Approximate modeling of continuous context in factorization algorithms

Balázs Hidasi[†][*]
balazs.hidasi@gravityrd.com

Domonkos Tikk[†]
domonkos.tikk@gravityrd.com

[†]Gravity R&D
Mészáros utca 58/B.
1016 Budapest
Hungary

[*]Budapest University of Technology and Economics
Magyar Tudósok krt. 2
1117 Budapest
Hungary

ABSTRACT

Factorization based algorithms – such as matrix or tensor factorization – are widely used in the field of recommender systems. These methods model the relations between the entities of two or more dimensions. The entity based approach is suitable for dimensions such as users, items and several context types, where the domain of the context is nominal. Continuous and ordinal context dimensions are usually discretized and their values are used as nominal entities. While this enables the usage of continuous context in factorization methods, still much information is lost during the process. In this paper we propose two approaches for better modeling of the continuous context dimensions. *Fuzzy event modeling* tackles the problem through the uncertainty of the value of the observation in the context dimension. *Fuzzy context modeling*, on the other hand, enables context-states to overlap, thus certain observations are influenced by multiple context-states. Throughout the paper seasonality is used as an example of continuous context. We incorporate the modeling concepts into the iTALS algorithm, without degrading its scalability. The effect of the two approaches on recommendation accuracy is measured on five implicit feedback databases.

Categories and Subject Descriptors

I.2.6 [[**Artificial Intelligence**]]: Learning - Parameter Learning

General Terms

Algorithms, Experimentation

Keywords

context-awareness, recommender systems, factorization, continuous context

CaRR'14, April 13, 2014, Amsterdam, The Netherlands.

1. INTRODUCTION

Recommender systems are information filtering tools that help users in information overload to find interesting items. For modeling user preferences, classical approaches either use item metadata (content based filtering, CBF; [13]), or user–item interactions (collaborative filtering, CF; [19]). CF algorithms proved to be more accurate than CBF methods, if sufficient interaction data (or *events*) is available [14].

Latent factor based CF methods gained popularity due to their attractive accuracy and scalability [11]. They intend to capture user preferences by uncovering latent features that explain the observed user–item events (ratings). Models are created by the factorization of the partially observed user–item rating matrix, and the user preferences are approximated by the scalar product of the user and item factors. Matrix factorization (MF) methods may differ in the learning method and the objective function. For learning, MF methods may apply, e.g., alternating least squares (ALS; [3]), stochastic gradient [20], or a probabilistic framework [18].

The dualistic user–item based modeling concept can be extended by considering additional information that may influence the user preferences at recommendation; such data are together termed *contextual information*, or briefly *context* [1]. The hypothesis of context-aware recommendation systems is that the integration of context into the model may improve the modeling capacity and accuracy.

There are several factorization algorithms that can use one or more context dimensions besides the users and items for both the implicit and the explicit case ([10], [17], [7], [15], [5]). A common property of these factorization methods is that they assume that the context dimension is *categorical*. This is needed, analogously to the user and item dimensions, also the context dimension is represented by a set of entities, to which feature vectors are assigned in the factorization model. Generally, factorization methods are not able to cope with continuous dimensions. However, continuous context dimensions are frequent in contextual modeling. For example time based information, such as seasonality, is one of the most dominantly used context dimension [10]. Therefore factorization methods need to transform continuous context dimensions into categorical ones, but the commonly used transformations are lossy, therefore by design decreases the modeling capacity of the model; see also section 1.2.

In this paper, we propose two modeling approaches, *fuzzy event modeling* and *fuzzy context modeling*, that can better model the continuous context, yet can be embedded into the standard factorization frameworks. The methods are fully compatible with most factorization algorithms in the sense that those can be easily adapted to the cope with continuous dimensions using the proposed modeling approaches. We demonstrate this on the iTALS algorithm [7]. The modifications are tested against five implicit feedback datasets.

The rest of the paper is organized as follows. Next we briefly review the starting point of this work: section 1.2 describes the current methods for handling continuous context dimensions in factorization methods and shows why they fall short; section 1.3 present seasonality that will be used as the continuous context dimension throughout the paper; section 1.4 summarizes [7] to provide a brief introduction on the iTALS algorithm. Fuzzy event modeling and fuzzy context modeling approaches and their incorporation into the iTALS algorithm are described section 2. Section 3 contains the comparison of the modified and the original iTALS algorithms w.r.t. recommendation accuracy. Finally, section 4 summarizes this work and implies possible future research directions.

