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[1] 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

[1]https://www.amazon.com/b?node=16008589011

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, suppose that a user shops for cake ingredients. At one occasion, the user may already have $\{milk, flour, sugar\}$ in her basket, and we may recommend *eggs*. At another occasion, the user may already have $\{milk, flour, eggs\}$ in her basket, and we thus recommend *sugar*. While 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.

Organization. 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 recommendation. 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 intent 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 $B_i \subset V$ 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. Θ denotes the parameters of the function.

3.1 Modeling Association Types

We develop our proposed BFM model by incorporating various types of useful associations.

User and Target Item. For personalized item recommendation, the basic type of association is between the user and the target item. This is a fundamental building block in most matrix factorization techniques [Koren *et al.*, 2009]. Building upon this foundation, we include in Θ, a latent vector $x_i \in \mathbb{R}^K$ in K dimensions for each user $u_i \in U$, as well as a latent vector $y_j \in \mathbb{R}^K$ for each target item $v_j \in V$. $F(u_i, B_i, v_j; \Theta)$ is assumed to be proportional to ${x_i}^T y_j$.

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

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 Θ 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.

$$F(u_i, B_i, v_j; \Theta) \propto \sum_{v_k \in B_i} {y_j}^T z_k \tag{2}$$

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.

$$F(u_i, B_i, v_j; \Theta) \propto \sum_{(v_k \neq v_{k'}) \in B_i} {z_k}^T z_{k'} \tag{3}$$

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

$$F(u_i, B_i, v_j; \Theta) \propto \sum_{v_k \in B_i} {x_i}^T z_k \tag{4}$$

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.

$$\begin{aligned} F(u_i, B_i, v_j; \Theta) \ &\propto \gamma_1 \cdot {x_i}^T y_j + \gamma_2 \cdot \sum_{v_k \in B_i} {y_j}^T z_k \\ &+ \gamma_3 \cdot \sum_{(v_k \neq v_{k'}) \in B_i} {z_k}^T z_{k'} + \gamma_4 \cdot \sum_{v_k \in B_i} {x_i}^T z_k \end{aligned} \tag{5}$$

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 Θ are learned, given a user u_i and a basket B_i, we construct a recommendation list of target items in the order of decreasing $F(u_i, B_i, v_j; \Theta)$.

3.2 Parameter Learning

We are given a set of tuples T, where each $t = \langle u_i, B_i, v_j, \delta \rangle \in T$ connotes 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 Θ, i.e., $\{x_i\}_{u_i \in U}$, and $\{y_j, z_j\}_{v_j \in V}$ 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 $\mathbf{h}$ be a vector of length p, with binary elements, i.e., $h_i \in \{0, 1\}$. A second-oder FM is as follows.

$$\mathcal{F}(\mathbf{h}) = \mu_0 + \sum_{i=1}^{p} \mu_i h_i + \sum_{i=1}^{p} \sum_{j=i+1}^{p} h_i h_j ({\phi_i}^T \phi_j) \tag{6}$$

The parameters include the global bias μ_0 and a bias coefficient μ_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 $t = \langle u_i, B_i, v_j, \delta \rangle \in T$, we construct a binary vector $\mathbf{h}^t$ of length p, where $p = N + 2M$. The first N terms in $\mathbf{h}^t$ are for the presence of a user. We have $h_i^t = 1$. The next M terms in $\mathbf{h}^t$ are for the target item. We have $h_{N+j}^t = 1$. The last M terms are for the basket items. For each basket item $v_k \in B_i$, we have $h_{N+M+k}^t = 1$. All other elements of $\mathbf{h}^t$ are zeros. The latent vectors of this factorization machine stand for those of BFM. ϕ_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 $F(u_i, B_i, v_j; \Theta)$ in Equation 5 is effectively transformed into Equation 7. Θ denotes the ϕ_i's that also stand for x, y, z's. The addition of biases μ_0 and μ_i's is appropriate, and it is a common practice in matrix factorization-based recommendation [Koren *et al.*, 2009].

