Factorization Bandits for Interactive Recommendation

Huazheng Wang, Qingyun Wu, Hongning Wang
Department of Computer Science
University of Virginia, Charlottesville VA, 22904 USA
{hw7ww,qw2ky,hw5x}@virginia.edu

Abstract

We perform online interactive recommendation via a factorization-based bandit algorithm. Low-rank matrix completion is performed over an incrementally constructed user-item preference matrix, where an upper confidence bound based item selection strategy is developed to balance the exploit/explore trade-off during online learning. Observable contextual features and dependency among users (e.g., social influence) are leveraged to improve the algorithm's convergence rate and help conquer cold-start in recommendation. A high probability sublinear upper regret bound is proved for the developed algorithm, where considerable regret reduction is achieved on both user and item sides. Extensive experimentations on both simulations and large-scale real-world datasets confirmed the advantages of the proposed algorithm compared with several state-of-the-art factorization-based and bandit-based collaborative filtering methods.

Introduction

Matrix factorization based collaborative filtering has become a standard practice in recommender systems (Koren, Bell, and Volinsky 2009; Koren 2008; Su and Khoshgoftaar 2009). The basic idea of such solutions is to characterize both recommendation items and users by vectors of latent factors inferred from *historical* user-item preference patterns via low-rank matrix completion (Candès and Recht 2009; Candès and Tao 2010), with an assumption that only a few factors contribute to an individual's choice (Koren, Bell, and Volinsky 2009).

Despite a few recent advances in specific factorization techniques (Agarwal and Chen 2009; Rendle 2012), recommendation remains a challenging problem for at least two reasons. First, a modern recommender system faces emerging new users and ever changing pools of recommendation candidates. The classical offline training and online testing paradigm for factorization models becomes incompetent to handle the dynamics of users' preferences, known as *cold-start* (Schein et al. 2002). Second, it is nutritiously difficult to perform online interactive recommendation, because the need to focus on items that raise users' interest and, simultaneously, the need to explore new items for improving users' satisfaction in the long run create an explore-exploit dilemma. Periodically repeating model estimation to update latent factors is inept to handle the interactions between a system and its users on the fly, because not only does it overly exploit the learnt model that is biased towards previously frequently recommended items, but it is also prohibitively expensive to afford in terms of computational complexity.

Some preliminary attempts have been made to perform online matrix factorization for collaborative filtering. Basically, multi-armed bandit algorithms (Auer et al. 1995; Auer 2002) are employed to control the exploration of currently less promising recommendations for user feedback, and factorization is applied over the incrementally constructed user-item matrix on the fly. However, these two components are integrated in an *ad-hoc* manner: both contextual and context-free bandits have been explored on top of various factorization methods (Zhao, Zhang, and Wang 2013; Nakamura 2014; Kawale et al. 2015), given they only provide an index of candidate items for feedback acquisition. As a result, little is known about whether such combinations would lead to a converging recommendation performance nor would it ensure long-term optimality in theory, i.e., regret bound analysis.

We address the aforementioned challenges via performing online interactive recommendation by placing a factorization-based bandit algorithm on each user in the system. Low-rank matrix completion is performed over an incrementally constructed user-item preference matrix, where an upper confidence bound (UCB) based item selection strategy is developed to balance the exploit/explore trade-off during online learning. To better conquer cold-start in recommendation, two special treatments are devised. First, observable contextual features are integrated with the estimated latent factors during matrix factorization. This improves recommendation when the number of candidate items is large, but the payoffs are interrelated, i.e., *context-aware*. Second, the dependence among users (e.g., social influence) is introduced to our bandit algorithm through a collaborative reward generation assumption (Wu et al. 2016). It enables information sharing among the neighboring users in online learning, so as to help reduce the overall regret.

More importantly, we rigorously prove that with high probability the developed algorithm achieves a sublinear upper regret bound for interactive recommendation, i.e., the average number of suboptimal recommendations made in our algorithm over time rapidly vanishes with high probability. And considerable regret reduction is achieved on both user and item sides because of our explicit modeling of observable contextual features and dependence among users. Extensive experimentations on both simulations and large-scale real-world datasets confirmed the advantages of the proposed algorithm compared with several state-of-the-art bandit-based factorization methods.

Related work

There are some recent developments that focus on online collaborative filtering with multi-armed bandit algorithms, a reference solution for explore-exploit trade-off (Auer et al. 1995; Auer 2002; Li et al. 2010). (Zhao, Zhang, and Wang 2013) studies interactive collaborative filtering via probabilistic matrix factorization. Both context-free and contextual bandit algorithms are introduced to perform online item selection based on the factorization results. (Kawale et al. 2015) performs online low-rank matrix completion, where the explore/exploit balance is achieved via Thompson sampling. (Nakamura 2014) introduces a UCB-like strategy to perform interactive collaborative filtering. The algorithm deterministically selects feedback user-item pairs using an index which depends on the covariance matrices of the posterior distributions of both latent user and item vectors. (Li, Karatzoglou, and Gentile 2016) performs co-clustering on users and items for collaborative filtering, where confidence bound on reward estimation is used to decide the clustering structures. However, because of the ad-hoc combinations of collaborative filtering methods and bandit methods in the aforementioned studies, limited theoretical understanding is available in those solutions. In this work, we provide a rigorous regret bound analysis of the developed factorization-based bandit algorithm, and demonstrate the algorithm's convergence property under different conditions. Moreover, our online factorization solution is general enough to incorporate several recent advances in factorization techniques, such as feature-based latent factor models (Agarwal and Chen 2009; Rendle 2012) and modeling mutual dependence among users (Ma et al. 2011; 2008), which further improve the proposed algorithm's convergence rate during interactive online learning with users.