1.1 Notation

We will use the following notation in the rest of this paper:

- $A \circ B \circ \ldots$: The Hadamard (elementwise) product of A, B, ... The operands are of equal size, and the result's size is also the same. The element of the result at index $(i, j, k, \ldots)$ is the product of the element of A, B, ... at index $(i, j, k, \ldots)$.
- A_i: The i^{th} column of matrix A.
- $A_{i_1,i_2,\ldots}$: The $(i_1, i_2, \ldots)$ element of tensor/matrix A.
- K: The number of features, the main parameter of the factorization.
- D: The number of dimensions of the tensor.
- T: A D dimensional tensor that contains only zeroes and ones (preference tensor).
- W: A tensor with the same size as T (weight/confidence tensor).
- S_X: The size of T in dimension X (e.g. $X = U$ (Users)).
- N^+: The number of ratings (explicit case); non-zero elements in tensor T (implicit case).
- $M^{(X)}$: A $K \times S_X$ sized matrix. Its columns are the feature vectors for the entities in dimension X.

1.2 Current usage of continuous context

Current factorization methods work on dimensions defined on categorical domains (e.g. the set of item identifiers). Each categorical value is treated as an entity in the factorization method and is coupled with a feature vector.

Continuous context dimensions – such as time or location based information – are widely used in recommender systems and in factorization algorithms [10, 12]. However, their proper modeling is mostly neglected. The continuous domain is directly transformed into a categorical one. The first step is the discretization of the continuous domain by dividing it into several intervals. Then each interval is assigned an identifier and the set of identifiers is used as the domain of the context dimension.

The above modeling of context dimension has the following shortcomings:

- **Context-state rigidness:** The boundaries of the intervals are stark. The context-state of an event is determined by the interval it belongs to and is not influenced by its relative location in the interval nor by neighboring intervals. For example when the start of the interval is at 20:00 and there are two events with timestamps 19:58 and 20:03, then those events are probably similar w.r.t. the context, but assigned to different context-states. Events with context value falling close to boundary of an interval are probably not belonging solely to that context-state, but also to the neighboring one.
- **Context-state ordinality:** The context-states are treated independently from each other. In the continuous context dimension there is an ordering defined on the context values. Analogously in a discretized context dimension, a coarser ordering exists between context states. By this ordering the distance between context states can be defined, and gradual changes in user/item behavior according to the context can be modeled. Intuitively, we do not expect a sudden change in the user behavior once the clock strikes 20:00 – using the above example. By the nature of the nominal modeling of context dimension, this information is lost. Neighboring context-states have no effect on each other.[1] The ordinality of consecutive context-states would be desirable for continuous context dimensions.

In Section 2 we propose two approaches that tackle these drawbacks without using full continuous factorization methods.

1.3 Seasonality

Many application areas of recommender systems exhibit the seasonality effect, therefore seasonal data is an obvious choice as context [12]. Strong periodicity can be observed in most of the human activities: as people have regular daily routines, they also follow similar patterns in TV watching at different time of a day, they do their summer/winter vacation around the same time in each year. Taking the TV watching example, it is probable that horror movies are typically watched at night and animation is watched in the afternoon or weekend mornings. Seasonality can be also observed in grocery shopping or in hotel reservation data.

In order to consider seasonality, first we have to define the length of the season. The value of the dimension for an event is the timestamp of the event modulo the length of the season. Seasonality is a periodical continuous context, meaning that the values of the context for two events that are N times the season's length apart is the same. The length of the season depends on the data and is usually is a hyperparameter of the modeling. To create entities for the seasonality dimension *time bands* (bins) are needed to be created in the seasons. These time bands are the entities, i.e. the possible context-states. Time bands specify the time resolution of a season, which is also a data dependent hyperparameter. Time bands can be created with equal or different length. Events are associated with the time bands as context-states via their timestamps.

Common examples for seasonal context are the days of a week, months of the year, or every several hours of a day.

[1] We note that certain similarity can be observed if the number of events is high around the boundary and thus lots of events fall into both intervals. However this effect is neglected if the intervals are long or if the density of the events is higher inside the intervals than at the boundaries.