$$\begin{aligned} \mathcal{F}(\mathbf{h}; \Theta) &= \mu_0 + \sum_{i=1}^{p} \mu_i h_i + \gamma_1 \sum_{i=1}^{N} \sum_{j=N+1}^{N+M} h_i h_j ({\phi_i}^T \phi_j) \\ &+ \gamma_2 \sum_{i=N+1}^{N+M} \sum_{j=N+M+1}^{p} h_i h_j ({\phi_i}^T \phi_j) \\ &+ \gamma_3 \sum_{i=N+M+1}^{p} \sum_{j=i+1}^{p} h_i h_j ({\phi_i}^T \phi_j) \\ &+ \gamma_4 \sum_{i=1}^{N} \sum_{j=N+M+1}^{p} h_i h_j ({\phi_i}^T \phi_j) \end{aligned} \tag{7}$$

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

$$\begin{aligned} &\text{OPT\_BFM}(T) \\ &= \text{argmin}_\Theta \left[\sum_{t \in T} - \ln(\sigma(\mathcal{F}(\mathbf{h}^t; \Theta) \times t.\delta)) + \sum_{\theta \in \Theta} \lambda_\theta \theta^2 \right] \end{aligned} \tag{8}$$

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

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 fulfil 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.

$$\left(\mathcal{F}(\mathbf{h}^{t_1};\Theta) - \mathcal{F}(\mathbf{h}^{t_2};\Theta)\right)^2 \tag{9}$$

Different pairs of tuples with the same intent may have different degrees of correlation, which we model by the Pointwise Mutual Information (PMI) [Bouma, 2009] 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.

$$PMI(t_1, t_2) = \ln \frac{P(v_1, v_2)}{P(v_1)(v_2)} \tag{10}$$

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^m that has the maximum score.

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

$$\begin{aligned} &\text{OPT\_CBFM}(T) \qquad (11)\\ &= \text{argmin}_\Theta \Bigg[\sum_{\theta\in\Theta}\lambda_\theta\theta^2 + \sum_{t\in T}\{-\ln(\sigma(\mathcal{F}(\mathbf{h}^t;\Theta)\times t.\delta)) \\ &+\frac{\alpha}{2}\times PMI(t,t^m)\times\left(\mathcal{F}(\mathbf{h}^t;\Theta)-\mathcal{F}(\mathbf{h}^{t^m};\Theta)\right)^2\}\Bigg]\end{aligned}$$

The objective function of CBFM subsumes that of BFM. It still has the logistic loss function of BFM. It also features the constraints. α is a coefficient controlling the strength of the constraint vis-á-vis the logistic loss, to be tuned empirically. As we are most concerned with strongly-correlated tuples, we apply the constraint only for positive tuples having positive PMI; otherwise it is considered zero with no effect.

The inference for CBFM's logistic loss is similar to that of BFM. The key difference 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 the constraint component. The latter is as follows.

$$\begin{aligned}&\frac{\partial}{\partial\theta}\mathcal{F}(\mathbf{h}^t;\Theta) - \mathcal{F}(\mathbf{h}^{t^m};\Theta))^2 \qquad (12)\\ &= 2(\mathcal{F}(\mathbf{h}^{t^m};\Theta)-\mathcal{F}(\mathbf{h}^t;\Theta))\frac{\partial}{\partial\theta}(\mathcal{F}(\mathbf{h}^{t^m};\Theta)-\mathcal{F}(\mathbf{h}^t;\Theta))\end{aligned}$$

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

$$\begin{aligned}\theta \leftarrow \theta - \eta \Bigg[&t.\delta\times(\sigma(\mathcal{F}(\mathbf{h}^t;\Theta)\times t.\delta)-1)\frac{\partial}{\partial\theta}\mathcal{F}(\mathbf{h}^t;\Theta)\\ &+\alpha\times PMI(t,t^m)\times(\mathcal{F}(\mathbf{h}^{t^m};\Theta)-\mathcal{F}(\mathbf{h}^t;\Theta))\\ &\times\frac{\partial}{\partial\theta}(\mathcal{F}(\mathbf{h}^{t^m};\Theta)-\mathcal{F}(\mathbf{h}^t;\Theta))+2\lambda_\theta\theta\Bigg] \qquad (13)\end{aligned}$$

The complexity of CBFM learning is similar to BFM, $\mathcal{O}(K\cdot|T|\cdot\bar{s})$, 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 comparisons to find the maximum instance t^m. The extra calculation is linearly proportional to the size of the respective basket.

Dataset	#Users	#Items	#Transactions	Average #Items per Transaction
TaFeng	11711	11035	71447	7.3
BeiRen	9245	5581	87224	6.1
Foursquare	1548	3619	31377	2.7

Table 1: Statistics for TaFeng, BeiRen & Foursquare.

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.