Methodology

A Bandit Solution for Interactive Recommendation

Matrix factorization based collaborative filtering solutions map both users $U = \{u_1, u_2, ..., u_N\}$ and recommendation items $\mathcal{A} = \{a_1, a_2, ..., a_M\}$ to a joint latent factor space. The expected reward of an item with respect to a given user is assumed to be an inner product of the latent item factor $\mathbf{v}_a \in \mathbb{R}^l$ and the latent user factor $\boldsymbol{\theta}_u \in \mathbb{R}^l$. Hence, the reward generation process can be formalized as $r_{a,u} = \mathbf{v}_a^\mathsf{T}\boldsymbol{\theta}_u + \eta$, where the random variable η is drawn from a Gaussian distribution $N(0, \sigma^2)$ to capture the noise in observations. Regularized quadratic loss over a given set of user-item feedback pairs is usually employed to estimate the latent factors. Formally,

$$\min_{\boldsymbol{\theta}_u, \mathbf{v}_a} \frac{1}{2} \sum_{(a,u)\in\mathcal{K}} (\mathbf{v}_a^\mathsf{T}\boldsymbol{\theta}_u - r_{a,u})^2 + \frac{\lambda_1}{2} \sum_{u\in U} \|\boldsymbol{\theta}_u\|_2 + \frac{\lambda_2}{2} \sum_{a\in\mathcal{A}} \|\mathbf{v}_a\|_2 \tag{1}$$

where $\mathcal{K}$ is a set of user-item pairs with known reward (e.g., the offline training set), λ_1 and λ_2 are the trade-off parameters. The key research challenge in interactive matrix factorization is how to select the next feedback user-item pair for model update. Current practice exploits the trained model to collect user feedback, which unfortunately reinforces the bias in a currently inaccurate model. Therefore, properly explore some currently less promising items for model correction becomes necessary for long-term optimality.

Upper Confidence Bound (UCB) has been proved to be an effective strategy to balance exploitation and exploration in multi-armed bandit problems. A UCB-style bandit algorithm uses its estimation confidence of the predicted reward on the candidate items for exploration: the item with the highest expected reward within a selected confidence set will be chosen for feedback (Auer 2002; Auer, Cesa-Bianchi, and Fischer 2002). Under the context of matrix factorization based collaborative filtering, the uncertainty of reward prediction comes from two sources: 1) the estimation error of latent user factors at trial t, i.e., $\|\hat{\boldsymbol{\theta}}_{u,t} - \boldsymbol{\theta}_u^*\|$, where $\hat{\boldsymbol{\theta}}_{u,t}$ is the current estimate of latent factors for user u, and $\boldsymbol{\theta}_u^*$ is the ground-truth factors; and 2) the estimation error of latent item factors at trial t, i.e., $\|\hat{\mathbf{v}}_{a,t} - \mathbf{v}_a^*\|$. Because of the regularized quadratic loss employed in Eq (1), the confidence sets of $\boldsymbol{\theta}_u$ and $\mathbf{v}_a$ estimation can be analytically computed (Abbasi-yadkori, Pál, and Szepesvári 2011), and thus readily be integrated to assemble a UCB-style bandit algorithm for interactive matrix factorization as follows,

$$\big(a_t, \langle\hat{\boldsymbol{\theta}}_{u,t}, \hat{\mathbf{v}}_{a,t}\rangle\big) = \underset{\big(a, \langle\boldsymbol{\theta}_u, \mathbf{v}_a\rangle\big)\in\mathcal{D}_t\times\mathcal{C}_{t-1}}{\arg\max} \boldsymbol{\theta}_u^\mathsf{T}\mathbf{v}_a \tag{2}$$

where $\mathcal{D}_t$ is the set of candidate items for recommendation at trial t, and $\mathcal{C}_{t-1}$ is the confidence set for latent user and item factors $\langle\boldsymbol{\theta}_u, \mathbf{v}_a\rangle$ constructed at last trial.

However, such a straightforward combination of bandit algorithm with matrix factorization cannot effectively solve the cold-start problem, as the estimation uncertainty of the latent factors for new users and new items is at the maximum. This inevitably requires more explorations on the new users and new items, and hence leads to a decreased convergence rate of online learning and reduced user satisfaction in practice. We propose to solve these limitations by introducing observed contextual features (Agarwal and Chen 2009; Rendle et al. 2011) and user dependence (Ma et al. 2011; Wu et al. 2016) into online factorization. Both of these two techniques have been proved to be effective in offline matrix factorization, but little is known about their utility in an online setting. In particular, we explicitly incorporate these two components into our bandit algorithm's reward generation assumption, to make it a unified framework for interactive matrix factorization.

First, to reduce the reward prediction uncertainty on new items, we introduce observable contextual features into the estimation of latent item factors. Typical item-level contextual features include topic categories for news recommendation (Li et al. 2010; Agarwal and Chen 2009) and genre for music recommendation (Cesa-Bianchi, Gentile, and Zappella 2013). Formally, we denote the observed contextual features for an item a as $\mathbf{x}_a \in \mathbb{R}^d$ and keep using $\mathbf{v}_a \in \mathbb{R}^l$ for its latent part (with $\|(\mathbf{x}_a, \mathbf{v}_a)\|_2 \leq L$). Accordingly, on the user side we redefine $\boldsymbol{\theta}_u = (\boldsymbol{\theta}_u^{\mathbf{x}}, \boldsymbol{\theta}_u^{\mathbf{v}}) \in \mathbb{R}^{d+l}$ (with

$\|\boldsymbol{\theta}_u\|_2 \leq S$), in which $\boldsymbol{\theta}_u^{\mathbf{x}} \in \mathbb{R}^d$ corresponds to the context feature $\mathbf{x}_a$ and $\boldsymbol{\theta}_u^{\mathbf{v}} \in \mathbb{R}^l$ corresponds to the latent item factor $\mathbf{v}_a$. These extended user and item factors now determine the rewards in recommendation.

Second, we incorporate mutual influence among users to reduce the reward prediction uncertainty on new users. Distinct from existing solutions, where the dependency among users (such as social network) is introduced as graph-based regularization over the latent user factors (Ma et al. 2011; Cesa-Bianchi, Gentile, and Zappella 2013), we encode such dependency directly into our reward generation assumption for matrix factorization. We assume the observed reward from each user is determined by a mixture of neighboring users (Wu et al. 2016). Formally, instead of assuming N independent users for factorization, we place them on a weighted graph $G = (V, E)$, which encodes the affinity relation among users, to perform the estimation across them simultaneously. Each node V_u in G is parameterized by the latent user factor $\boldsymbol{\theta}_u$ for user u; and each edge in E represents the influence across users in reward generation. We encode this graph as an $N \times N$ stochastic matrix $\mathbf{W}$, in which each element w_{ij} is nonnegative and proportional to the influence that user j has on user i in determining the reward of different items. $\mathbf{W}$ is column-wise normalized such that $\sum_{j=1}^{N} w_{ij} = 1$ for $i \in \{1, ..., N\}$, and we assume $\mathbf{W}$ is time-invariant and known to the algorithm beforehand.