1.4 iTALS algorithm

The iTALS algorithm [7] is a context-aware factorization method. It factorizes the user-item-context(s) D dimensional tensor. The preferences are estimated by a three-way model. If $D = 3$, the predicted preferences are calculated as:

$$\hat{r}_{u,i,c} = 1^T \left(M_u^{(U)} \circ M_c^{(C)} \circ M_i^{(I)} \right) \tag{1}$$

It uses pointwise preference estimation, through weighted root mean squared error (wRMSE) based loss function:

$$L = \sum_{u=1,i=1,c=1}^{S_U,S_I,S_C} w_{u,i,c} \left(r_{u,i,c} - \hat{r}_{u,i,c} \right)^2 \tag{2}$$

where $w_{u,i,c}$ is the weight or confidence associated with the combination of the u^{th} user, i^{th} item and c^{th} context-state.

The weights are calculated as:

$$w_{u,i,c} = \begin{cases} \gamma \cdot \text{supp}(u,i,c) + w_0 \gg w_0, & \text{if } (u,i,c) \text{ is an event} \\ w_0, & \text{otherwise,} \end{cases} \tag{3}$$

where $\text{supp}(u,i,c)$ is the number of occurrences (support) of the (u,i,c) combination in the training data. This means that the presence of an event implies strong positive feedback, but lack of an event is just a very uncertain implication of negative feedback.

The learning is via alternating least squares (ALS) and its complexity is $O(K^2 N^+ K^3 (S_U + S_I + S_C))$. Using approximate least squares solvers the method's complexity can be improved to $O(KN^+ + K^2(S_U + S_I + S_C))$, as shown in [8]. Note that in practice $N^+ \gg S_U + S_I + S_C$, otherwise the problem is too sparse for factorization methods to handle. Thus the first term dominates the complexity expression, if K is small. Therefore the learning scales with K^2 or K with exact or approximate solvers respectively.

2. MODELING CONTINUOUS CONTEXT

To overcome the rigidness and ordinality problems, we propose two approaches to model continuous context dimensions in the factorization framework. Instead of a new, fully continuous factorization method we solve the aforementioned two shortcomings by proposing modeling techniques that lessen those effects and can be relatively easily embedded into existing factorization algorithms. The proposed two methods are termed *(fuzzy) event modeling* and *(fuzzy) context modeling*. The former is simpler and mainly targets the rigidness problem (it has some effect on the ordinality problem as well). The latter is more complex and targets both drawbacks simultaneously.

2.1 Fuzzy event modeling

The rigidness of the context-states is disadvantageous, because events that take similar values in the context dimension might be associated with different context-states if the boundary of the context-state intervals falls between them. Thus training data on the one side of the boundary will support the model building of the first context-state, while the training data of the other side supports the model of the other context-state. Intuitively, the problem is caused by the rigidness of the boundaries of the context-states' intervals and the instantaneous nature of the events.

The fuzzy event modeling approach overcomes the rigidness problem by assigning a validity interval to the formerly instantaneous events. Each event is valid in an interval of Δ radius around its original context value (i.e. in $[x-\Delta, x+\Delta]$, if its original context value is x). The validity intervals may intersect with multiple context-state intervals. Each event is associated with all context-states whose interval intersects with the event's validity interval. Context-states are trained using all events that are associated with them. However, the context-states are trained independently from one another, thus one can think of this method as cloning some of the events that are close to the boundaries of the context-states and distributing the clones between the neighboring context-states. Due the duplication of events that are near the context-state boundaries, the feature vectors of neighboring context-states become similar. Note that this approach solves the ordinality problem only partially, because the feature vectors of the context-states are trained independently.

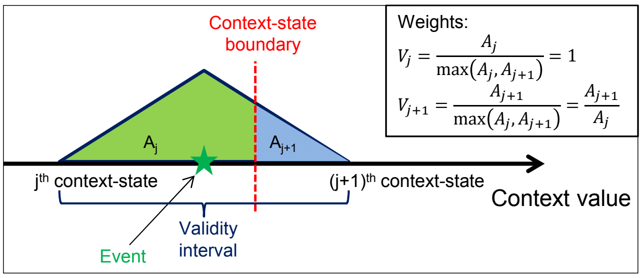

Figure 1: Weighting scheme with a triangular shape as example

This simple and straightforward approach can be easily coupled with every factorization algorithm. To apply fuzzy event modeling with the iTALS algorithm, the association of events with context-states should be translated to exact computation of preferences and confidences. There are several ways to compute these values; here we propose the following two:

- **Constant:** The value is the same in every associated context-state. (By preference computation the value is 1, by confidence computation it is $\gamma \cdot \text{supp}(u,i,c)$, using the original context-state to calculate the support.
- **Weighted:** The extent of the validity of an event may vary through its validity interval. For example, the further it is from the event's original value, the lower the extent of validity is. Shapes can be assigned to the validity intervals that describe the extent of validity in each point. (E.g. normal distribution, triangle, trapezoid, rectangle, etc.) We use the area under the shape as weights. The weights are scaled so that the largest of them is 1; see figure 1. The weights are used to modify the preference and/or the confidence by scaling them. The proposed normalization implies that the values in the original context-state[2] does not change, but the values in the other context-state are scaled down.