TaFeng[2]: 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[3]: 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[4]: This consists of users' check-ins at various points of interest in Singapore [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 [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 $t = \langle u_i, B_i, v_j, 1\rangle$. 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.

[2]http://recsyswiki.com/wiki/Grocery_shopping_datasets

[3]http://www.brjt.cn

[4]http://www.ntu.edu.sg/home/gaocong/datacode.htm

γ_1	γ_2	γ_3	γ_4	TaFeng HLU	TaFeng R@10 (%)	BeiRen HLU	BeiRen R@10 (%)	Foursquare HLU	Foursquare R@10 (%)
1	0	0	0	0.06	0.10	1.94	3.16	5.45	8.29
1	1	0	0	1.47†	2.27†	3.35†	5.11†	8.11†	11.98†
1	1	1	0	**2.14**†§	**3.41**†§	**3.75**†§	5.77†	**8.51**†§	**12.48**†§
1	1	0	1	1.64†	2.59†	3.59†	5.54†	7.08†	10.50†
1	1	1	1	2.08†	3.31†	3.74†	**5.78**†	8.02†	11.84†

Table 2: Performance Comparison for BFM with Various Association Types on TaFeng, BeiRen and Foursquare.The four association types include γ_1: User & Target, γ_2: Target & Basket, γ_3: Basket & Basket, γ_4: User & Basket.

As the datasets only have positive examples, following [Pan *et al.*, 2008], we create negative examples by sampling. From each positive tuple $t = \langle u_i, B_i, v_j, 1\rangle$, we can create a negative tuple $t^{\neg} = \langle u_i, B_i^{\neg}, v_j^{\neg}, -1\rangle$. As $v_j^{\neg}$, we randomly pick an item never selected by the user. $B_i^{\neg}$ contains items that never co-occur with either the user, $v_j^{\neg}$ or other items in $B_i^{\neg}$. For parity, we have $|B_i| = |B_i^{\neg}|$. As we expect there are more items that a user does not prefer than those that a user does, we have twice as many negative tuples. Training set has both positive and negative tuples for learning, while validation and testing sets consist of only positive tuples.

Evaluation Task & Metrics. The evaluation task is top-n recommendations. For each tuple $t = \langle u_i, B_i, v_j, 1\rangle$ in the testing set, we hide the observed item v_j, and require each model to produce a ranked list of items for u_i based on B_i. A list that ranks the observed v_j higher is better. For the proposed models, the performance numbers are averaged across 25 runs with different random initializations.

We rely on two evaluation metrics frequently used for top-n recommendation [Rendle *et al.*, 2010]. The first metric is Half-Life Utility or HLU [Breese *et al.*, 1998]. It measures how likely a user will adopt an item at a ranking position k. [Breese *et al.*, 1998] proposed this probability as $2^{\frac{1-k}{\beta-1}}$ with β as the half-life parameter. In our context, HLU is defined:

$$HLU = \frac{1}{|T_{\text{test}}|} \times C \times \sum_{t \in T_{\text{test}}} 2^{\frac{1-r_t}{\beta-1}}$$

where T_{test} is the testing set, and r_t is the rank of the item v_j in tuple t in the list. C is the scaling parameter. Following [Rendle *et al.*, 2010], we set $\beta = 5$ and $C = 100$.

The second metric is recall (*R@n*), defined as the percentage of testing instances with the ground truth item in top-n. The higher the percentage, the better the model is. In experiments, we primarily investigate top-10 recommendations, i.e., *R@10*, but will show performances for several other top-n as well. Precision may not be suitable, as the unobserved items may not necessarily be negative examples, but rather simply unlabeled positive examples [Wang and Blei, 2011].

5.2 Results

BFM **with Various Association Types.** First, we investigate several combinations of association types to determine what constitutes a good configuration for BFM. We implement BFM in Java based on libFM [5]. For these experiments, we use latent factor dimension $K = 8$ and regularization parameter $\lambda_\theta = 0.01$, 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 η 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].

