Bandits Warm-up Cold Recommender Systems

Jérémie Mary, Romaric Gaudel, Philippe Preux

Bandits Warm-up Cold Recommender Systems

Jérémie Mary, Romaric Gaudel, Philippe Preux

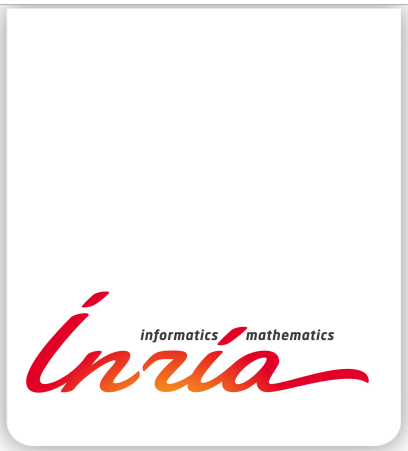
RESEARCH REPORT

N° 8563

July 2014

Project-Team SequeL

ISSN 0249-6399 ISRN INRIA/RR--8563--FR+ENG



Bandits Warm-up Cold Recommender Systems

Jérémie Mary, Romaric Gaudel, Philippe Preux *

Project-Team SequeL

Research Report n° 8563 — July 2014 — 15 pages

Abstract: We address the cold start problem in recommendation systems assuming no contextual information is available neither about users, nor items. We consider the case in which we only have access to a set of ratings of items by users. Most of the existing works consider a batch setting, and use cross-validation to tune parameters. The classical method consists in minimizing the root mean square error over a training subset of the ratings which provides a factorization of the matrix of ratings, interpreted as a latent representation of items and users. Our contribution in this paper is 5-fold. First, we explicit the issues raised by this kind of batch setting for users or items with very few ratings. Then, we propose an online setting closer to the actual use of recommender systems; this setting is inspired by the bandit framework. The proposed methodology can be used to turn any recommender system dataset (such as Netflix, MovieLens,...) into a sequential dataset. Then, we explicit a strong and insightful link between contextual bandit algorithms and matrix factorization; this leads us to a new algorithm that tackles the exploration/exploitation dilemma associated to the cold start problem in a strikingly new perspective. Finally, experimental evidence confirm that our algorithm is effective in dealing with the cold start problem on publicly available datasets. Overall, the goal of this paper is to bridge the gap between recommender systems based on matrix factorizations and those based on contextual bandits.

Key-words: Recommendation system, cold-start problem, matrix factorization, latent factors, multi-armed bandit, contextual bandit, exploration/exploitation dilemma, sequential decision making under uncertainty, efficient algorithm.

* Université de Lille, LIFL (UMR CNRS), INRIA, Villeneuve d'Ascq, France, firstname.lastname@univ-lille3.fr

RESEARCH CENTRE
LILLE – NORD EUROPE

Parc scientifique de la Haute-Borne
40 avenue Halley - Bât A - Park Plaza
59650 Villeneuve d'Ascq

Démarrage à froid des systèmes de recommendation par bandits

Résumé : Nous nous intéressons au problème du démarrage à froid dans les systèmes de recommendation. Nous supposons ne disposer d'aucune information, que ce soit à propos des utilisateurs ou des produits. Nous considérons le cas où nous n'avons accès qu'à un ensemble de notes données à des produits par des utilisateurs. La plupart des travaux concernant ce problème considèrent une approche par lôts et utilisent la validation croisée pour régler les paramètres. La méthode classique consiste à réaliser une décomposition de faible rang de la matrice des notes en minimisant l'erreur quadratique moyenne sur un sous-ensemble des notes disponibles. Cette factorisation est interprêtée comme exhibant des facteurs latents décrivant les produits et les utilisateurs. Dans ce rapport, notre contribution concerne 5 points. Tout d'abord, nous explicitons les problèmes posés par ce type d'approches par lôt pour des utilisateurs ou des produits ayant très peu de notes qui leur sont associées (utilisateurs et produits froids). Ensuite, nous proposons une approche séquentielle qui se rapproche fortement du mode d'utilisation réelle des systèmes de recommendation. Cette approche est inspirée par le problème du bandit multi-bras. Cette méthodologie permet de transformer tout jeu de données issu d'un système de recommendation (tels Netflix, MovieLens, ...) en un jeu de données séquentiel. Alors, nous explicitons une forte connexion entre les bandits contextuels et la factorisation de matrices ; nous pensons que la mise à jour de cette relation est la contribution conceptuelle essentielle de ce rapport ; cette relation éclaire cette problématique d'un jour nouveau. Cela nous amène à un nouvel algorithme qui prend en charge le dilemme exploration/exploitation existant dans le problème du démarrage à froid. Finalement, une étude expérimentale de cet algorithme montre que l'approche fonctionne efficacement pour gérer le démarrage à froid sur des jeux de données disponibles publiquement. Pour résumer nos contributions en une phrase, l'objectif de ce rapport est de mettre à jour un pont entre les systèmes de recommendation basés sur la factorisation de matrices d'une part, les bandits contextuels d'autre part.

