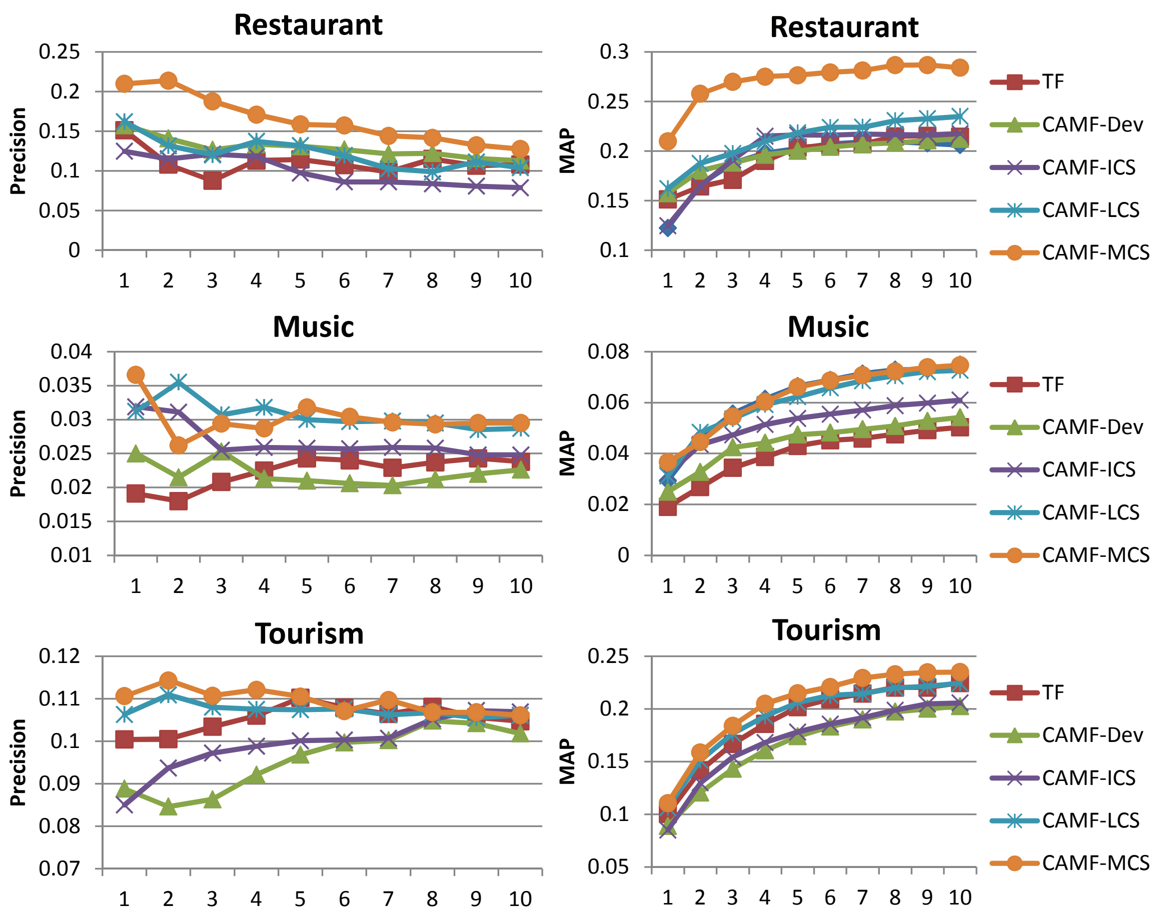

Figure 2: Experimental Comparison (x-axis denotes the number of recommendations.)

Table 3: Context-aware Data Sets

	Restaurant	Music	Tourism
Rating Profiles	50 users, 40 items 2314 ratings	40 users, 139 items 3940 ratings	25 users, 20 items 1678 ratings
# of contextual dimensions	2	8	14
# of contextual conditions	7	34	67
Rating Scale	1-5	1-5	1-5