The computation of the preference and confidence is independent, either method can be used for each and the shapes

[2]In practical settings, the area under the shape will be the largest in the original context-state, therefore its weight will there be 1.

for computing the weights do not have to be the same.

The iTALS itself requires no modification, it only needs to be fed with the "extra" events. Therefore the scaling of the algorithm does not change. The running time increases slightly as the number of events increased due to the cloning. Note however, that the length of the validity interval should be shorter than half of the length of the context-state interval, since we do not want to associate events with unrelated contexts. Therefore each event is cloned at most once.

2.2 Fuzzy context modeling

The context modeling approach aims to overcome both the rigidness and ordinality problems. In this model, the intervals of the context states overlap and the events remain instantaneous. With the overlapping intervals, there are no strict boundaries between context-states, rather there are zones in which multiple context-states are valid. The weight of a context-state outside the overlapping zone is 1. The weights of the two context-states in the overlapping zone are governed by the weighting method (see figure 2). We use two simple weighting schemes:

- **Equal:** The weight is 0.5 for both context-states in the overlapping dimension.
- **Linear:** The weight of the previous context-state starts from 1 and decreases linearly to 0, the weight of the next context-state starts from 0 and linearly increases to 1 throughout the overlapping zone. The sum of the two weights at any given point is 1.

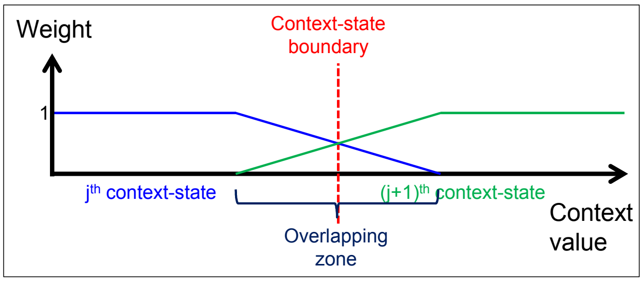

Figure 2: Example of overlapping context-states with linear weighting.

This approach modifies the preference prediction model by substituting $M_c^{(C)}$ by $\alpha(t)M_c^{(C)} + (1-\alpha(t))M_{c+1}^{(C)}$ in the model, where $\alpha(t)$ is the weight of the c^{th} context-state at the t value of continuous context dimension if t is in the interval of the c^{th} context-state. This means that in the overlapping zones preferences are estimated by using the linear combination of the feature vectors of the two overlapping context-states.

Incorporation of this technique into iTALS requires some work. First, the model should be modified. Let's use the original dimensions U, I and C. (Note: here C is the original discretized "rigid" context dimension.) Instead of a full continuous approach, a finer discretization (compared to C) of the context dimension is defined, denoted by X. An interval in X must intersect with exactly one interval from C. This step is needed, because the loss function of the iTALS should sum over discrete entities of a dimension. The finer the resolution is, the better is the approximation of the continuous case. The model in (1) is modified as follows:

$$\hat{r}_{u,i,x} = 1^T \left(M_u^{(U)} \circ \left(\alpha_x M_c^{(C)} + (1-\alpha_x) M_{c+1}^{(C)} \right) \circ M_i^{(I)} \right) \quad (4)$$

Here x is the identifier of the interval in the X dimension and $\alpha_x = \alpha(y)$, where y is the mean value of the x^{th} interval in X. The value of c is determined by x (the c^{th} interval of C contains the x^{th} interval of X). Note that preference is predicted with the resolution of X, thus allowing the usage of the overlapping and non-overlapping zones and different weight pairs for the context-state pairs of the overlapping zones. This also suggests why it is enough to make X with a relatively high resolution compared to C: the finer resolution is needed to model the overlapping zones properly. Also note that if C is a periodical context, such as seasonality, then the $(S_C)^{\text{th}}$ context-state overlaps with the 1^{st}.