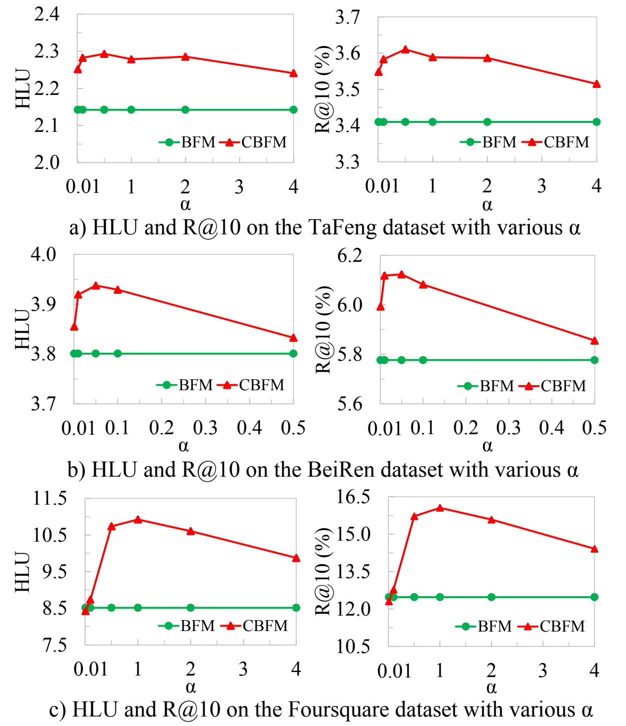

Figure 1: Performance Comparison for BFM and CBFM for Various α on TaFeng, BeiRen and Foursquare

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

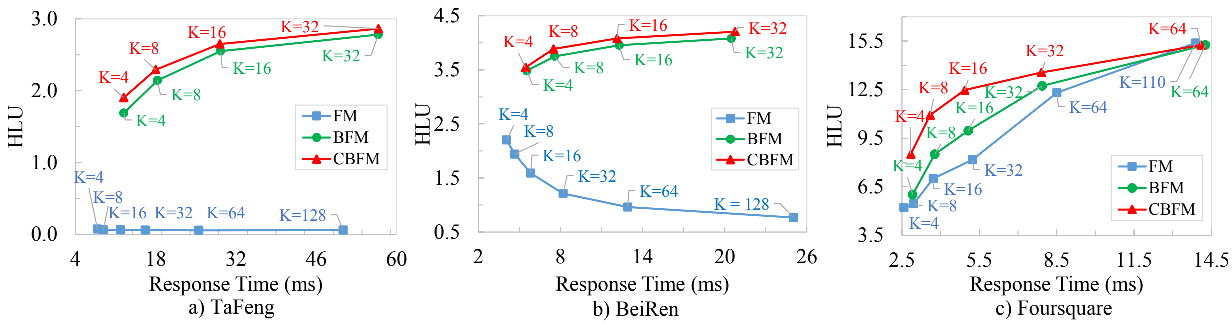

Figure 2: Half-life Utility and Response Times

Dataset	Model	HLU	R@n (%) 10	20	50
TaFeng	CBFM	**2.29**‡	**3.61**‡	**6.00**‡	**11.11**‡
	BFM	2.14	3.41	5.77	10.79
	ASR	1.97	2.72	3.56	4.83
BeiRen	CBFM	**3.89**‡	**6.12**‡	**10.42**‡	**19.04**‡
	BFM	3.75	5.78	9.91	18.79
	ASR	3.74	5.31	7.60	11.56
Foursquare	CBFM	**10.92**‡	**16.06**‡	**21.83**‡	**30.81**‡
	BFM	8.51	12.48	17.86	26.41
	ASR	6.54	10.36	12.56	15.64

Table 3: Performance Comparison to Association Rules (ASR) on TaFeng, BeiRen and Foursquare.

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 $\alpha = 0$. The performance generally rises and then falls. The best on TaFeng is $\alpha = 0.5$. Figure 1(b) is for BeiRen, where the best configuration is $\alpha = 0.05$. Finally, Figure 1(c) shows the corresponding results on Foursquare, with best performance at $\alpha = 1$. 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 $R@10$, 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 $R@10$ 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

[Agrawal *et al.*, 1994] Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. In *VLDB*, volume 1215, pages 487–499, 1994.

[Bachrach *et al.*, 2014] Yoram Bachrach, Yehuda Finkelstein, Ran Gilad-Bachrach, Liran Katzir, Noam Koenigstein, Nir Nice, and Ulrich Paquet. Speeding up the xbox recommender system using a euclidean transformation for inner-product spaces. In *RecSys*, pages 257–264, 2014.

[Battiti, 1989] Roberto Battiti. Accelerated backpropagation learning: Two optimization methods. *Complex Systems*, 3(4):331–342, 1989.