Mots-clés : Système de recommendation, démarrage à froid, factorisation de matrice, facteurs latents, bandit multi-bras, bandit contextuel, dilemme exploration/exploitation, prise de décision séquentielle dans l'incertain, algorithme efficace.

1 Introduction

We consider the online version of the problem of the recommendation of items, that is, the one faced by websites. Items may be ads, news, music, videos, movies, books, diapers, ... Daily, of even more often, these systems have to cope with users that have never visited the website, and new items introduced in the catalog. Appetence of the new users towards available items, and appeal of new items towards existing users have to be estimated as fast as possible: this is the cold start problem. Currently, this situation is handled thanks to side information available either about the user, or about the item (see Agarwal et al. 2008; Yue, Hong, and Guestrin 2012). In this paper, we consider this problem from a different perspective. Though perfectly aware of the potential utility of side information, we consider the problem without any side information, only focussing on acquiring appetence of new users and appeal new items as fast as possible; side information can be mixed with the ideas presented in this paper. This combination is left as future work. This poblem fits perfectly into the sequential decision making framework, and more specifically, the bandit without side information setting. However, in rather sharp contrast with the traditional bandit setting, here the set of bandits is continuously being renewed; the number of bandits is not small, though not being huge (from a few dozens to hundreds arms in general, up to dozens of millions in some application): this makes the problem very different from the 2-armed bandit problem, though asymptotic approximation is still irrelevant; we look for efficient and effective ways to achieve this goal, since we want the proposed solution to be able to cope with real applications on the web. For obvious practical and economical reasons for real applications, the strategy can not merely consist in repeatedly presenting all available items to users until the appetence seems accurately estimated. We have to consider the problem as an exploration vs. exploitation problem in which exploration is a necessary evil to acquire information and eventually improve the performance of the recommendation system (RS for short).

This being said, comes the problem of the objective function to optimize. Since the Netflix challenge, at least in the machine learning community, the recommendation problem is often boiled down to a matrix factorization problem, performed in batch, learning on a training set, and minimizing the root mean squared error (RMSE) on a testing set. However, the RMSE comes along very heavy flaws:

- Using the RMSE makes no difference between the items that are highly rated by a user and items poorly rated by the same user; however, for a user, there is a big difference between well rated items and the others: the user wants to be recommended with items she will rate high; she does not care about unattractive items; to illustrate that idea in a rating context alike the Netflix challenge using integers in the range 1 to 5, making an error between a 4 and a 5 is qualitatively very different from making an error between 1 and 2. Furthermore, the restricted set of possible ratings implies that a 5 corresponds to more or less highly rated items. If ratings were real numbers, 5 would spread into $[4.5, 5.5]$ allowing a more precise ranking of preferences by each user. Finally, it is well-known that users have a propensity to rate items they like, rather than rate items they dislike Steck 2010.
- RMSE does not make any difference between the outcome of recommending an item to a heavy user (a user who has already rated a lot of items) as to observe the outcome of the first recommendation to a user during her first visit to the website.
- Usually, the training set and the testing set are unordered, all information regarding the history of the interactions being left aside. Then, we consider average appetence over time, completely neglecting the fact that a given item does not have the same appeal from its

birth to its death, and the fact that the appeal of items is often correlated to the set of available items at a given time, and those available in the past. Koren 2009 has shown the importance of taking timestamps into account.

- Though item recommendation is often presented as a prediction problem, it is really a ranking problem: however, RMSE is not meant to evaluate a ranking Cremonesi, Koren, and Turrin 2010.