The loss function must be modified as well, to sum over X instead of C. The loss of (2) is rewritten as follows:

$$\begin{aligned} L &= \sum_{u=1,i=1,x=1}^{S_U,S_I,S_X} w_{u,i,x} \left(r_{u,i,x} - \hat{r}_{u,i,x} \right)^2 = \\ &= \sum_{u=1,i=1,c=1}^{S_U,S_I,S_C} \sum_{x \in c} w_{u,i,x} \left(r_{u,i,x} - \hat{r}_{u,i,x} \right)^2, \end{aligned} \quad (5)$$

where $x \in c$ means that the c^{th} interval of C contains the x^{th} interval of X.

The computation of user and item feature vectors is slightly modified. The derivative of L by a user feature vector is:

$$\begin{aligned} &\frac{\partial L}{\partial M_u^{(U)}} = \\ &= \left(\sum_{i=1,c=1}^{S_I,S_C} \sum_{x \in c} w_{u,i,x} \left(M_i^{(I)} \circ \mathcal{M}_{c,x} \right) \left(M_i^{(I)} \circ \mathcal{M}_{c,x} \right)^T \right) M_u^{(U)} - \\ &- \sum_{i=1,c=1}^{S_I,S_C} \sum_{x \in c} w_{u,i,x} r_{u,i,x} M_i^{(I)} \circ \mathcal{M}_{c,x}, \end{aligned} \quad (6)$$

where $\mathcal{M}_{c,x} = \alpha_x M_c^{(C)} + (1-\alpha_x) M_{c+1}^{(C)}$. $\frac{\partial L}{\partial M_u^{(U)}} = 0$ can be solved efficiently for $M_u^{(U)}$ using the steps of the original algorithm [7]. The computation of an item feature vector can be obtained analogously (by switching users and items).

However, the computation of the context-state feature vectors is needed to be significantly modified. The derivative of L by a context-state feature vector is:

$$\begin{aligned}\frac{\partial L}{\partial M_c^{(C)}} = &\sum_{u=1,i=1,x\in c^{(0)}}^{S_U,S_I} w_{u,i,x}\mathcal{S}_{u,i}M_c^{(C)}+ \\ &+ \sum_{u=1,i=1,x\in c^{(-1)}}^{S_U,S_I} w_{u,i,x}\mathcal{S}_{u,i}\left(\alpha_x M_{c-1}^{(C)} + (1-\alpha_x)M_c^{(C)}\right) \\ &+ \sum_{u=1,i=1,x\in c^{(+1)}}^{S_U,S_I} w_{u,i,x}\mathcal{S}_{u,i}\left(\alpha_x M_c^{(C)} + (1-\alpha_x)M_{c+1}^{(C)}\right) \\ &- \sum_{u=1,i=1,x\in c^{(0)}}^{S_U,S_I} w_{u,i,x}r_{u,i,x}M_u^{(U)}\circ M_i^{(I)} \\ &- \sum_{u=1,i=1,x\in c^{(-1)}}^{S_U,S_I} (1-\alpha_x)w_{u,i,x}r_{u,i,x}M_u^{(U)}\circ M_i^{(I)} \\ &- \sum_{u=1,i=1,x\in c^{(+1)}}^{S_U,S_I} \alpha_x w_{u,i,x}r_{u,i,x}M_u^{(U)}\circ M_i^{(I)},\end{aligned} \tag{7}$$

where $c^{(0)}$ is the non-overlapping zone of the c^{th} context-state, $c^{(-1)}$ is the overlapping with the previous, $c^{(+1)}$ is the overlapping with the next context-state and $\mathcal{S}_{u,i} = \left(M_i^{(U)}\circ M_i^{(I)}\right)\left(M_u^{(U)}\circ M_i^{(I)}\right)^T$.

Based on equation (7), $\frac{\partial L}{\partial M_c^{(C)}} = 0$ can be rewritten as follows:

$$\mathcal{A}^{(c)}M_{c-1}^{(C)} + \mathcal{B}^{(c)}M_c^{(C)} + \mathcal{C}^{(c)}M_{c+1}^{(C)} = \mathcal{Y}^{(c)}, \tag{8}$$

