Basket-Sensitive Personalized Item Recommendation

Duc-Trong Le†, Hady W. Lauw† and Yuan Fang‡
†School of Information Systems, Singapore Management University, Singapore
‡Institute for Infocomm Research, A*STAR, Singapore
{ductrong.le.2014, hadywlauw}@smu.edu.sg, yfang@i2r.a-star.edu.sg

Abstract

Personalized item recommendation is useful in narrowing down the list of options provided to a user. In this paper, we address the problem scenario where the user is currently holding a basket of items, and the task is to recommend an item to be added to the basket. Here, we assume that items currently in a basket share some association based on an underlying latent need, e.g., ingredients to prepare some dish, spare parts of some device. Thus, it is important that a recommended item is relevant not only to the user, but also to the existing items in the basket. Towards this goal, we propose two approaches. First, we explore a factorization-based model called BFM that incorporates various types of associations involving the user, the target item to be recommended, and the items currently in the basket. Second, based on our observation that various recommendations towards constructing the same basket should have similar likelihoods, we propose another model called CBFM that further incorporates basket-level constraints. Experiments on three real-life datasets from different domains empirically validate these models against baselines based on matrix factorization and association rules.

1 Introduction

Traditional recommender systems are premised on modeling the associations between users and items. For instance, collaborative filtering learns a user’s preference from other users who have expressed similar behaviors. Recent works are mostly based on matrix factorization [Koren et al., 2009], where every user $u_i$ is associated with a latent vector $x_i \in \mathbb{R}^K$ in $K$ dimensions, and every product $v_j$ is associated with a latent vector $y_j \in \mathbb{R}^K$. A user is recommended the item $v_j$ with the highest inner product $x_i^T v_j$. The implicit assumption is that a user is interested in only one item at a time.

In reality, user buys an item to address a specific need, which frequently could only be fulfilled by multiple related items. When shopping for clothes, a user may be looking for matching top, bottom, and accessories. To make a cake, a user needs flour, milk, eggs, and sugar, among other ingredients. Someone on an errand may wish to visit several places in one trip: dropping mail, collecting laundry, having lunch, and buying groceries. In these cases, the items (e.g., products, places, songs) sought by users are not independent.

Problem. We are interested in the notion of basket. Given a user who is holding a basket of items, we seek to recommend another item to add to the basket. This problem is relevant in both online and offline scenarios. For instance, an online shopper at Amazon.com, or an offline shopper at an upcoming Amazon Go physical store, may be recommended relevant products based on her current cart. In brick-and-mortar supermarkets, RFID-tagged items and smart shopping carts [Yewatkar et al., 2016] allow real-time recommendation of items based on a user’s smart cart. A basket may also refer to items adopted by a user within a specific period of time, e.g., points of interest visited in a trip. While seeking the latent need represented by a basket of items, recommendation shall still be personalized, as a user may have preferences as to the exact items involved (a specific brand, size, color, etc.).

The literature on market basket analysis is dominated by association rule mining [Agrawal et al., 1994]. For recommendation, we can mine historical transactions for rules in the form of $B_i \Rightarrow v_j$, where $B_i$ is a set of items currently in the user’s basket, and $v_j$ would be the recommended item. The set $B_i \cup \{v_j\}$ must occur in at least some minimum number of transactions. Confidence is the fraction of the transactions that contain $B_i \cup \{v_j\}$, among the transactions that contain $B_i$. An item with a higher confidence is higher on the recommendation list. Association rule-based approach suffers from a couple of shortcomings. First, the rules are “rigid” in that items in $B_i$ must all occur in the same transaction. Thus, it may not model associations that have not been previously seen, but could have been inferred. Second, the rules are general and apply to all users. While there exist ways to make it “personalized” [Sarwar et al., 2000] by ensuring that the rule is supported by a user’s historical transaction before using it for recommendation, the model itself essentially does not learn personalized association among items. We seek to address these shortcomings with a factorization approach.