The objective function may be tinkered to handle certain of these aspects. However, we think that the one and only way to really handle the problem of recommendation is to address it as a sequential decision making problem, since the history should be taken into ccount. Such a sequential decision making problem faces an exploration vs. exploitation dilemma as detailed in section 4 the exploration being meant to acquire information in order to exploit it and to perform better subsequently; information gathering has a cost that can not be merely minimized to 0, or simply left as an unimportant matter. This means that the evaluation of the recommendation algorithm dealing with the cold start problem has to be done online.

Based on these ideas, our contribution in this paper is the following:

> we propose an original way to tackle the cold start problem of recommendation systems: we cast this problem as a sequential decision making problem to be played online that selects items to recommend in order to optimize the exploration/exploitation balance; our solution is then to perform the rating matrix factorization driven by the policy of this sequential decision problem in order to focus on the most useful terms of the factorization.
>
> The reader familiar with the bandit framework can think of this work as a contextual bandit building its own context from the observed reward using the hypothesis of the existence of a latent space of dimension k.
>
> We also introduce a methodology to use a classical partially filled rating matrix to assess the online performance of a bandit-based recommendation algorithm.

After introducing our notation in the next section, Sec. 3 presents the matrix factorization approach. Sec. 4 introduces the necessary background in bandit theory. In Sec. 5 and Sec. 6, we solve the cold start setting in the case of new users and in the case of new items. Sec. 7 provides an experimental study on artificial data, and on real data. Finally, we conclude and draw some future lines of work in Sec. 8.

2 Notations and Vocabulary

Uppercase, bold-face letters denote matrices, such as: $\mathbf{A}$. $\mathbf{A}^T$ is the transpose matrix of $\mathbf{A}$, and $\mathbf{A}_i$ denotes its i^{th} row i. Lowercase, bold-face letters denote vectors, such as $\mathbf{u}$. $\#\mathbf{u}$ is the number of components (dimension) of $\mathbf{u}$. Normal letters denote scalar value. Except for ζ, greek letters are used to denote the parameters of the algorithms. We use calligraphic letters to denote sets, such as $\mathcal{S}$. $\#\mathcal{S}$ is the number of elements of the set $\mathcal{S}$. For a vector $\mathbf{u}$ and a set of integers $\mathcal{S}$ (s.t. $\forall s \in \mathcal{S}, s \leq \#\mathbf{u}$), $\mathbf{u}_S$ is the sub-vector of $\mathbf{u}$ composed of the elements of $\mathbf{u}$ which indices are contained in $\mathcal{S}$. Accordingly, $\mathbf{U}$ being a matrix, $\mathcal{S}$ a set of integers smaller or equal to the number of lines of $\mathbf{U}$, $\mathbf{U}_\mathcal{S}$ is the sub-matrix made of the rows of $\mathbf{U}$ which indices form $\mathcal{S}$ (the ordering of the elements in $\mathcal{S}$ does not matter, but one can assume that the elements of the set are sorted). Now, we introduce a set of notations dedicated to the RS problem. We consider:

- as we consider a time-evolving number of users and items, we will note n the current number of users, and m the current number of items. These should be indexed by a t to denote time, though often in this paper, t is dropped to simplify the notation. i indices the users, whereas j indices the items. Without loss of generality, we assume $n < N$ and $m < M$, that is N and M are upper bounds of the number of ever seen users and items (those figures may as large as necessary).

- $\mathbf{R}^*$ represents the ground truth, that is the matrix of ratings. Obviously in a real application, this matrix is unknown. Each row is associated to one and only one user, whereas each column is associated to one and only one item. Hence, we will also use these row indices, and column indices to represent users, and items.

  $\mathbf{R}^*$ is of size $N \times M$. $r^*_{i,j}$ is the rating given by user i to item j.

  We suppose that there exists an integer k and two matrices $\mathbf{U}$ of size $N \times k$ and $\mathbf{V}$ of size $M \times k$ such that $\mathbf{R}^* = \mathbf{U}\mathbf{V}^T$. This is a standard assumption Dror et al. 2011.