where $\mathcal{A}^{(c)}$, $\mathcal{B}^{(c)}$, $\mathcal{C}^{(c)}$ are $K\times K$ matrices and $\mathcal{Y}^{(c)}$ is a vector of K length. $\mathcal{A}^{(c)}$, $\mathcal{B}^{(c)}$, $\mathcal{C}^{(c)}$ and $\mathcal{Y}^{(c)}$ can be computed efficiently, using similar steps to the original algorithm [7]. It is important to note that the context-state feature vectors can not be computed independently, because the computation of the c^{th} requires the $(c-1)^{\text{th}}$ and $(c+1)^{\text{th}}$ feature vectors. The feature vectors must be computed at the same time, resulting a system of linear equations of $S_CK\times S_CK$ size. However ,the coefficient matrix has a special structure: it is a tridiagonal block matrix for non-periodic context dimensions and cyclic tridiagonal block matrix for periodic contexts (see 9). The blocks in the matrix are $\mathcal{A}^{(c)}$, $\mathcal{B}^{(c)}$, $\mathcal{C}^{(c)}$ $K\times K$ sized matrices. Fortunately, these systems can be solved in $O(S_CN^3)$ time. [3] We use seasonality, therefore our system has a cyclic tridiagonal block coefficient matrix and we use the method introduced in [2] to solve the system.

$$\begin{pmatrix} \mathcal{B}^{(1)} & \mathcal{C}^{(1)} & \cdots & \mathcal{A}^{(1)} \\ \mathcal{A}^{(2)} & \mathcal{B}^{(2)} & \mathcal{C}^{(2)} & \cdots \\ \ddots & \ddots & \ddots & \\ \cdots & \mathcal{A}^{(S_C-1)} & \mathcal{B}^{(S_C-1)} & \mathcal{C}^{(S_C-1)} \\ \mathcal{C}^{(S_C)} & \cdots & \mathcal{A}^{(S_C)} & \mathcal{B}^{(S_C)} \end{pmatrix} \cdot \begin{pmatrix} M_1^{(C)} \\ M_2^{(C)} \\ \vdots \\ M_{S_C-1}^{(C)} \\ M_{S_C}^{(C)} \end{pmatrix} = \begin{pmatrix} \mathcal{Y}^{(1)} \\ \mathcal{Y}^{(2)} \\ \vdots \\ \mathcal{Y}^{(S_C-1)} \\ \mathcal{Y}^{(S_C)} \end{pmatrix} \tag{9}$$

The computation of $\mathcal{A}^{(c)}$, $\mathcal{B}^{(c)}$, $\mathcal{C}^{(c)}$ and $\mathcal{Y}^{(c)}$ does not increase the complexity, since they are computed with equal cost to the computation of the covariance matrices in the original algorithm. The computation of all context-state feature vectors takes $O(S_CN^3)$ time, therefore the complexity of the original algorithm does not change. The memory requirement of the algorithm increased, because the method introduced in [2] requires the storage of $\mathcal{B}^{(c)}$, $\mathcal{C}^{(c)}$ matrices and $\mathcal{Y}^{(c)}$ vectors. This requires $2S_CK^2 + S_CK$ space. This causes no problems in practice because K is a small number ($k < 500$, usually $k \in [20, 100]$) and S_C is usually also small, compared to S_U or S_I.

2.3 Differences between the two approaches

Note that assigning validity intervals to events and allowing context-states to overlap have apparently the same effect. Event modeling was introduced using the former and context modeling was presented using the latter, because of which approach explains which modeling better. The main difference is that by event modeling context-states remain mainly independent. Although through event duplication there is a loose connection between neighboring context states, but their feature vectors are learnt independently. This is in contrast with the context modeling, where the events that fall in the overlapping zones tie the subsequent context-states together. Therefore the feature vectors of the context-states must be learnt simultaneously which allows the approach to tackle the ordinality problem.

3. RESULTS

We compared the original and modified iTALS variants using five implicit feedback datasets: three public (LastFM 1K, [4]; TV1, TV2, [6]) , and 2 proprietary (Grocery, VoD). The properties of the data sets are summarized in Table 1. The column "Multi" shows the average multiplicity of user–item pairs in the training events.[4] The train–test splits are time-based: the first event in the test set is after the last event of the training set. The length of the test period was selected to be at least one day, and depends on the domain and the frequency of events. We used the artists as items in LastFM.

Our primary evaluation metric is recall@20, because it is a good proxy for assessing the accuracy of live recommendations. Recall is defined as the ratio of relevant recommended items and relevant items. An item is considered relevant for

[3] $\mathcal{A}^{(c)}$, $\mathcal{B}^{(c)}$, $\mathcal{C}^{(c)}$ are symmetric, positive definite matrices. This property can be used as well to achieve some speeding-up.

[4] TV1 and TV2 data might have been filtered for duplicate events.