Based on the introduced contextual features and user relational graph G, we define a $(d + l) \times N$ matrix $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N)$, which consists of latent user factors from all N users in graph G, and define $\mathbf{X}_{a_t} = (\mathbf{x}_{a_t,1}, ..., \mathbf{x}_{a_t,N})$ and $\mathbf{V}_{a_t} = (\mathbf{v}_{a_t,1}, ..., \mathbf{v}_{a_t,N})$ for the observable contextual features and latent item factors of the items to be presented to the N users respectively. To simplify the notations for discussion, we decompose $\boldsymbol{\Theta}$ into two sub-matrices, $\boldsymbol{\Theta}^{\mathbf{x}} = (\boldsymbol{\theta}_1^{\mathbf{x}}, \ldots, \boldsymbol{\theta}_N^{\mathbf{x}})$ and $\boldsymbol{\Theta}^{\mathbf{v}} = (\boldsymbol{\theta}_1^{\mathbf{v}}, \ldots, \boldsymbol{\theta}_N^{\mathbf{v}})$, corresponding to the observed context features and latent factors for items. As a result, we enhance our reward generation assumption as follows,

$$r_{a_t,u} = (\mathbf{x}_{a_t}, \mathbf{v}_{a_t})^{\mathsf{T}} \boldsymbol{\Theta} \mathbf{w}_u + \eta_t = \mathbf{x}_{a_t}^{\mathsf{T}} \boldsymbol{\Theta}^{\mathbf{x}} \mathbf{w}_u + \mathbf{v}_{a_t}^{\mathsf{T}} \boldsymbol{\Theta}^{\mathbf{v}} \mathbf{w}_u + \eta_t \tag{3}$$

Intuitively, in Eq (3) not only the observed contextual features, but also the estimated latent factors will be propagated through the user graph to determine the expected reward of items across users. Later we prove such information sharing greatly reduces sample complexity in learning the latent factors for both users and items.

Plugging the enhanced reward generation assumption defined in Eq (3) into the regularized quadratic loss function in Eq (1), we can easily derive the closed-form solutions for $\boldsymbol{\Theta}$ and $\mathbf{v}_a$ *after* trial t via the alternating least square (ALS) method as $vec(\hat{\boldsymbol{\Theta}}_t) = \mathbf{A}_t^{-1}\mathbf{b}_t$ and $\hat{\mathbf{v}}_{a,t} = \mathbf{C}_{a,t}^{-1}\mathbf{d}_{a,t}$, where the detailed computation of $(\mathbf{A}_t, \mathbf{b}_t, \mathbf{C}_{a,t}, \mathbf{d}_{a,t})$ can be found in Algorithm 1. $vec(\cdot)$ is the vectorization operation, and $\mathbf{I}_1$ and $\mathbf{I}_2$ are identity matrices with dimensions of $(d + l)N \times (d + l)N$ and $l \times l$ respectively. We define $\mathring{\mathbf{X}}_{a_t}$ as a special case of $\mathbf{X}_{a_t}$: only the column corresponding to user u is set to $\mathbf{x}_{a_t,u}$ and all the other columns are zero; and the same notation applies to $\mathring{\mathbf{V}}_{a_t}$.

One potential issue with this closed-form solution is its computational complexity: matrix inverse has to be performed on $\mathbf{A}_t$ and $\mathbf{C}_{a,t}$ at every trial. First, because rank one update is performed on these two matrices (11th and 14th step in Algorithm 1), quadratic computational complexity is possible via applying the Sherman-Morrison formula. Second, we can update these two matrices in a mini-batch manner to further reduce computation, with some extra penalty in regret. We will leave this as our future research.

Under our enhanced reward generation assumption defined in Eq (3), the confidence set of $\langle \boldsymbol{\theta}_u, \mathbf{v}_a \rangle$ estimation can be analytically computed by the following lemma.

Lemma 1 *With proper initialization of ALS, the Hessian matrix of Eq* (1) *is positive definite at the optimizer* $\boldsymbol{\Theta}^*$ *and* $\mathbf{v}_a^*$*, such that for any* $\epsilon_1 > 0$*,* $\epsilon_2 > 0$*, and* $\delta \in (0, 1)$*, with probability at least* $1 - \delta$*, the estimation error of latent user and item factors satisfies,*

$$\|vec(\hat{\boldsymbol{\Theta}}_t) - vec(\boldsymbol{\Theta}^*)\|_{\mathbf{A}_t} \leq \sqrt{\ln\big(\frac{det(\mathbf{A}_t)}{\delta\lambda_1}\big)} + \sqrt{\lambda_1}S \tag{4}$$
$$+ \frac{2}{\sqrt{\lambda_1}} \frac{(q_1 + \epsilon_1)(1 - (q_1 + \epsilon_1)^t)}{1 - (q_1 + \epsilon_1)}$$
$$\|\hat{\mathbf{v}}_{a,t} - \mathbf{v}_a^*\|_{\mathbf{C}_{a,t}} \leq \sqrt{\ln\big(\frac{det(\mathbf{C}_{a,t})}{\delta\lambda_2}\big)} + \sqrt{\lambda_2}L \tag{5}$$
$$+ \frac{2}{\sqrt{\lambda_2}} \frac{(q_2 + \epsilon_2)(1 - (q_2 + \epsilon_2)^t)}{1 - (q_2 + \epsilon_2)}$$

in which $q_1 \in (0, 1)$ *and* $q_2 \in (0, 1)$.

In Lemma 1, ϵ_1 and ϵ_2 are the precision parameters for ALS, and q_1 and q_2 can be explicitly estimated as described in (Uschmajew 2012). The key assumption behind this lemma is the noise distribution in reward generation defined in Eq (3) is *stationary*. As a result, this lemma gives us a reasonable construction of the confidence sets for $\boldsymbol{\Theta}$ and $\mathbf{v}_a$ estimation, which can be easily transformed to the estimation uncertainty of payoff $r_{a_t,u}$. The proof sketch of this lemma can be found in the supplementary material.

Based on Lemma 1, we define α_t^u and α_t^a as the upper bound of $\|vec(\hat{\boldsymbol{\Theta}}_t) - vec(\boldsymbol{\Theta}^*)\|_{\mathbf{A}_t}$ and $\|\hat{\mathbf{v}}_{a,t} - \mathbf{v}_a^*\|_{\mathbf{C}_{a,t}}$ respectively. By applying the UCB principle, the item selection strategy for our bandit algorithm can be derived as step 9 in Algorithm 1. In particular, the first term in our item selection strategy is an online prediction of the expected reward based on the current estimation of latent user factors and item factors. It reflects the tendency of exploiting the current estimates. The second and third terms are related to the estimation uncertainty of $\mathbf{v}_a$ and $\boldsymbol{\Theta}$. They reflect the tendency of exploring currently less promising but highly uncertain items. It is easy to verify that the exploration terms shrink when more observations become available, such that the exploit/explore trade-off is balanced dynamically. Later on we prove that because of the explicit modeling of user dependency (i.e., Eq (3)), the exploration term also uniformly shrinks for new users and new items, which lead to considerable regret reduction over all users. We name the resulting bandit algorithm as FactorUCB, and illustrate the detailed procedure of it in Algorithm 1.