Approach. We advocate an approach that factorizes basket-level associations. As our first contribution, we propose a model that we call BASKET-SENSITIVE FACTORIZATION MACHINE or BFM, which models the recommendation
as a function of four types of associations. The first is association between the user and the target item to be recommended (where most matrix factorization approaches stop). In addition, we model association between the target item and each item currently in the basket, association among basket items, and association between the user and each basket item. We investigate empirically which associations are most useful.

While BFM captures the notion of relationship among items within a basket, we further observe relationship among baskets with similar intent. Continuing an earlier example, we investigate empirically which associations are most useful. The first is association between the user and each basket item. We may be recommending different items (eggs in one case, and sugar in the other), the suggested instances are addressing similar needs. As our second contribution, we propose a set of constraints to BFM to make the likelihood of recommendations that eventually belong to the same basket similar. We refer to this second model as CONSTRAINED BFM or CBFM, and investigate empirically whether the constraints are effective.

**Organizations.** This paper is organized as follows. In Section 2, we review the literature on recommendations related to baskets, as well as on other types of associations. We then describe our first model BFM in Section 3, as well as how its parameters could be learned. This is followed in Section 4 by a discussion on the second model CBFM. Through experiments on three real-life datasets from different domains, we conduct an empirical analysis of BFM and CBFM in Section 5, which also includes a comparison to an association rule-based baseline. We conclude the paper in Section 6.

### 2 Related Work

The original aim of association rule mining is not recommendation, but finding insightful associations from transaction data. The research focus was mainly on computational efficiency, with pruning strategies such as Apriori [Agrawal et al., 1994] or FP-tree [Han et al., 2000]. The rules are general rules, and are not personalized. Some works described ways to use association rules for personalized item recommendations. For comparison in Section 5, we follow the approach in [Sarwar et al., 2000] as a baseline. Other works inspired by association rules are not directly comparable. [Kim and Kim, 2003] used information on product categories for multi-level rules. [Wang et al., 2014] considered association rules across baskets, instead of within a basket. [Pradel et al., 2011] used bigram rules, with an item each in antecedent and consequent. [Li et al., 2009] proposed non-personalized recommendation based on random walk. It is different from ours where the recommendation is sensitive to both the user and the basket.

Another orthogonal direction is to recommend a user’s next basket. The key association is sequence based on time. One approach is based on integrating matrix factorization and Markov chains [Rendle et al., 2010]. Subsequent work applies recurrent neural networks [Yu et al., 2016] or a hierarchical representation model [Wang et al., 2015]. In contrast, our intention is to predict which item to be added into the current basket, and the key is correlation among items within the basket. Another problem is bundle recommendation [Zhu et al., 2014] to recommend a bundle of items. This is akin to next-basket recommendation, a different scenario from ours.

There are other non-basket associations that are not the focus of our work. One is the sequence of items [Le et al., 2016; Xiang et al., 2010]. Another is taxonomy-induced associations [Shan et al., 2012; Koenigstein et al., 2011]. There is also similarity- or co-occurrence-induced associations [Liu et al., 2015; Liang et al., 2016], including content-based recommendation [Pazzani and Billsus, 2007]. They recommend items independently, whereas we factor in the items in the user’s basket to arrive at the personalized recommendation.

### 3 Basket-Sensitive Factorization Machine

Consider $N$ users $U = \{u_1, u_2, ..., u_N\}$ and $M$ items $V = \{v_1, v_2, ..., v_M\}$. Given a user $u_i \in U$, a basket $B_i \subseteq V$ is defined as a subset of items that $u_i$ is currently “holding”. We refer to them as basket items. For instance, these could be items in the user’s shopping cart or places already visited by the user on that day. Our objective is to recommend a target item $v_j \in V \setminus B_i$ (or a ranked list of items) to $u_i$. We seek to learn a real-valued function $F(u_i, B_i, v_j; \Theta)$, such that if $F(u_i, B_i, v_j; \Theta) > F(u_i, B_i, v_{j'}; \Theta)$, then the target item $v_j$ is preferable to $v_{j'}$, and would be more likely to be recommended to $u_i$. $\Theta$ denotes the parameters of the function.

$$F(u_i, B_i, v_j; \Theta) \propto x_i^T y_j$$ (1)

Matrix factorization essentially assumes that the basket items are irrelevant, implying that a user chooses items independently of one another. In practice, we expect that items in a basket may be associated with one another.