- Not all ratings have been observed. We denote $\mathcal{S}$ the set of elements that have been observed (yet). Then we define $\mathbf{R}$:

$$r_{i,j} = \begin{cases} r^*_{i,j} + \eta_{i,j} \text{ if } (i,j) \in \mathcal{S} \\ \text{NA otherwise} \end{cases}$$

  where $\eta_{i,j}$ is a noise with zero mean, and finite variance. The $\eta_{i,j}$ are i.i.d.

  In practice, the vast majority of elements of $\mathbf{R}$ are unknown.

  In this paper, we assume that $\mathbf{R}^*$ is fixed during all the time; at a given moment, only a submatrix made of n rows and m columns is actually useful. This part of $\mathbf{R}^*$ that is observed is increasing along time. That is, the set $\mathcal{S}$ is growing along time.

- $\mathcal{J}(i)$ denotes the set of indices of the columns with available values in row number i of $\mathbf{R}$ (*i.e.* the set of items rated by user i). Likewise, $\mathcal{I}(j)$ denotes the sets of rows of $\mathbf{R}$ with available values for column j (*i.e.* the set of users who rated item j).

- Symbols i and $\mathcal{I}$ are related to users, thus rows of the matrices containing ratings, while symbols j and $\mathcal{J}$ refer to items, thus columns of these matrices.

- $\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$ denote estimates (with the statistical meaning) of the matrices $\mathbf{U}$ and $\mathbf{V}$ respectively. Their product $\hat{\mathbf{U}}\hat{\mathbf{V}}^T$ is denoted $\hat{\mathbf{R}}$. The relevant part of matrices $\mathbf{R}^*$ and $\mathbf{R}$ has dimensions $n_t \times m_t$ at a given moment t, $\mathbf{U} \in \mathbb{R}^{n_t \times k}$ and $\mathbf{V} \in \mathbb{R}^{m_t \times k}$.

To clarify things, let us consider $n = 4$ users, $m = 8$ items, and $\mathbf{R}^*$ as follows:

$$\mathbf{R}^* = \begin{pmatrix} 3 & 2 & 3 & 4 & 1 & 2 & 5 & 4 \\ 1 & 1 & 4 & 3 & 3 & 5 & 2 & 1 \\ 5 & 1 & 4 & 3 & 2 & 1 & 1 & 2 \\ 2 & 4 & 5 & 5 & 4 & 3 & 1 & 1 \end{pmatrix}$$

Let us suppose that $\mathcal{S} = \{(1,3),(1,6),(2,1),(2,4),(2,6),(3,2),(4,4),(4,6),(4,7)\}$, then assuming no noise:

$$\mathbf{R} = \begin{pmatrix} \text{NA} & \text{NA} & 3 & \text{NA} & \text{NA} & 2 & \text{NA} & \text{NA} \\ 1 & \text{NA} & \text{NA} & 3 & \text{NA} & 5 & \text{NA} & \text{NA} \\ \text{NA} & 1 & \text{NA} & \text{NA} & \text{NA} & \text{NA} & \text{NA} & \text{NA} \\ \text{NA} & \text{NA} & \text{NA} & 5 & \text{NA} & 3 & 1 & \text{NA} \end{pmatrix}$$

We use the term "observation" to mean a triplet (user, item, rating of this item by this user such as $(2, 1, 1)$). Each known value of $\mathbf{R}$ is an observation. The RS receives a stream of observations. We use the term "rating" to mean the value associated by a user to an item. It can be a rating as in the Netflix challenge, or a no-click/click, no-sale/sale, ...

For the sake of legibility, in the online setting we omit the t subscript for time dependency. In particular, $\mathcal{S}$, $\hat{\mathbf{U}}$, $\hat{\mathbf{V}}$, n, m should be subscripted with t.

3 Matrix Factorization

Since the Netflix challenge Bennett, Lanning, and Netflix 2007, many works have been using matrix factorization: the matrix of observed ratings is assumed to be the product of two matrices of low rank k. We refer the interested reader to Koren, Bell, and Volinsky 2009 for a short survey. As most of the values of the rating matrix are unknown, the factorization can only be done using this set of observed values. The classical approach is to solve the regularized minimization problem $(\hat{\mathbf{U}}, \hat{\mathbf{V}}) \stackrel{def}{=} \operatorname{argmin}_{\mathbf{U},\mathbf{V}} \zeta(\mathbf{U}, \mathbf{V})$ where:

$$\zeta(\mathbf{U}, \mathbf{V}) \stackrel{def}{=} \sum_{\forall (i,j) \in \mathcal{S}} \left(r_{i,j} - \mathbf{U}_i \cdot \mathbf{V}_j^T\right)^2 + \lambda \cdot \Omega(\mathbf{U}, \mathbf{V}),$$

in which $\lambda \in \mathbb{R}^+$ and the usual regularization term is:

$$\Omega(\mathbf{U}, \mathbf{V}) \stackrel{def}{=} ||\mathbf{U}||^2 + ||\mathbf{V}||^2 = \sum_i ||\mathbf{U}_i||^2 + \sum_j ||\mathbf{V}_j||^2.$$

ζ is not convex. The minimization is usually performed either by stochastic gradient descent (SGD), or by alternate least squares (ALS). ALS-WR Zhou et al. 2008 weighs users and items according to their respective importance in the matrix of ratings.

$$\Omega(\mathbf{U}, \mathbf{V}) \stackrel{def}{=} \sum_i \#\mathcal{J}(i) ||\mathbf{U}_i||^2 + \sum_j \#\mathcal{I}(j) ||\mathbf{V}_j||^2.$$

This regularization is known to have a good empirical behavior — that is limited overfitting, easy tuning of λ and k, low RMSE.

4 Bandits

Let us consider a bandit machine with m independent arms. When pulling arm j, the player receives a reward drawn from $[0, 1]$ which follows a probability distribution ν_j. Let μ_j denote the mean of ν_j, $j^* \stackrel{def}{=} \operatorname{argmax}_j \mu_j$ be the best arm and $\mu^* \stackrel{def}{=} \max_j \mu_j = \mu_{j^*}$ be the best expected reward (we assume there is only one best arm). The parameters $\{\nu_j\}$, $\{\mu_j\}$, j^* and μ^* are unknown.

A player aims at maximizing its cumulative reward after T consecutive pulls. More specifically, by denoting j_t the arm pulled at time t and r_t the reward obtained at time t, the player wants to maximize the quantity $\text{CumRew}_T = \sum_{t=1}^{T} r_t$. As the parameters are unknown, at each time-step (except the last one), the player faces the dilemma:

- either *exploit* by pulling the arm which seems the best according to the estimated values of the parameters;

- or *explore* to improve the estimation of the parameters of the probability distribution of an arm by pulling it;

A well-known approach to handle the exploration vs. exploitation trade-off is the Upper Confidence Bound strategy (UCB) Auer, Cesa-Bianchi, and Fischer 2002 which consists in playing the arm j_t:

$$j_t = \underset{j}{\operatorname{argmax}}\, \hat{\mu}_j + \sqrt{\frac{2 \ln t}{t_j}}, \tag{1}$$

where $\hat{\mu}_j$ denotes the empirical mean reward incured when on pulls of arm j up to time t and t_j corresponds to the number of pulls of arm j since $t = 1$. UCB is optimal up to a constant. This equation clearly expresses the exploration-exploitation trade-off: while the first term of the sum ($\hat{\mu}_j$) tends to exploit the seemingly optimal arm, the second term of the sum tends to explore less pulled arms.

Li *et al.* Li et al. 2010 extend the bandit setting to contextual arms. They assume that a vector of real features $\mathbf{v} \in \mathbb{R}^k$ is associated to each arm and that the expectation of the reward associated to an arm is $\mathbf{u}^* \cdot \mathbf{v}$, where $\mathbf{u}^*$ is an unknown vector. The algorithm handling this setting is known as LinUCB. LinUCB follows the same scheme as UCB in the sense that it consists in playing the arm with the largest upper confidence bound on the expected reward:

$$j_t = \underset{j}{\operatorname{argmax}}\, \hat{\mathbf{u}}.\mathbf{v}_j^T + \alpha \sqrt{\mathbf{v}_j \mathbf{A}^{-1} \mathbf{v}_j^T},$$

where $\hat{\mathbf{u}}$ is an estimate of $\mathbf{u}^*$, α is a parameter and $\mathbf{A} = \sum_{t'=1}^{t-1} \mathbf{v}_{j_{t'}}.\mathbf{v}_{j_{t'}}^T + \mathbf{Id}$, where $\mathbf{Id}$ is the identity matrix. Note that $\hat{\mathbf{u}}.\mathbf{v}_j^T$ corresponds to an estimate of the expected reward, while $\sqrt{\mathbf{v}_j \mathbf{A}^{-1} \mathbf{v}_j^T}$ is an optimistic correction of that estimate.