Regret Analysis

To quantify the performance of factorUCB, we consider the cumulated (pseudo) regret defined as the expected difference

Algorithm 1 FactorUCB

```
1: Inputs: $\lambda_1, \lambda_2 \in (0, +\infty), l \in \mathbb{Z}^+$
2: Initialize: $\mathbf{A}_1 \leftarrow \lambda_1 \mathbf{I}_1$, $\mathbf{b}_1 \leftarrow \mathbf{0}^{(d+l)N}$, $vec(\hat{\mathbf{\Theta}}_1) \leftarrow \mathbf{A}_1^{-1}\mathbf{b}_1$
3: for $t = 1$ to $T$ do
4:     Receive user $u_t$
5:     Observe feature vectors, $\mathbf{x}_a \in \mathbb{R}^d$
6:     if item $a$ is new then
7:         initialize $\mathbf{C}_{a,t} \leftarrow \lambda_2 \mathbf{I}_2$, $\mathbf{d}_{a,t} \leftarrow \mathbf{0}^l$, $\hat{\mathbf{v}}_{a,t} \leftarrow \mathbf{0}^l$
8:     end if
9:     Select item by $a_t = \arg\max_{a \in \mathcal{A}} \Big( (\mathbf{x}_a, \hat{\mathbf{v}}_{a,t})^\mathsf{T} \hat{\mathbf{\Theta}}_t \mathbf{w}_{u_t} + \alpha_t^u \sqrt{vec\big((\mathring{\mathbf{X}}_{a_t}, \mathring{\hat{\mathbf{V}}}_{a_t})\mathbf{W}^\mathsf{T}\big) \mathbf{A}_t^{-1} vec\big((\mathring{\mathbf{X}}_{a_t}, \mathring{\hat{\mathbf{V}}}_{a_t})\mathbf{W}^\mathsf{T}\big)^\mathsf{T}} \Big) + \alpha_t^a \sqrt{(\hat{\mathbf{\Theta}}_t \mathbf{w}_{u_t}) \mathbf{C}_{a,t}^{-1} (\hat{\mathbf{\Theta}}_t \mathbf{w}_{u_t})^\mathsf{T}}$
10:    Observe reward $r_{a_t,u_t}$ from user $u_t$
11:    $\mathbf{A}_{t+1} \leftarrow \mathbf{A}_t + vec((\mathring{\mathbf{X}}_{a_t}, \mathring{\hat{\mathbf{V}}}_{a_t})\mathbf{W}^\mathsf{T}) vec((\mathring{\mathbf{X}}_{a_t}, \mathring{\hat{\mathbf{V}}}_{a_t})\mathbf{W}^\mathsf{T})^\mathsf{T}$
12:    $\mathbf{b}_{t+1} \leftarrow \mathbf{b}_t + vec((\mathring{\mathbf{X}}_{a_t}, \mathring{\hat{\mathbf{V}}}_{a_t})\mathbf{W}^\mathsf{T}) r_{a_t,u_t}$
13:    $vec(\hat{\mathbf{\Theta}}_{t+1}) \leftarrow \mathbf{A}_{t+1}^{-1} \mathbf{b}_{t+1}$
14:    $\mathbf{C}_{a_t,t+1} \leftarrow \mathbf{C}_{a_t,t} + (\hat{\mathbf{\Theta}}_t^\mathbf{v} \mathbf{w}_{u_t})(\hat{\mathbf{\Theta}}_t^\mathbf{v} \mathbf{w}_{u_t})^\mathsf{T}$
15:    $\mathbf{d}_{a_t,t+1} \leftarrow \mathbf{d}_{a_t,t} + (\hat{\mathbf{\Theta}}_t^\mathbf{v} \mathbf{w}_{u_t})(r_{a_t,u_t} - \mathbf{x}_{a_t}^\mathsf{T} (\hat{\mathbf{\Theta}}_t^\mathbf{x} \mathbf{w}_{u_t}))$
16:    $\hat{\mathbf{v}}_{a_t,t+1} \leftarrow \mathbf{C}_{a_t,t+1}^{-1} \mathbf{d}_{a_t,t+1}$
17:    Project $\hat{\mathbf{\Theta}}_{t+1}$ and $\hat{\mathbf{v}}_{a_t,t+1}$ with respect to the constraints $\|\boldsymbol{\theta}_u\|_2 \leq S$ and $\|(\mathbf{x}_a, \mathbf{v}_a)\|_2 \leq L$
18: end for
```

between the optimal reward obtained by the oracle item selection strategy and the reward received following the algorithm over T trials,

$$\mathbf{R}(T) = \sum_{t=1}^{T} R_t = \sum_{t=1}^{T} (r_{a_t^*,u_t} - r_{a_t,u_t}) \tag{6}$$

in which a_t^* is the best item to be presented to the current user u_t according to the oracle and a_t is the item selected by the algorithm, and R_t is the one-step regret at trial t.

Based on Lemma 1 and the developed item selection strategy, we have the following theorem about the upper regret bound of our FactorUCB algorithm.

Theorem 1 *Under proper initialization of ALS in Algorithm 1, with probability at least $1 - \delta$, the cumulated regret of FactorUCB algorithm satisfies,*

$$\begin{aligned} \boldsymbol{R}(T) \leq & 2\alpha_T^u \sqrt{2(d+l)NT \ln\Big(1 + \frac{L^2 \sum_{t=1}^{T} \sum_{j}^{N} w_{u_t,j}^2}{\delta \lambda_1 (d+l) N}\Big)} \\ & + 2\alpha_T^a \sqrt{2lT \ln\Big(1 + \frac{S^2 \sum_{t=1}^{T} \sum_{j}^{N} w_{u_t,j}^2}{\delta \lambda_2 l}\Big)} \\ & + 2\alpha_T^a \frac{(q_2 + \epsilon_2)\big(1 - (q_2 + \epsilon_2)^T\big)}{1 - (q_2 + \epsilon_2)} \end{aligned} \tag{7}$$