**Basket Item and Target Item.** It is important to model the associations between items in the basket and the target item. For instance, a supermarket shopper who is picking up ingredients for curry would be recommended items differently from when she is picking up ingredients for cake. To model the influence of a basket item on the choice of the target item, we further include in $\Theta$ a latent vector $z_j \in \mathbb{R}^K$ for every item $v_j \in V$. As opposed to $y_j$ that models $v_j$’s behavior as a target item, $z_j$ models $v_j$’s behavior as a basket item. Without prior knowledge, we assume that all items in the basket would have an influence on the choice of the target item.

$$F(u_i, B_i, v_j; \Theta) \propto \sum_{v_k \in B_i} y_j^T z_k$$ (2)
Among Basket Items. A basket of items may not always share a strong association among themselves. On one occasion, a shopper may buy a complete set of ingredients for some dish. On another occasion, the shopper may pick up loose ends, resulting in a basket of less related items. The strength of association among items in the basket may influence the choice of the target item. We model this as well.

\[ F(u, B, v; \Theta) \propto \sum_{(v_k \neq v_j) \in B} z_k^T z_k' \]  

(3)

User and Basket Item. For completeness, we also model the association between the user and each basket item.

\[ F(u, B, v; \Theta) \propto \sum_{v_k \in B} x_i^T z_k \]  

(4)

This association is potentially redundant if the associations between the user and the target item, as well as between the target item and each basket item are already modeled.

Overall Function. We now encapsulate the above association types into one overall function as follows.

\[ F(u, B, v; \Theta) \propto \gamma_1 \cdot x_i^T y_j + \gamma_2 \cdot \sum_{v_k \in B} y_j^T z_k \]  

(5)

+ \gamma_3 \cdot \sum_{(v_k \neq v_j) \in B} z_k^T z_k' + \gamma_4 \cdot \sum_{v_k \in B} x_i^T z_k

For flexibility in whether to incorporate an association type, we indicate each association type with a binary variable \( \gamma_1, \gamma_2, \gamma_3, \gamma_4 \in \{0, 1\} \) to be specified according to each application scenario. We will experiment with different combinations of association types to see which are most useful.

Prediction. Once the parameters \( \Theta \) are learned, given a user \( u \) and a basket \( B \), we construct a recommendation list of target items in the order of decreasing \( F(u, B, v; \Theta) \).

3.2 Parameter Learning

We are given a set of tuples \( T \), where each \( t = (u_i, B_i, v_j, \delta) \in T \) denotes a user \( u_i \) holding a basket \( B_i \). If \( \delta = 1 \), the user ends up adopting a target item \( v_j \). If \( \delta = -1 \), the user does not adopt \( v_j \). A user may have multiple tuples in \( T \). The goal is to learn the parameters in \( \Theta \), i.e., \( \{x_i\}_{i \in U}, \{y_j\}_{j \in V} \), and \( \{z_k\}_{k \in \mathbb{R}^t} \) to maximize the likelihood of observing \( T \).

We make the interesting observation that the model parameters can be mapped into a factorization machine or FM [Rendle, 2012]. Let \( h \) be a vector of length \( p \), with binary elements, i.e., \( h_i \in \{0, 1\} \). A second-order FM is as follows.

\[ F(h) = \mu_0 + \sum_{i=1}^p \mu_i h_i + \sum_{i=1}^p \sum_{j=i+1}^p h_i h_j (\phi_i^T \phi_j) \]  

(6)

The parameters include the global bias \( \mu_0 \) and a bias coefficient \( \mu_i \) for each component. Each \( \phi_i \in \mathbb{R}^K \) is a \( K \)-dimensional latent vector associated with the \( i \)-th component.