While the objective of UCB and LinUCB is to maximize the cumulative reward, theoretical results Li et al. 2010; Abbasi-yadkori, Pal, and Szepesvari 2011 are expressed in term of *cumulative regret* (or regret for short)

$$\text{Regret}_T \overset{def}{=} \sum_{t=1}^{T} r_t^* - r_t,$$

where r_t^* stands for the best expected reward at time t (either μ^* in the UCB setting or $\max_j \mathbf{u}^*.\mathbf{v}_{j_t}^T$ in the LinUCB setting). Hence, the regret measures how much the player looses (in expectation), in comparison to playing the optimal strategy. Standard results proove regrets of order $\tilde{O}(\sqrt{T})$ or $O(\ln T)$, depending on the assumptions on the distributions and depending on the precise analysis [1].

Of course LinUCB, and more generally contextual bandits require the context (values of features) to be provided. In real applications this is done using side information about the items and the users Shivaswamy and Joachims 2011 –*i.e.* expert knowledge, categorization of items, Facebook profiles of users, implicit feedback ... The core idea of this paper is to use matrix factorization techniques to build a context online using the known ratings. To this end, one assumes that the items and the arms can be represented in the same space of dimension k and assuming that the rating of user u for item v is the scalar product of u and v.

We study the introduction of new items and/or new users into the RS. This is done without using any side information on users or items.

[1] $\tilde{O}$ means O up to a logarithmic term on T.

5 Cold Start for a New User

Let us now consider a particular recommendation scenario. At each time-step t,

1. a user i_t requests a recommendation to the RS,
2. the RS selects an item j_t among the set of items that have never been recommended to user i_t beforehand,
3. user i_t returns a rating $r_t = r_{i_t,j_t}$ for item j_t.

Obviously, the objective of the RS is to maximize the cumulative reward $\text{CumRew}_T = \sum_{t=1}^{T} r_t$.

In the context of such a scenario, the usual matrix factorization approach of RS recommends item j_t which has the best predicted rating for user i_t. This corresponds to a pure exploitation (greedy) strategy of bandits setting, which is well-known to be suboptimal to manage CumRew_T: to be optimal, the RS has to balance the exploitation and exploration.

Let us now describe the recommendation algorithm we propose at time-step t. We aim at recommending to user i_t an item j_t which leads to the best trade-off between exploration and exploitation in order to maximize CumRew_∞. We assume that the matrix $\mathbf{R}$ is factorized into $\hat{\mathbf{U}}\hat{\mathbf{V}}^T$ by ALS-WR - discussed later - which terminated by optimizing $\hat{\mathbf{U}}$ holding $\hat{\mathbf{V}}$ fixed. In such a context, the UCB approach is based on a confidence interval on the estimated ratings $\hat{r}_{i_t,j} = \hat{\mathbf{U}}_{i_t} \cdot \hat{\mathbf{V}}_j^T$ for any allowed item j.

We assume that we already observed a sufficient number of ratings for each item, but only a few ratings (possibly none) from user i_t. As a consequence the uncertainty on $\hat{\mathbf{U}}_{i_t}$ is much more important than on any $\hat{\mathbf{V}}_j$. In other words, the uncertainty on $\hat{r}_{i_t,j}$ mostly comes from the uncertainty on $\hat{\mathbf{U}}_{i_t}$. In the following, we express this uncertainty.