Table 1: Main properties of the data sets

Dataset	Domain	Training set #Users	#Items	#Events	Multi	Test set #Events	Length
Grocery	E-grocery	24947	16883	6238269	3.0279	56449	1 month
TV1	IPTV	70771	773	544947	1.0000	12296	1 week
TV2	IPTV	449684	3398	2528215	1.0000	21866	1 day
VoD	IPTV/VoD	480016	46745	22515406	1.2135	1084297	1 day
LastFM	Music	992	174091	18908597	21.2715	17941	1 day

Table 2: Recall@20 values for algorithm variants and improvements over the original algorithm

Dataset	iTALS	Event modeling	Context modeling (equal)	Context modeling (linear)
Grocery	0.1062	0.1107 (+4.25%)	0.1480 (+39.40%)	0.1456 (+37.11%)
TV1	0.1371	0.1414 (+3.14%)	0.2671 (+94.78%)	0.2788 (+103.32%)
TV2	0.1794	0.2242 (+24.96%)	0.2582 (+43.89%)	0.2009 (+11.98%)
VoD	0.0339	0.1027 (+203.12%)	0.2023 (+496.88%)	0.1298 (+282.89%)
LastFM	0.0994	0.1028 (+3.48%)	0.3178 (+219.80%)	0.3047 (+206.58%)

a user if there is an event in the test data with the given user and item. Recall does not take into account the position of an item on the recommendation list. We estimate that users are exposed to 20 recommendations in average during a visit (e.g. 4 pageviews, 5 items per recommendation), therefore we choose cutoff at 20. In practice recommended items are usually randomly selected from the first N elements of the ranked item list. N is small but larger than the number of recommendation boxes (e.g.: $N = 20$). We reward when the user clicks one of the items, but it is irrelevant whether it was the first or the N^{th} item in our ranking. Therefore recall@N suits the offline evaluation of recommender algorithms from the practical viewpoint.

Seasonality was used as the continuous context dimension in the experiments. The length of the season is one week and one day for Grocery and other datasets respectively. Time bands of equal length were used within the season. The length of a time band was one day for Grocery and four hours for the other datasets.

Hyperparameters, such as regularization coefficients, were optimized using a part of the training data as validation set. Then the methods were retrained on the whole training set using the optimal hyperparameters. In addition to the hyperparameter of the original algorithm, the length of overlapping was optimized for fuzzy context modeling; for fuzzy event modeling the length of the event validity, preference and confidence computation (i.e. shape of the event validity and weighting type) was optimized as well. We found that for event modeling, usually confidences should be kept constant for each intersected context-state, while preferences should be weighted. This setting gave the best results in three out of five cases. However, for some datasets other setting might be more preferable. With context modeling, we ran separate experiments with equal and linear weighting.

Table 2 shows the results of the experiments. With event modeling $\sim 3-4\%$ improvement can be achieved in terms of recall@20 three out of five cases. The improvements on TV2 and LastFM are, however, more remarkable. We reported in earlier work [7] that the daily seasonality with 4 hours long time bands does not suit the TV2 dataset as the result of both iTALS and even the context-aware baseline is worse than that of the standard matrix factorization. The reason for this is partly the rigidness of the context-states, that does not suit the data well. Once the rigidness of context-states is waived, the usage of context no longer hinders the learning, but in fact improved the performance, yielding $\sim$ 25% improvement in the case of TV2 data set. The huge improvement on LastFM is also due to the not well suited context information. Time bands of four hours are too long for that dataset (see [8]), because music tracks are shorter than VODs, therefore the users' behavior can change more rapidly. Fuzzy modeling – even with long timebands – allows to capture this behavior more accurately. The results for context modeling are definitely better than that of the event modeling. This difference can be attributed to the fact that the effect of context-states on each other is much higher in context modeling because of the joint computation of the feature vectors. Equal weighting seems to be more efficient than linear (4 out of 5).

It is interesting to note that the optimal value for overlapping/event validity was between 75–100% of length of the time band. This means that the "ordinality" of the context feature is important for achieving good results.

4. CONCLUSION

In this paper we proposed two approaches to model continuous context dimensions in factorization methods.

The fuzzy event modeling approach extends the validity of the instantaneous events in the context dimension to a finite interval centered around the value of the event. Thus an event may belong to multiple context-states as its validity interval intersects with multiple context-state intervals. If this occurs, the event is used for the training of all corresponding context-states' feature vector separately. The fuzzy context modeling approach allows context-states of overlapping intervals, thus some events are influenced by two context-states. These events are used to train both context-states simultaneously. The former method solves the rigidness of context-states, while the latter also lessens the lack of ordinality of the context-states.