We transform our model into the appropriate factorization machine. For \( t = (u_i, B_i, v_j, \delta) \in T \), we construct a binary vector \( h^t \) of length \( p \), where \( p = N + 2M \). The first \( N \) terms in \( h^t \) are for the presence of a user. We have \( h^t_i = 1 \). The next \( M \) terms in \( h^t \) are for the target item. We have \( h^t_{N+i} = 1 \). The last \( M \) terms are for the basket items. For each basket item \( v_k \in B_i \), we have \( h^t_{N+M+k} = 1 \). All other elements of \( h^t \) are zeros. The latent vectors of this factorization machine stand for those of BFM. \( \phi_i \) stands for a user latent vector \( x \) when \( i \leq N \), for a target item latent vector \( y \) when \( N < i \leq N + M \), and for a basket item latent vector \( z \) when \( N + M < i \).

For BFM, the function \( F(u, B, v; \Theta) \) in Equation 5 is effectively transformed into Equation 7. \( \Theta \) denotes the \( \phi_i \)'s that also stand for \( x, y, z \). The addition of biases \( \mu_0 \) and \( \mu_i \)'s is appropriate, and it is a common practice in matrix factorization-based recommendation [Koren et al., 2009].

\[ F(h; \Theta) = \mu_0 + \sum_{i=1}^p \mu_i h_i + \gamma_1 \sum_{i=1}^N \sum_{j=N+1}^{N+M} h_i h_j (\phi_i^T \phi_j) \]  

\[ + \gamma_2 \sum_{i=N+1}^{N+M} \sum_{j=N+M+1}^{N+M+2} h_i h_j (\phi_i^T \phi_j) \]  

\[ + \gamma_3 \sum_{i=N+M+1}^{N} \sum_{j=N+M+1}^{N+1} h_i h_j (\phi_i^T \phi_j) \]  

\[ + \gamma_4 \sum_{i=1}^N \sum_{j=1}^N h_i h_j (\phi_i^T \phi_j) \]  

(7)

To learn from training data \( T \), we would like \( F(h^t; \Theta) \) to be high when \( \delta = 1 \), and to be low when \( \delta = -1 \). To penalize errors during training, we adopt the following optimization criterion incorporating a logistic loss function.

\[ \text{OPT}_{\text{BFM}}(T) = \arg\min_{\Theta} \left[ \sum_{t \in T} -\ln(\sigma(F(h^t; \Theta) \times t, \delta)) + \sum_{\theta \in \Theta} \lambda_0 \theta^2 \right] \]  

(8)

where \( \sigma(a) = 1/(1 + e^{-a}) \) is the sigmoid function, and \( \lambda_0 \in \mathbb{R^+} \) is the regularization coefficient for \( \theta \).

The parameters could be estimated via several methods, e.g., stochastic gradient descent (SGD), alternating least-squares and Markov Chain Monte Carlo [Rendle, 2012].

4 Constrained BFM or CBFM

We describe CONstrained BFM or CBFM that incorporates constraints relating baskets of similar intent. Intuitively, if a user shops for a number of items to fulfill a need, conceivably on different occasions the user may put items in different sequences and construct different intermediate baskets that make up the same collection of items. For example, if the intent is served by four items \( v_1, v_2, v_3, v_4 \), on one occasion when a user’s basket contains \( \{v_1, v_2, v_3\} \), we would recommend \( v_4 \), while on a different occasion when the user’s basket contains \( \{v_1, v_3, v_4\} \), we would recommend \( v_2 \). Because the recommendations go on to serve the same intent for the user, we postulate that their likelihoods should be similar.

Definition 1. TUPLES OF THE SAME INTENT We say that two tuples \( t_1 \) and \( t_2 \) in the training data \( T \) have the same intent if the following conditions hold:

- \( t_1 \) and \( t_2 \) concern the same user;
- the union of the basket items and the target item is identical between \( t_1 \) and \( t_2 \);
- both are positive examples, i.e., \( t_1.\delta = 1 \) and \( t_2.\delta = 1 \).
Given two tuples $t_1$ and $t_2$ of the same intent, we seek to minimize the difference between their function values.

$$F(h^{t_1}; \theta) - F(h^{t_2}; \theta)^2$$  \hspace{1cm} (9)