Let $\mathbf{u}^*$ denote the (unknown) true value of $\mathbf{U}_{i_t}$ and let us introduce the $k \times k$ matrix:

$$\begin{aligned}\mathbf{A} &\stackrel{def}{=} (\hat{\mathbf{V}}_{\mathcal{J}(i_t)})^T \cdot \hat{\mathbf{V}}_{\mathcal{J}(i_t)} + \lambda \cdot \#\mathcal{J}(i_t) \cdot \mathbf{Id} \\ &= \sum_{j \in \mathcal{J}(i_t)} \hat{\mathbf{V}}_j^T \cdot \hat{\mathbf{V}}_j + \lambda \cdot \#\mathcal{J}(i_t) \cdot \mathbf{Id}.\end{aligned}$$

As shown by Zhou et al. 2008, as $\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$ comes from ALS-WR (which last iteration optimized $\hat{\mathbf{U}}$ with $\hat{\mathbf{V}}$ fixed),

$$\begin{aligned}\hat{\mathbf{U}}_{j_t} &= \mathbf{A}^{-1} \sum_{j \in \mathcal{J}(i_t)} \hat{\mathbf{V}}_j^T \cdot r_{i_t,j} \\ &= \mathbf{A}^{-1} \hat{\mathbf{V}}_{\mathcal{J}(i_t)}^T \mathbf{R}_{i_t,\mathcal{J}(i_t)}^T.\end{aligned}$$

Using Azuma's inequality over the weighted sum of random variables (as introduced by Walsh et al. 2012 for linear systems), it follows that there exists a value $C \in \mathbb{R}$ such as, with probability $1 - \delta$:

$$(\hat{\mathbf{U}}_{i_t} - \mathbf{u}^*)\mathbf{A}^{-1}(\hat{\mathbf{U}}_{i_t} - \mathbf{u}^*)^T \leq C\frac{\log(1/\delta)}{t}$$

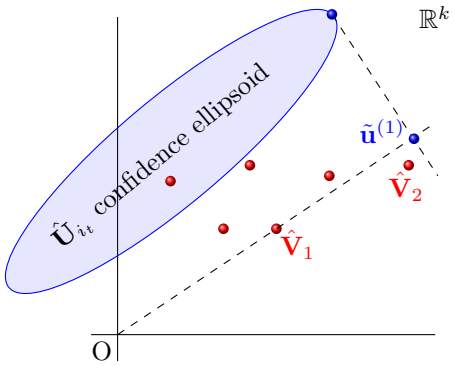


Figure 1: This figure illustrates the use of the upper confidence ellipsoid for item selection in the context of a new user. As explained in the paper, items and users are represented as vectors in $\mathbb{R}^k$. In the figure, the red dots correspond to the known items vectors. The blue area indicates the confidence ellipsoid on the unknown vector associated to the user. The optimistic rating of the user for item 1 is the maximum scalar product between $\hat{\mathbf{V}}_1$ and any point in this ellipsoid. By a simple geometrical argument based on iso-contours of the scalar product, this maximum scalar product is equal to the scalar product between $\hat{\mathbf{V}}_1$ and $\tilde{\mathbf{u}}^{(1)}$. The optimistic recommendation system recommends the item maximizing the scalar product $\langle \tilde{\mathbf{u}}^{(j)}, \hat{\mathbf{V}}_j \rangle$.

This inequality defines the confidence bound around the estimate $\hat{\mathbf{U}}_{i_t}$ of $\mathbf{u}^*$. Therefore, a UCB strategy selects item j_t:

$$j_t \stackrel{def}{=} \operatorname*{argmax}_{1 \leq j \leq m, j \notin \mathcal{J}(i_t)} \max_{\mathbf{u}, s.t. ||\mathbf{u} - \hat{\mathbf{U}}_{i_t}||_A^2 < C \frac{\log(1/\delta)}{t}} \hat{\mathbf{U}}_{i_t} \cdot \hat{\mathbf{V}}_j$$

which amounts to:

$$j_t = \operatorname*{argmax}_{1 \leq j \leq m, j \notin \mathcal{J}(i_t)} \hat{\mathbf{U}}_{i_t} \cdot \hat{\mathbf{V}}_j^T + \alpha \sqrt{\hat{\mathbf{V}}_j \mathbf{A}^{-1} \hat{\mathbf{V}}_j^T}$$

where $\alpha \in \mathbb{R}$ is an exploration parameter to be tuned. Fig. 1 illustrates the transition from the maximum on a confidence ellipsoid to its closed-form $\hat{\mathbf{U}}_{i_t} \cdot \hat{\mathbf{V}}_j^T + \alpha \sqrt{\hat{\mathbf{V}}_j \mathbf{A}^{-1} \hat{\mathbf{V}}_j^T}$.