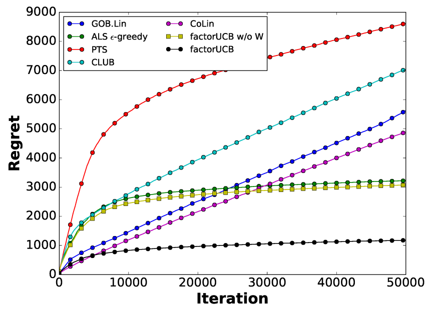
in which q_2 and ϵ_2 are the same as those defined in Lemma 1, α_T^u and α_T^a are the upper bound of $\|vec(\hat{\mathbf{\Theta}}_t) - vec(\mathbf{\Theta}^*)\|_{\mathbf{A}_t}$ and $\|\hat{\mathbf{v}}_{a,t} - \mathbf{v}_a^*\|_{\mathbf{C}_{a,t}}$ over all $t \in \{1, \ldots, T\}$ respectively, and δ is also encoded in α_T^u and α_T^a as shown in Eq (4) and (5). Though required by the theorem that λ_1 and λ_2 have to be sufficiently large, in our empirical evaluations the algorithm's performance is not sensitive to this setting. The specific form of α_T^u and α_T^a and the proof sketch of this theorem are provided in the supplementary material. As highlighted in the proof, because the confidence interval is shrinking via exploration, a sublinear regret is achieved after T trials of interactions; otherwise without proper exploration, such as in the conventional offline training and online testing paradigm of matrix factorization, a linear regret is inevitable. Some other ad-hoc exploration strategies have been proposed in literature (Zhao, Zhang, and Wang 2013; Nakamura 2014), but little is known about their regret bound, or analysis is only provided for overly simplified situations (e.g., the user-item matrix is one rank one and all the latent factors can be properly discretized beforehand (Kawale et al. 2015)). For an online learning algorithm, a sublinear upper regret bound is vital, as it indicates the average number of suboptimal recommendations a system makes vanishes rapidly over time (a linear regret bound means the algorithm makes constant errors).

Moreover, the resulting regret bound of factorUCB has the following important theoretical properties under different conditions.

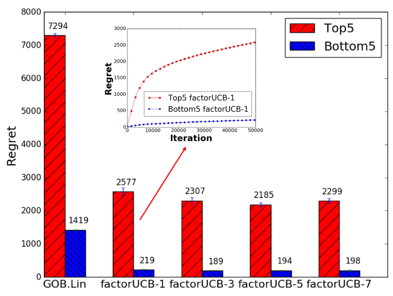
First, the dependency structure among users plays an important role in reducing the regret on both user side and item side. Consider the following two extreme cases. In the first case, when $\mathbf{W}$ is an identity matrix, i.e., no dependency among users, the first term of the upper regret bound in Eq (7) degenerates to $O\big(N(d+l)\sqrt{T} \ln \frac{T}{N}\big)$, which roots in the reward prediction uncertainty from the estimated latent user factors. And the second term degenerates to $O\big(l\sqrt{T} \ln T\big)$, which corresponds to the estimated latent item factors. In the second case, when users are homogenous and have uniform influence among each other, i.e., $\forall i, j, w_{ij} = \frac{1}{N}$, the first term in the regret bound decreases to $O\big(N(d+l)\sqrt{T} \ln \frac{T}{N^2}\big)$ and the second decreases to $O\big(l\sqrt{T} \ln \frac{T}{N}\big)$. As a result, via modeling user dependency, FactorUCB achieves an $O\big(N(d+l)\sqrt{T} \ln N\big)$ regret reduction on the user side and an $O\big(l\sqrt{T} \ln N\big)$ regret reduction on the item side. The best known upper regret bound for a linear bandit algorithm is $O\big(\sqrt{Td} \log(Td)\big)$ (Abbasi-yadkori, Pál, and Szepesvári 2011), which increases linearly with respect to the number of users in a recommender system. In factorUCB, our worst case upper regret bound matches that known bound, but its average regret per user is decreasing as more users interact with the system to provide feedback. The same analogy also applies to the number of recommendation candidates. This is an advantageous property for a practical system to provide satisfactory recommendations rapidly in an online setting.

Second, as denoted in Eq (7), the user arrival sequence is recorded in the summation term of $\sum_{t=1}^{T} \sum_{j=1}^{N} w_{u_t,j}^2$, which is bounded by T from above, no matter how users arrive to the system (as $\mathbf{w}_u$ is a stochastic vector). Therefore, the upper regret bound of factorUCB stays in $O\big(N(d+l)\sqrt{T} \ln \frac{T}{N}\big)$ in the worse case scenario, such as users arrive in an adversarial way – the least connected users come first and most often.

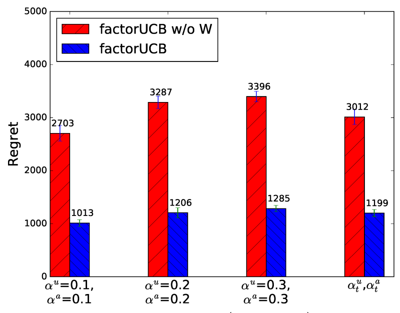
Third, following our enhanced reward generation assumption specified in Eq (3), the estimation quality of latent user factors in factorUCB satisfies the following inequality (similar result applies to the estimation quality of latent item fac-



(a) Cumulated regret



(b) Effect of latent dimensions



(c) Effect of (α_t^u, α_t^a)

Figure 1: Analysis of regret, hidden feature dimension and parameter tuning.

tors as well),

$$\|vec(\hat{\mathbf{\Theta}}_t) - vec(\mathbf{\Theta}^*)\|_{\mathbf{A}_t} \leq \sqrt{\ln\big(\frac{det(\mathbf{A}_t)}{\delta\lambda_1}\big)} + \sqrt{\lambda_1}S \qquad (8)$$
$$+ \frac{2}{\sqrt{\delta\lambda_1}}\sum_{t'=1}^{t}\|\mathbf{v}^*_{a_{t'},u} - \hat{\mathbf{v}}_{a_{t'},u}\|_2$$

If the dimension of latent factors matches the ground-truth, based on the proved convergence property of ALS in (Uschmajew 2012), the estimation of $\mathbf{\Theta}$ and $\mathbf{v}_a$ is q-linearly convergent to the optimum $(\mathbf{\Theta}^*, \mathbf{v}_a^*)$, which is the conclusion in Lemma 1. But if the dimension is not correctly set and those latent factors are independent from each other, the third term in Eq (8) will not converge. It makes α_t^u linearly increase over time as α_t^u is the upper bound of $\|vec(\hat{\mathbf{\Theta}}_t) - vec(\mathbf{\Theta}^*)\|_{\mathbf{A}_t}$. This leads to a linear regret in factorUCB at the worst case. Admittedly, determining the correct dimension of latent factors is always a bottleneck of factorization-based methods in practice. But by introducing the observable contextual features, especially those strongly correlated with the expected rewards, the reward prediction uncertainty can be reduced as the latent factors only need to fit the residual of reward prediction from the observed features (as shown in the estimation of $\mathbf{v}_a$ in Algorithm 1). This leads to reasonable performance of factorUCB in our empirical evaluations.