Different pairs of tuples with the same intent may have different degrees of correlation, which we model by the Pointwise Mutual Information (PMI) \cite[2009]{Bouma} of their target items. Suppose for two same-intent tuples $t_1$ and $t_2$, their target items are $v_1$ and $v_2$. The PMI is the joint probability of $v_1$ and $v_2$, estimated through their joint co-occurrence across transactions, divided by the marginal probabilities of $v_1$ and $v_2$ respectively, as shown in Equation 10. The higher the PMI the more likely two items appear in the same basket.

$$PMI(t_1; t_2) = \ln \frac{P(v_1, v_2)}{P(v_1)P(v_2)}$$  \hspace{1cm} (10)

In practice, there may be more than two tuples sharing the same intent. For a collection of such tuples from positive examples, the objective is to learn high scores. Imposing similarity across all pairs of such tuples may have the unintended effect of making them equally low, instead of equally high. Therefore, we would only add the constraint between a tuple $t$ and its same-intent tuple $t''$ that has the maximum score.

We now define the optimization criterion for the CONSTRAINED BFM or CBFM, as shown in Equation 11 below.

$$\text{OPT}_{\text{CBFM}}(T) = \arg\min_{\theta} \sum_{t \in T} \lambda \theta^2 + \sum_{t \in T} \{ -\ln(\sigma(F(h^{t_1}; \theta) \times t)) + \frac{\alpha}{2} \times PMI(t; t'') \times \left( F(h^{t_1}; \theta) - F(h^{t''}; \theta) \right)^2 \}$$  \hspace{1cm} (11)

The objective function of CBFM subsumes that of BFM. It still has the logistic loss function of BFM. It also features the constraint component. The key difference is the constraint component. The inference for CBFM’s logistic loss is similar to that of BFM. The objective is the constraint component. The overall gradient of parameters consists of the gradient due to the logistic loss, as well as the gradient due to the derivative of the constraint component. The latter is as follows.

$$\frac{\partial}{\partial \theta} (F(h^{t_1}; \theta) - F(h^{t''}; \theta))^2$$  \hspace{1cm} (12)

$$= 2(F(h^{t''}; \theta) - F(h^{t_1}; \theta)) \frac{\partial}{\partial \theta} (F(h^{t''}; \theta) - F(h^{t_1}; \theta))$$

Subsequently, we perform the following update iteratively in the SGD algorithm, where $\eta$ is the learning rate.

$$\theta \leftarrow \theta - \eta \left[ t \cdot \delta \times (\sigma(F(h^{t_1}; \theta) \times t \cdot \delta) - 1) \frac{\partial}{\partial \theta} F(h^{t_1}; \theta) + \alpha \times PMI(t; t'') \times (F(h^{t''}; \theta) - F(h^{t_1}; \theta)) \times \frac{\partial}{\partial \theta} (F(h^{t''}; \theta) - F(h^{t_1}; \theta)) + 2\lambda \theta \right]$$  \hspace{1cm} (13)

The complexity of CBFM learning is similar to BFM, $O(K \cdot |T| \cdot \bar{s})$, where $K$ is the vector dimensionality, $|T|$ is the size of the training data, and $\bar{s}$ is the average basket size in $T$. Compared to BFM, CBFM takes additional computations to find the maximum instance $t''$. The extra calculation is linearly proportional to the size of the respective basket.

### 5 Experiments

The objective of experiments is to investigate the effectiveness of the BFM and CBFM models.

#### 5.1 Setup

**Datasets.** We experiment with three public real-life datasets from different domains bearing basket-like associations. The dataset sizes are summarized in Table 1.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Users</th>
<th>#Items</th>
<th>#Transactions</th>
<th>Average #Items per Transaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>TaFeng</td>
<td>11711</td>
<td>11035</td>
<td>71447</td>
<td>7.3</td>
</tr>
<tr>
<td>BeiRen</td>
<td>9245</td>
<td>5581</td>
<td>87224</td>
<td>6.1</td>
</tr>
<tr>
<td>Foursquare</td>
<td>1548</td>
<td>3619</td>
<td>31377</td>
<td>2.7</td>
</tr>
</tbody>
</table>

Table 1: Statistics for TaFeng, BeiRen & Foursquare.

**TaFeng:** This is a retail market dataset. There are a series of transactions, where each transaction involves a user and multiple grocery items. The hypothesis is that items in a basket may be related as they go towards household needs.