The iTALS algorithm was modified to incorporate these modeling concepts, without degrading its complexity. Using seasonality as the continuous context dimension, we measured the effect of the proposed approaches on the accuracy of recommendations using 5 implicit feedback databases. Event modeling was found to be less effective in increasing the accuracy, however it can be simply incorporated into any factorization method and hardly increases the complexity of the original algorithm. On the other hand, context modeling caused significant improvements in the accuracy, but its incorporation into factorization methods requires more work and the resulting algorithm becomes significantly more complex.

Future research in this area includes (a) examination of the modeling concepts in other learning strategies (e.g. BPR [16]), as well as with additional factorization models (e.g. the ones in [9]); (b) true continuous modeling of the context dimension instead of approximate solutions.

Acknowledgements

The work leading to these results has received partial funding from the European Union's Seventh Frame work Programme (FP7/2007-2013) under CrowdRec Grant Agreement nř 610594

5. REFERENCES

[1] Adomavicius, G., and Tuzhilin, A. Context-aware recommender systems. In *Recsys'08: ACM Conf. on Recommender Systems* (2008), 335–336.

[2] Batista, M. A cyclic block-tridiagonal solver. *Advances in Engineering Software 37*, 2 (2006), 69–74.

[3] Bell, R., and Koren, Y. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *ICDM'07: IEEE Int. Conf. on Data Mining* (2007), 43–52.

[4] Celma, O. *Music Recommendation and Discovery in the Long Tail.* Springer, 2010.

[5] Chen, T., Zhang, W., Lu, Q., Chen, K., Zheng, Z., and Yu, Y. SVDFeature: A toolkit for feature-based collaborative filtering. *Journal of Machine Learning Research 13* (2012), 3619–3622.

[6] Cremonesi, P., and Turrin, R. Analysis of cold-start recommendations in IPTV systems. In *Proc. of the 2009 ACM Conference on Recommender Systems* (2009).

[7] Hidasi, B., and Tikk, D. Fast ALS-based tensor factorization for context-aware recommendation from implicit feedback. In *Proc. of the ECML-PKDD, Part II*, no. 7524 in LNCS. Springer, 2012, 67–82.

[8] Hidasi, B., and Tikk, D. Context-aware recommendations from implicit data via scalable tensor factorization. *ArXiv e-prints* (2013).

[9] Hidasi, B., and Tikk, D. General factorization framework for context-aware recommendations. *ArXiv e-prints* (Jan. 2014).

[10] Karatzoglou, A., Amatriain, X., Baltrunas, L., and Oliver, N. Multiverse recommendation: N-dimensional tensor factorization for context-aware collaborative filtering. In *Recsys'10: ACM Conf. on Recommender Systems* (2010), 79–86.

[11] Koren, Y., and Bell, R. Advances in collaborative filtering. In *Recommender Systems Handbook*, F. Ricci et al., Eds. Springer, 2011, 145–186.

[12] Liu, N. N., Zhao, B. C. M., and Yang, Q. Adapting neighborhood and matrix factorization models for context aware recommendation. In *CAMRa'10: Workshop on Context-Aware Movie Recommendation* (2010), 7–13.

[13] Lops, P., Gemmis, M., and Semeraro, G. Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook*. Springer, 2011, 73–105.

[14] Pilászy, I., and Tikk, D. Recommending new movies: Even a few ratings are more valuable than metadata. In *Recsys'09: ACM Conf. on Recommender Systems* (2009), 93–100.

[15] Rendle, S. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST) 3*, 3 (2012), 57.

[16] Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. BPR: Bayesian personalized ranking from implicit feedback. In *UAI '09: 25th Conf. on Uncertainty in Artificial Intelligence* (2009), 452–461.

[17] Rendle, S., and Schmidt-Thieme, L. Pairwise interaction tensor factorization for personalized tag recommendation. In *WSDM'10: ACM Int. Conf. on Web Search and Data Mining* (2010), 81–90.

[18] Salakhutdinov, R., and Mnih, A. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems 20*. MIT Press, 2008.

[19] Su, X., and Khoshgoftaar, T. M. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* (2009), Article ID 421425 (1–19).

[20] Takács, G., Pilászy, I., Németh, B., and Tikk, D. Major components of the Gravity recommendation system. *SIGKDD Explor. Newsl. 9* (December 2007), 80–83.