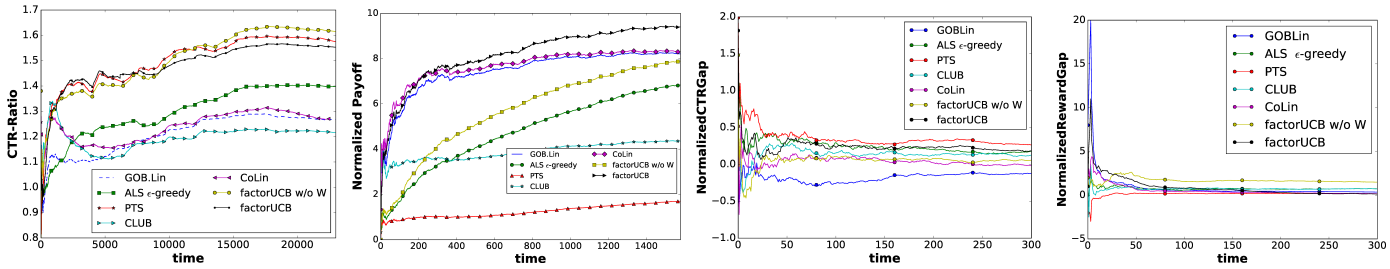
Experiment

We performed extensive empirical evaluations of our proposed factorUCB algorithm against several state-of-the-art factorization-based and bandit-based collaborative filtering methods, including: 1) Alternating Least Square (ALS) with ϵ-greedy (Zhao, Zhang, and Wang 2013), which applies context-free ϵ-greedy algorithm based on both observed features and latent factors, but cannot utilize the user relational graph; 2) Particle Thompson Sampling for matrix factorization (PTS) (Kawale et al. 2015), which combines Thompson sampling with probabilistic matrix factorization based on Rao-Blackwellized particle filter, and it cannot utilize observed features and user relational graph; 3) GOB.Lin (Cesa-Bianchi, Gentile, and Zappella 2013), which models the dependency among a set of contextual bandits over users via a graph Laplacian based model regularization, but it cannot estimate the latent factors; 4) CLUB (Gentile, Li, and Zappella 2014), which clusters users during online learning to enable model sharing; but it only works with contextual features; 5) CoLin (Wu et al. 2016), which imposes a similar collaborative reward generation assumption over the user relational graph as that in our algorithm, but it does not model the latent item factors; 6) factorUCB w/o W, which is factorUCB with an identity $\mathbf{W}$ matrix, i.e., the dependency among users is not considered; it demonstrates of utility of modeling user dependency in interactive recommendation.

Experiments on synthetic dataset

In simulation, we generated a size-K item pool $\mathcal{A}$, in which each item a is associated with a $(d + l)$-dimension feature vector $(\mathbf{x}_a, \mathbf{v}_a)$. Each dimension is drawn from a set of zero-mean Gaussian distributions with variances sampled from a uniform distribution $U(0, 1)$. Principle Component Analysis (PCA) was performed to make all the dimensions *orthogonal* to each other. To simulate the reward generation defined in Eq (3), we used all the $(d+l)$-dimension features to compute the true reward for each item, but only revealed the first d dimensions (i.e., $\mathbf{x}_a$) to an algorithm. We simulated N users, each of who is associated with a $(d+l)$-dimension preference vector $\boldsymbol{\theta}_u^*$. Each dimension of $\boldsymbol{\theta}_u^*$ is drawn from a uniform distribution $U(0, 1)$. $\boldsymbol{\theta}_u^*$ is treated as the ground-truth latent user factor in reward generation, and is unknown to the algorithms. We then constructed the golden relational stochastic matrix $\mathbf{W}$ for the dependency graph of users by defining $w_{ij} \propto \langle\boldsymbol{\theta}_i^*, \boldsymbol{\theta}_j^*\rangle$, and normalize each column of $\mathbf{W}$ by its L1 norm. The resulting $\mathbf{W}$ was disclosed to all the algorithms. To increase the learning complexity, at each trial t, our simulator only disclosed a subset of items in $\mathcal{A}$ to the learners for selection, e.g., randomly selected 10 items from $\mathcal{A}$ without replacement. At each trial t, the same set of items were presented to all the algorithms; and the Gaussian noise η_t in Eq (3) was sampled once for all those items at each trial. We fixed the dimension d of observable features to 20, the dimension l of latent item factors to 5, user size N to 100, the standard derivation σ of Gaussian noise to 0.1, and the item pool size K to 1000 in our simulation.

Cumulated regret defined in Eq (6) was used to evaluate the performance of different algorithms in Figure 1 (a), where we set the dimension for latent factors in PTS to 10 (which gave us the best performance) and 5 in ALS ϵ-greedy and factorUCB. We observed that PTS took much longer time to converge, because PTS cannot utilize the observed



(a) Yahoo (b) LastFM (c) Collaboration on Yahoo (d) Collaboration on LastFM

Figure 2: Experimental comparisons on real-world datasets.