**BeiRen:** This comes from a large retailer in China, capturing the period from 2012 to 2013. Similar to TaFeng, each transaction contains a set of items bought by a given user.

**Foursquare:** This consists of users’ check-ins at various points of interest in Singapore \cite{Yuan et al., 2013}. We treat the check-ins within the same day as a transaction. The hypothesis is that these check-ins involve related purposes. Previous works on point-of-interest \cite{Yuan et al., 2013} rely on modeling temporal or sequential associations. That is not the scope of our work, which is modeling in-basket associations.

Similar pre-processing is applied on all datasets. For sufficient statistics, we filter out items bought by too few users, i.e., 10 users for TaFeng & BeiRen and 5 users for Foursquare corresponding to their data sizes. As our focus is on modeling associations, we remove items that behave like “stop words” with presence in a large fraction of transactions (more than 5%). There are merely 2 or 3 such items in TaFeng and BeiRen respectively and none in Foursquare. As transactions with single items do not contain item-item association, we retain only transactions with more than 2 items. We also filter out users with fewer than 3 transactions, which is the minimum needed to have a training/validation/testing split.

**Training, Validation & Testing.** We further split the transactions as follows. For each user, we sort her transactions chronologically. The last transaction will be part of the testing set. The second-last transaction will be part of the validation set. The rest will be part of the training set.

For each transaction, we induce positive tuples in the form of $t = (u_i, B_i, v_j, 1)$. For each item $v_j$ in the transaction of user $u_i$, we hide $v_j$ as the item to be predicted. The remaining items observed in that transaction will form the basket $B_i$. Hence, a transaction containing $n$ items will result in $n$ positive tuples. In addition, as we discuss previously in Section 4, these $n$ tuples are said to have the same intent.

---

3\http://recsyswiki.com/wiki/Grocery_shopping_datasets
4\http://www.brjt.cn
5\http://www.ntu.edu.sg/home/gaocong/datacode.htm
Table 2: Performance Comparison for BFM with Various Association Types on TaFeng, BeiRen and Foursquare. The four association types include \( \gamma_1 \): User & Target, \( \gamma_2 \): Target & Basket, \( \gamma_3 \): Basket & Basket, \( \gamma_4 \): User & Basket.

<table>
<thead>
<tr>
<th>Association</th>
<th>TaFeng</th>
<th>BeiRen</th>
<th>Foursquare</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \gamma_1 ) ( \gamma_2 ) ( \gamma_3 ) ( \gamma_4 )</td>
<td>HLU R@10 (%)</td>
<td>HLU R@10 (%)</td>
<td>HLU R@10 (%)</td>
</tr>
<tr>
<td>1 0 0 0</td>
<td>0.06 0.10</td>
<td>1.94 3.16</td>
<td>5.45 8.29</td>
</tr>
<tr>
<td>1 1 0 0</td>
<td>1.47† 2.27†</td>
<td>3.35† 5.11†</td>
<td>8.11† 11.98†</td>
</tr>
<tr>
<td>1 1 1 0</td>
<td>2.14‡ 3.41‡</td>
<td>3.75‡ 5.77‡</td>
<td>8.51‡ 12.48‡</td>
</tr>
<tr>
<td>1 1 0 1</td>
<td>1.64† 2.59†</td>
<td>3.59† 5.54†</td>
<td>7.08† 10.50†</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>2.08† 3.31†</td>
<td>3.74† 5.78†</td>
<td>8.02† 11.84†</td>
</tr>
</tbody>
</table>

5.2 Results

BFM in Java based on libFM.\(^5\) For these experiments, we use latent factor dimension \( K = 8 \) and regularization parameter \( \lambda_0 = 0.01 \), which are also the defaults of libFM. The various numbers of latent factor dimensions \( K \) are empirically investigated in the last experiment. The initial learning rate \( \eta \) is 0.0001 for TaFeng, BeiRen and 0.001 Foursquare respectively to reflect their relative sparsity. We further apply the Bold-Driver adaptive learning rate [Battiti, 1989].