[Bouma, 2009] Gerlof Bouma. Normalized (pointwise) mutual information in collocation extraction. In *Biennial GSCL Conference*, volume 156, 2009.

[Breese *et al.*, 1998] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *UAI*, pages 43–52, 1998.

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

[Kim and Kim, 2003] Choonho Kim and Juntae Kim. A recommendation algorithm using multi-level association rules. In *IEEE/WIC*, pages 524–527, 2003.

[Koenigstein *et al.*, 2011] Noam Koenigstein, Gideon Dror, and Yehuda Koren. Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In *Recsys*, pages 165–172, 2011.

[Koenigstein *et al.*, 2012] Noam Koenigstein, Parikshit Ram, and Yuval Shavitt. Efficient retrieval of recommendations in a matrix factorization framework. In *CIKM*, pages 535–544, 2012.

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

[Le *et al.*, 2016] Duc-Trong Le, Yuan Fang, and Hady W Lauw. Modeling sequential preferences with dynamic user and context factors. In *ECML-PKDD*, pages 145–161, 2016.

[Li *et al.*, 2009] Ming Li, Benjamin M Dias, Ian Jarman, Wael El-Deredy, and Paulo JG Lisboa. Grocery shopping recommendations based on basket-sensitive random walk. In *SIGKDD*, pages 1215–1224, 2009.

[Liang *et al.*, 2016] Dawen Liang, Jaan Altosaar, Laurent Charlin, and David M. Blei. Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence. In *Recsys*, pages 59–66, 2016.

[Liu *et al.*, 2015] Qiang Liu, Shu Wu, and Liang Wang. Collaborative prediction for multi-entity interaction with hierarchical representation. In *CIKM*, pages 613–622, 2015.

[Pan *et al.*, 2008] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *ICDM*, pages 502–511, 2008.

[Pazzani and Billsus, 2007] Michael J Pazzani and Daniel Billsus. Content-based recommendation systems. In *The Adaptive Web*, pages 325–341. Springer, 2007.

[Pradel *et al.*, 2011] Bruno Pradel, Savaneary Sean, Julien Delporte, Sébastien Guérif, Céline Rouveirol, Nicolas Usunier, Françoise Fogelman-Soulié, and Frédéric Dufau-Joel. A case study in a recommender system based on purchase data. In *SIGKDD*, pages 377–385, 2011.

[Rendle *et al.*, 2010] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *WWW*, pages 811–820, 2010.

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

[Sarwar *et al.*, 2000] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Analysis of recommendation algorithms for e-commerce. In *ACM Conference on Electronic commerce*, pages 158–167, 2000.

[Shan *et al.*, 2012] H Shan, Jens Kattge, P Reich, A Banerjee, F Schrodt, and Markus Reichstein. Gap filling in the plant kingdom-trait prediction using hierarchical probabilistic matrix factorization. In *ICML*, 2012.

[Wang and Blei, 2011] Chong Wang and David M Blei. Collaborative topic modeling for recommending scientific articles. In *KDD*, pages 448–456, 2011.

[Wang *et al.*, 2014] Pengfei Wang, Jiafeng Guo, and Yanyan Lan. Modeling retail transaction data for personalized shopping recommendation. In *CIKM*, pages 1979–1982, 2014.

[Wang *et al.*, 2015] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. Learning hierarchical representation model for nextbasket recommendation. In *SIGIR*, pages 403–412, 2015.

[Xiang *et al.*, 2010] Liang Xiang, Quan Yuan, Shiwan Zhao, Li Chen, Xiatian Zhang, Qing Yang, and Jimeng Sun. Temporal recommendation on graphs via long-and short-term preference fusion. In *KDD*, pages 723–732, 2010.

[Yewatkar *et al.*, 2016] Ankush Yewatkar, Faiz Inamdar, Raj Singh, Amol Bandal, et al. Smart cart with automatic billing, product information, product recommendation using rfid & zigbee with anti-theft. *Elsevier Computer Science*, 79:793–800, 2016.

[Yu *et al.*, 2016] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. A dynamic recurrent model for next basket recommendation. In *SIGIR*, pages 729–732, 2016.

[Yuan *et al.*, 2013] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat Thalmann. Time-aware point-of-interest recommendation. In *SIGIR*, pages 363–372, 2013.

[Zhu *et al.*, 2014] Tao Zhu, Patrick Harrington, Junjun Li, and Lei Tang. Bundle recommendation in ecommerce. In *SIGIR*, pages 657–666, 2014.