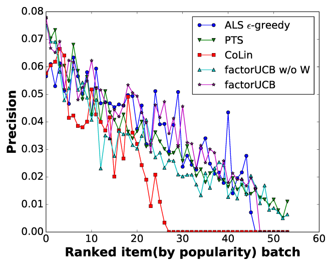
context features for reward prediction, so that it requires much more observations to improve the accuracy of latent factor estimation. Instead, ALS ϵ-greedy and factorUCB leveraged the context features to quickly reduce the reward prediction uncertainty (i.e., less exploration). Two contextual bandits, i.e., GOB.Lin and CoLin, suffered from linear regret, since they do not model the latent item factors. In addition, factorUCB converged much faster than factorUCB w/o W, which confirmed our theoretical analysis about the regret reduction from user dependency modeling. Because factorUCB requires the dimension of latent factor as input, we test its sensitivity to the setting of latent dimension l. To investigate the importance of correct setup of latent factor dimension in factorUCB, we tested two different ways of latent factor construction in our simulator: 1) we chose the top 5 dimensions with the largest eigenvalue from PCA's result as latent item factors, i.e., we hid the top 5 most informative factors in reward generation from the learners; 2) we hid the bottom 5 most informative factors. And on the algorithm side, we varied the dimension of latent factors used in factorUCB from 1 to 7. From the results shown in Figure 1 (b), we can reach three conclusions. First, when the latent factors were the most informative ones, we obtained much worse regret than that in the case of the least informative factors were hidden. Second, the large difference between the regret of a bandit algorithm that does not model the latent factors (such as GOB.Lin) and the one that models latent factors (factorUCB, even with wrong dimensions) emphasizes the necessity of latent factor learning in online recommendation. Third, although our theoretical analysis predicts a linear regret if the latent factor dimension was not accurately set, the actual performance was much more promising. One reason is that our theoretical analysis is for the worst case scenario (upper regret bound), which does not preclude a sub-linear converging regret in practice.

In addition, we also investigated the effect of exploration parameter α_t^u and α_t^a in factorUCB, compared with factorUCB w/o W. In Figure 1 (c), each column illustrates a combination of α_t^u and α_t^a used in factorUCB and factorUCB w/o W. The last column indexed by (α_t^u, α_t^a) represents the theoretical values of those two parameters computed from the algorithm's corresponding regret analysis. As shown in the results, the empirically tuned (α^u, α^a) yielded comparable performance to the theoretical values, and made online computation more efficient. As a result, in all our following experiments we will use the manually set α_t^u and α_t^a.

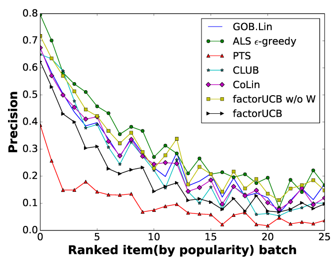
Experiments on real-world datasets

Yahoo dataset: This data set contains 10-days clickstream logs from Yahoo! Today Module collected in May 2009, totalling 45,811,883 user visits (Li et al. 2010). In each logged event, both the user and each of the 10 candidate articles are associated with a feature vector of 6 dimensions. However, this data set does not contain any user identity due to privacy concern. To associate the observations with individual users, we first clustered all users into user groups by applying a k-means algorithm on the given user features. Each logged event was assigned to its closest user group as its user ID. The adjacency matrix $\mathbf{W}$ was estimated by the dot product between the centroids from k-means' output, i.e., $w_{ij} \propto \langle u_i, u_j \rangle$. We set the dimension of latent factors in FactorUCB and ALS ϵ-greedy to 5, and that in PTS to 10.

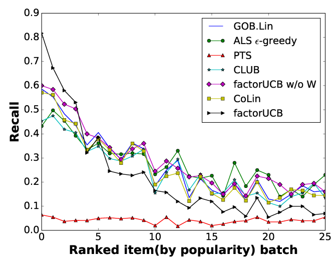
Click-Through-Rate (CTR) was used as the performance metric. Average CTR was computed in every 2000 observations (*not* the cumulated CTR) for each algorithm based on the unbiased offline evaluation protocol developed in (Li et al. 2011; 2010), and normalized by the corresponding logged random strategy's CTR. We reported the normalized CTR results from different algorithms over 160 derived user groups in Figure 2 (a) (similar relative improvement was obtained with different number of derived user groups), where we had several important observations. First, comparing to the conventional contextual bandits (i.e., GOB.Lin, CoLin and CLUB), which do not model the latent factors, factorUCB demonstrated significant improvement in recommendation quality. This proves the benefit of learning latent factors to enhance reward prediction, especially when the given context features are not informative. Second, factorUCB improved more rapidly than PTS and factorUCB w/o W at the early stage. This confirms the value of user dependency modeling for addressing cold-start in online recommendation. But these two factorization-based baselines caught up in the later stage, as more observations became available for them to accurately estimate the latent factors and our approximated user dependency graph might introduce unnecessary bias that prevented factorUCB from accurately recovering the latent factors. This also indicates the importance of accurate user dependency modeling in factorUCB, and we plan to explore $\mathbf{W}$ learning in factorUCB as our future work. Third, the performance improvement in ALS ϵ-greedy was much slower compared with factorUCB (and factorUCB w/o W), while they are using the same information and mechanism for latent factor learning. This further proves the advantage of estimation confidence based



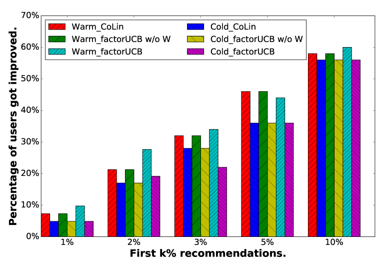
(a) Yahoo



(b) LastFM precision



(c) LastFM recall



(d) LastFM user improvement

Figure 3: Item-based and user-based analysis

exploration strategy over the simple context-free random exploration in interactive recommendation.

LastFM dataset: This dataset is extracted from the online music streaming service Last.fm (`http://www.last.fm`). It contains 1,892 users, 17,632 items (artists), and the users' social network graph. Following the same settings as used in (Cesa-Bianchi, Gentile, and Zappella 2013), we preprocessed the dataset to fit it into an online recommendation setting. The dimension of context features was set to 25 by applying PCA over the text descriptions of each item, and the dimension of latent factors in factorUCB and ALS ϵ-greedy was fixed to 5, and 10 in PTS.

We normalized the cumulated reward from different algorithms by that from a random algorithm, and reported the results in Figure 2 (b). We can clearly notice that PTS performed the worst, while two contextual bandits (i.e., GOB.Lin and CoLin) achieved much better performance than it. This indicates the observed context features in this dataset were sufficiently informative for the algorithms to make accurate recommendations. A purely factorization-based method got penalized by not utilizing such information. On the other hand, we also noticed that factorUCB converged much faster than factorUCB w/o W, which again demonstrates the utility of user dependency modeling for addressing cold-start in recommendation.

To further investigate the effect of modeling context features and user dependency in alleviating cold-start in recommendation, we designed a set of controlled experiments. We first split users into two groups using a max-cut algorithm on the constructed user relational graph to maximize the connectivity between these two groups. Observations in the first user group are called "learning group" and those in the second group are called "testing group." To simulate cold-start, we only executed algorithms on the testing group. Correspondingly, we simulated *warm-start* by first running algorithms on the learning group to pre-estimate the models, and then continued executing them on the testing group. Since users in the testing group were isolated from the learning group, their model parameters could only be initialized by the propagated information via the user relational graph, if an algorithm explicitly modeled that.