5.2 Evaluation Protocols

We use a five-fold cross validation on our data sets, performing top 10 recommendation task and using precision and mean average precision (MAP) as the evaluation metrics. Precision is defined as the ratio of relevant items selected to number of items recommended in a specific context. MAP is another popular ranking metric which additional takes the ranks of the recommended items into consideration. It is calculated by Equation 8, where M denotes the number of the users, and N is the size of the recommendation list, where $P(k)$ means the precision at cut-off k in the item recommendation list, i.e., the ratio of number of users followed up to the position k over the number k, where m in $ap@N$ denotes the number of relevant items.

$$MAP@N = \sum_{i=1}^{M} ap@N/M, \text{ where } ap@N = \frac{\sum_{k=1}^{N} P(k)}{\min(m, N)} \quad (8)$$

We choose the CAMF [Baltrunas *et al.*, 2011c] as baseline in our experiments. We tried all three versions: CAMF_CI, CAMF_CU and CAMF_C, and only present the best performing one which is denoted as CAMF-Dev in the following sections. In addition, we also add tensor factorization (TF) [Karatzoglou *et al.*, 2010] as baseline.

5.3 Analysis and Findings

The results are depicted by Figure 2, where the correlation-based CAMF are represented by three algorithms accordingly: CAMF-ICS, CAMF-LCS and CAMF-MCS.

In terms of the best performing algorithm, CAMF-MCS outperforms the other ones for the restaurant and tourism data sets. It is able to obtain comparable results with CAMF-LCS in the music data. Compared with the deviation-based CAMF, the correlation-based CAMF can always outperform CAMF-Dev if the appropriate correlation modeling is applied. For example, in the restaurant data, CAMF-Dev works better than CAMF-ICS in precision, but CAMF-MCS

Table 4: Comparison Between CAMF and CSLIM

		Restaurant				Music				Tourism			
		CAMF-Dev	CAMF-MCS	CSLIM-Dev	CSLIM-MCS	CAMF-Dev	CAMF-MCS	CSLIM-Dev	CSLIM-MCS	CAMF-Dev	CAMF-MCS	CSLIM-Dev	CSLIM-MCS
Precision	@5	0.1309	0.1586	**0.2044**	**0.2151**	0.0210	0.0385	**0.0332**	**0.0385**	0.0968	0.1105	**0.1200**	**0.1522**
	@10	0.1130	0.1276	**0.1496**	**0.1723**	0.0226	0.0361	**0.0359**	**0.0361**	0.1018	0.1061	**0.1101**	**0.1265**
MAP	@5	0.2001	0.2765	**0.2889**	**0.2993**	0.0474	0.0661	**0.0871**	**0.0913**	0.1740	0.2148	**0.2199**	**0.3379**
	@10	0.2122	0.2840	**0.3128**	**0.3187**	0.0542	0.0747	**0.0995**	**0.1047**	0.2026	0.2350	**0.2442**	**0.3518**

outperforms CAMF-Dev significantly, which states that MCS is the better representation to model contextual correlations than the ICS for this data set.

Tensor Factorization (TF) only works better than some CAMF algorithms in the tourism data set, but CAMF-MCS is still the best one for this data. In TF, contextual dimensions are considered as extra dimensions in addition to the user and item dimensions, and they are assumed as independent with each other, where CAMF algorithms are actually dependent algorithms which further measures either contextual rating deviations or contextual correlations.

In our previous work [Zheng *et al.*, 2014c; 2015], we have incorporated contextual deviations and correlations into S-LIM respectively to formulate CSLIM algorithms. Therefore, it is necessary to compare the CAMF and CSLIM, which can be viewed from Table 4, where we only present the best performing deviation and correlation models. From the table, we can see that CSLIM outperforms CAMF significantly when the same strategy (either deviation or correlation modeling) is applied. And CSLIM-MCS works the best in general. It is not surprising, since earlier work [Ning and Karypis, 2011] has demonstrated that SLIM is able to outperform the state-of-the-art traditional recommendation algorithms, including the matrix factorization. Those results based on contextual recommendations further confirms this pattern, since CSLIM outperforms CAMF in view of the Table 4.

In short, those experimental results demonstrate that correlation-based CAMF is able to outperform deviation-based CAMF and the TF algorithm, but the representation for contextual correlation should be carefully selected. Generally, the multidimensional context similarity is the best choice to represent contextual correlations.