Table 2 shows BFM of different configurations. The first configuration \([\gamma_1, \gamma_2, \gamma_3, \gamma_4] = [1, 0, 0, 0]\) has only associations between each user and the target item. This is a factorization machine (FM) akin to matrix factorization, which does not feature any basket effects. The second configuration \([1, 1, 0, 0]\) adds associations between each basket item to the target item, with a higher performance than FM in terms of HLU and R@10, implying that basket items indeed have an influence on the target item. We further experiment with several more configurations. It emerges that the best configuration is \([1, 1, 1, 0]\), which includes associations among basket items, but excludes those between users and basket items.

Table 2 shows that models with basket associations are better than FM. The symbol † indicates that paired samples t-test shows statistical significance (at 0.05 level) in the improvements over FM. That the best configuration \([1, 1, 1, 0]\) shows statistically significant improvements over the second-best configuration \([1, 1, 1, 1]\) is indicated by the symbol ‡. Subsequently, we will use \([1, 1, 1, 0]\) as the default for BFM.

\(^5\)http://www.libfm.org
Effect of Constraint. We take the best configuration of BFM, and add the basket-level constraint to form CBFM. Figure 1(a) illustrates the CBFM’s HLU and R@10 on TaFeng when we vary α. BFM is equivalent to CBFM when α = 0. The performance generally rises and then falls. The best on TaFeng is α = 0.5. Figure 1(b) is for BeiRen, where the best configuration is α = 0.05. Finally, Figure 1(c) shows the corresponding results on Foursquare, with best performance at α = 1. Subsequently, we will use these α settings.

Comparison to Association Rules. We include a comparison to a baseline based on association rules [Sarwar et al., 2000]. First, we learn association rules from the training data, with minimum support of 10 on TaFeng, BeiRen and 5 on Foursquare (the same filters as for our training data). For a user and a basket, a rule is applicable if the antecedent items are contained in the basket, and have been adopted by the user previously. For each target item, if there are multiple applicable rules, we use the rule with maximum confidence. We then construct a ranked list in decreasing order of confidence.

Table 3 shows a comparison to the association rule-based baseline ASR. In addition to HLU and HLU@10, we also show R@20 and R@50. CBFM and BFM both outperform ASR across all measures. We hypothesize this is due to their use of factorization that allows them to discover other latent associations among items. The symbol † indicates the statistically significant (at 0.05 level) improvement of CBFM over BFM.

Model Complexity and Response Time. The goal of recommender systems is to provide users with relevant results in a timely and responsive manner [Koenigstein et al., 2012]. To retrieve the top-n recommendation at run time, we need to evaluate the prediction score for all possible items in the inventory [Bachrach et al., 2014]. One setting that has a direct effect on retrieval speed is the number of latent factors. We define response time as the time required to do the prediction computation for all items (required for top-n). Timing is based on a PC with Intel Core i5 3.2GHz with 8GB RAM.

Figure 2 demonstrates how HLU and response time are affected by different number of latent factors K. For CBFM, we tune the α based on the validation set for each K. We see a trend that increasing K leads to higher HLU but also higher response times. For TaFeng and BeiRen, Figures 2(a) and 2(b) show that CBFM has a slight gap over BFM throughout (statistically significant at 0.05 level). FM has relatively low performance, probably due to significant basket effects and data sparsity. Foursquare is an “easier” dataset, as shown by Figure 2(c). For very high number of latent factors (which also result in higher response times), eventually the models achieve similar performance. Importantly, CBFM shows a good trade-off behavior. For most response times, especially for the fast response times, it has significantly better HLU. The trends for HLU@10 are similar, and not shown here due to space constraint.

6 Conclusion

We investigate recommendation models that take into account of a user’s current basket in making personalized recommendations. We propose two models: BFM that incorporates various association types, and CBFM that further integrates constraints for baskets with similar intent. Experiments show some improvements over factorization machine that does not model basket associations, and association rules that do not benefit from latent associations discovered by factorization.

Acknowledgments

This research is supported by the National Research Foundation, Prime Minister’s Office, Singapore under its NRF Fellowship Programme (Award No. NRF-NRFF2016-07).
References


[Han et al., 2000] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In SIGMOD, pages 1–12, 2000.