We measured the differences in average CTR on Yahoo and differences in cumulated rewards on LastFM between *warm-start* and *cold-start* in Figure 2 (c) and (d). On the Yahoo dataset, factorization-based algorithms (i.e., factorUCB, PTS and ALS ϵ-greedy) benefit the most from the collaboration in latent factor estimation: latent item factors estimated in the learning group helped them better estimate user preferences in testing group. On the LastFM dataset, considerable improvement was achieved in algorithms explicitly modeling user dependency, i.e., factorUCB, GOB.Lin and CoLin. We should note Figure 2 (c) and (d) demonstrated the relative performance improvement from *warm-start* to *cold-start*, so that it does not represent the algorithm's final recommendation quality in these experiments. In the final results, factorUCB achieved the best performance in both *warm-start* and *cold-start* on Yahoo dataset, and *cold-start* on LastFM dataset (factorUCB w/o W was best in *warm-start* in this data set).

We performed both item-based and user-based analysis to understand where the improved recommendations were achieved. On the item side, we computed the precision and recall of the recommendations made by different algorithms on both datasets (because of the offline evaluation protocol used in Yahoo dataset, recall is undefined there) and summarized the results in Figure 3 (a)-(c). In the reported results, we ranked the items based on their popularity in the corresponding datasets. As we can notice that factorUCB achieved encouraging precision over the less popular items in Yahoo dataset, and best recall on popular items in LastFM dataset. In the user side analysis, we investigated how soon a user would receive improved recommendations during the interactive process. We defined an improved user as the user who is served with improved recommendations from a target recommendation algorithm than those from the purely factorization-based algorithm PTS. The design behind this experiment is that because the PTS algorithm cannot utilize observable contextual features nor user dependency relation, it serves as a good basis to assess the value of contextual features and user dependency for online recommendation. In addition, to understand the specific contribution of context feature modeling and user dependency modeling in factorUCB, we also included factorUCB w/o W and CoLin in this analysis, where the CoLin baseline can be considered as factorUCB w/o V. We applied the same evaluation setting as previously used in *warm-start* and *cold-start* comparison, and reported the percentage of improved users in the first 1%, 2%, 3%, 5%, and 10% observations during the interactive recommendation in Figure 3 (d). The results clearly demonstrated that combining both components gave us the most advantage in providing users with improved recommendations in the early stage (first 3% recommendations), and it becomes more beneficial in the *warm-start* setting, where information was prorogated from the users and items in learning group to those in the testing group.

Conclusions

In this work, we studied the problem of online interactive recommendation via a factorization-based bandit algorithm. Observable contextual features and dependency among users are leveraged to improve the algorithm's convergence rate and help conquer cold-start in recommendation. A high probability sublinear upper regret bound is proved, where considerable regret reduction is achieved on both user and item sides. Our current solution assumes the knowledge of ground-truth user dependency and hidden feature dimension. It is important to explore how to determine the user dependency structure and dimension of hidden features during online learning. In addition, our regret analysis is based on the assumption of proper initialization of alternating least square. It is necessary to explore other techniques in matrix analysis or optimization procedures for model parameter estimation and derive the corresponding provable arm selection strategies.

Acknowledgments

We thank all the anonymous reviewers for their helpful comments. This project was supported by the National Science Foundation under grant IIS-1553568 and IIS-1618948.

References

Abbasi-yadkori, Y.; Pál, D.; and Szepesvári, C. 2011. Improved algorithms for linear stochastic bandits. In *NIPS*. 2312–2320.

Agarwal, D., and Chen, B.-C. 2009. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD*, 19–28. ACM.

Auer, P.; Cesa-Bianchi, N.; Freund, Y.; and Schapire, R. E. 1995. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, 322–331.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.* 47(2-3):235–256.

Auer, P. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* 3:397–422.

Candès, E. J., and Recht, B. 2009. Exact matrix completion via convex optimization. *Foundations of Computational mathematics* 9(6):717–772.

Candès, E. J., and Tao, T. 2010. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory* 56(5):2053–2080.

Cesa-Bianchi, N.; Gentile, C.; and Zappella, G. 2013. A gang of bandits. *In Pro. NIPS*.

Gentile, C.; Li, S.; and Zappella, G. 2014. Online clustering of bandits. In *ICML*, 757–765.

Kawale, J.; Bui, H. H.; Kveton, B.; Tran-Thanh, L.; and Chawla, S. 2015. Efficient thompson sampling for online matrix-factorization recommendation. In *Advances in Neural Information Processing Systems*, 1297–1305.

Koren, Y.; Bell, R.; and Volinsky, C. 2009. Matrix factorization techniques for recommender systems. *Computer* (8):30–37.

Koren, Y. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 426–434. ACM.

Li, L.; Chu, W.; Langford, J.; and Schapire, R. E. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of 19th WWW*, 661–670. ACM.

Li, L.; Chu, W.; Langford, J.; and Wang, X. 2011. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of 4th WSDM*, 297–306. ACM.

Li, S.; Karatzoglou, A.; and Gentile, C. 2016. Collaborative filtering bandits. In *arXiv preprint, arXiv:1502.03473*.

Ma, H.; Yang, H.; Lyu, M. R.; and King, I. 2008. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management*, 931–940. ACM.

Ma, H.; Zhou, D.; Liu, C.; Lyu, M. R.; and King, I. 2011. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, 287–296. ACM.

Nakamura, A. 2014. A ucb-like strategy of collaborative filtering. In *ACML*.

Rendle, S.; Gantner, Z.; Freudenthaler, C.; and Schmidt-Thieme, L. 2011. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, 635–644. New York, NY, USA: ACM.

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

Schein, A. I.; Popescul, A.; Ungar, L. H.; and Pennock, D. M. 2002. Methods and metrics for cold-start recommendations. In *Proceedings of 25th SIGIR*, 253–260. ACM.

Su, X., and Khoshgoftaar, T. M. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009:4.

Uschmajew, A. 2012. Local convergence of the alternating least squares algorithm for canonical tensor approximation. *SIAM Journal on Matrix Analysis and Applications* 33(2):639–652.

Wu, Q.; Wang, H.; Gu, Q.; and Wang, H. 2016. Contextual bandits in a collaborative environment. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 529–538. ACM.

Zhao, X.; Zhang, W.; and Wang, J. 2013. Interactive collaborative filtering. In *Proceedings of the 22nd CIKM*, 1411–1420. ACM.