6 Conclusions and Future Work

In this paper, we highlighted the importance of contextual correlation and incorporate this notion into the matrix factorization technique to formulate correlation-based Context-Aware Matrix Factorization (CAMF) algorithm. Our experimental results reveal that correlation-based CAMF is able to outperform the standard deviation-based CAMF and the tensor factorization algorithms. In our future work, we plan to incorporate contextual correlation modeling strategies into more recommendation algorithms, such as the slope one recommender.

References

[Abowd *et al.*, 1999] Gregory D Abowd, Anind K Dey, Peter J Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *Handheld and ubiquitous computing*, pages 304–307. Springer, 1999.

[Adomavicius *et al.*, 2011] Gediminas Adomavicius, Bamshad Mobasher, Francesco Ricci, and Alexander Tuzhilin. Context-aware recommender systems. *AI Magazine*, 32(3):67–80, 2011.

[Baltrunas and Ricci, 2014] Linas Baltrunas and Francesco Ricci. Experimental evaluation of context-dependent collaborative filtering using item splitting. *User Modeling and User-Adapted Interaction*, 24(1-2):7–34, February 2014.

[Baltrunas *et al.*, 2011a] Linas Baltrunas, Marius Kaminskas, Bernd Ludwig, Omar Moling, Francesco Ricci, Aykan Aydin, Karl-Heinz Lüke, and Roland Schwaiger. Incarmusic: Context-aware music recommendations in a car. In *E-Commerce and Web Technologies*, pages 89–100. Springer, 2011.

[Baltrunas *et al.*, 2011b] Linas Baltrunas, Bernd Ludwig, Stefan Peer, and Francesco Ricci. Context-aware places of interest recommendations for mobile users. In *Design, User Experience, and Usability. Theory, Methods, Tools and Practice*, pages 531–540. Springer, 2011.

[Baltrunas *et al.*, 2011c] Linas Baltrunas, Bernd Ludwig, and Francesco Ricci. Matrix factorization techniques for context aware recommendation. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 301–304. ACM, 2011.

[Karatzoglou *et al.*, 2010] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 79–86. ACM, 2010.

[Koren *et al.*, 2009] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[Ning and Karypis, 2011] Xia Ning and George Karypis. SLIM: Sparse linear methods for top-n recommender systems. In *2011 IEEE 11th International Conference on Data Mining*, pages 497–506. IEEE, 2011.

[Ramirez-Garcia and Garca-Valdez, 2014] Xochilt Ramirez-Garcia and Mario Garca-Valdez. Post-filtering for a restaurant context-aware recommender system. In *Recent Advances on Hybrid Approaches for Designing Intelligent Systems*, volume 547, pages 695–707. Springer, 2014.

[Zheng *et al.*, 2012] Y. Zheng, R. Burke, and B. Mobasher. Differential context relaxation for context-aware travel recommendation. In *E-Commerce and Web Technologies*, pages 88–99. Springer, 2012.

[Zheng *et al.*, 2013] Y. Zheng, R. Burke, and B. Mobasher. Recommendation with differential context weighting. In

User Modeling, Adaptation, and Personalization, pages 152–164. 2013.

[Zheng *et al.*, 2014a] Y. Zheng, R. Burke, and B. Mobasher. Splitting approaches for context-aware recommendation: An empirical study. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, pages 274–279. ACM, 2014.

[Zheng *et al.*, 2014b] Y. Zheng, B. Mobasher, and R. Burke. CSLIM: Contextual SLIM recommendation algorithms. In *Proceedings of the 8th ACM Conference on Recommender Systems*, pages 301–304. ACM, 2014.

[Zheng *et al.*, 2014c] Y. Zheng, B. Mobasher, and R. Burke. Deviation-based contextual SLIM recommenders. In *Proceedings of the 23rd ACM Conference on Information and Knowledge Management*, pages 271–280. ACM, 2014.

[Zheng *et al.*, 2015] Yong Zheng, Bamshad Mobasher, and Robin Burke. Integrating context similarity with sparse linear recommendation model. In *User Modeling, Adaptation, and Personalization*, volume 9146 of *Lecture Notes in Computer Science*, pages 370–376. Springer Berlin Heidelberg, 2015.

[Zheng, 2015] Y. Zheng. Improve general contextual SLIM recommendation algorithms by factorizing contexts. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 929–930. ACM, 2015.