Our complete algorithm, named BeWARE.User (which stands for "Bandit WARms-up REcommenders") is described in Alg. 1. The presentation is optimized for clarity rather than for computational efficiency. Of course, if the exploration parameter α is set to 0 BeWARE.User chooses the same item as ALS-WR. The estimate of the center of the ellipsoid and its size can be influenced by the use of an other regularization term. BeWARE.User uses a regularization based on ALS-WR. It is possible to replace all $\#\mathcal{J}(.)$ by 1. This amounts to the standard regularization: we call this slightly different algorithm BeWARE.ALS.User. In fact one can use any regularization ensuring that $\hat{\mathbf{U}}_{i_t}$ is a linear combination of observed rewards. Please, note that BeWARE.ALS.User with $\lambda = 1$ is a LinUCB building its context using matrix decomposition - if the $\hat{\mathbf{V}}$ matrix does not changes after observation this is exactly a LinUCB.

5.1 Discussion on the Analysis of BeWARE.User

The analysis of BeWARE.User is rather similar to the LinUCB proof Abbasi-yadkori, Pal, and Szepesvari 2011 but it requires to take care of the vectors of context which in our case are es-

Algorithm 1 BeWARE.User: for a user i_t, selects and recommends an item to this user.

Input: i_t, λ, α
Input/Output: $\mathbf{R}$, $\mathcal{S}$
1: $(\hat{\mathbf{U}}, \hat{\mathbf{V}}) \leftarrow \text{MatrixFactorization}(\mathbf{R})$
2: $\mathbf{A} \leftarrow (\hat{\mathbf{V}}_{\mathcal{J}(i_t)})^T \cdot \hat{\mathbf{V}}_{\mathcal{J}(i_t)} + \lambda \cdot \#\mathcal{J}(i_t) \cdot \mathbf{Id}$.
3: $j_t \leftarrow \underset{j \notin \mathcal{J}(i_t)}{\text{argmax}}\, \hat{\mathbf{U}}_{i_t} \cdot \hat{\mathbf{V}}_j^T + \alpha\sqrt{\hat{\mathbf{V}}_j \mathbf{A}^{-1} \hat{\mathbf{V}}_j^T}$
4: Recommend item j_t and receive rating $r_t = r_{i_t,j_t}$
5: Update $\mathbf{R}$, $\mathcal{S}$

timated through a matrix decomposition. As matrix decomposition error bounds are classically not distribution free Chatterjee 2012 (they require at least independancy between the observations), we cannot provide a complete proof. However, we can have one for a modified algorithm using the same LinUCB degradation as Chu et al. 2011. The trick is to inject some independancy in the observed values in order to guarantee an unbiased estimation of V.

6 Cold Start for New Items

When a new item is added, it is a larger source of uncertainty than the descriptions of the users. To refect this fact, we compute a confidence bound over the items instead of the users. As the second step of the ALS is to fix $\hat{\mathbf{U}}$ and optimize $\hat{\mathbf{V}}$, it is natural to adapt our algorithm to handle the uncertainty on $\hat{\mathbf{V}}$ accordingly. This will take care of the exploration on the occurrence of new items. With the same criterion and regularization on $\hat{\mathbf{V}}$ as above, we obtain at timestep t:

$$\mathbf{B}(j) \stackrel{def}{=} (\hat{\mathbf{U}}_{\mathcal{I}(j)})^T \hat{\mathbf{U}}_{\mathcal{I}(j)} + \lambda \cdot \#\mathcal{I}(j) \cdot \mathbf{Id}$$

and

$$\hat{\mathbf{V}}_j = \mathbf{B}(j)^{-1} (\hat{\mathbf{U}}_{\mathcal{I}(j)})^T \mathbf{R}_{\mathcal{I}(j),j}.$$

So considering the confidence ellipsoid on $\hat{\mathbf{V}}$, the upper confidence bound of the rating for user i on item j is

$$\hat{\mathbf{U}}_i \cdot \hat{\mathbf{V}}_j^T + \alpha\sqrt{\hat{\mathbf{U}}_j \mathbf{B}(j)^{-1} \hat{\mathbf{U}}_j^T}.$$

This leads to the algorithm BeWARE.Item presented in Alg. 2. Again, the presentation is optimized for clarity rather than for computational efficiency. BeWARE.Item can be parallelized and has the complexity of one step of ALS. Fig. 2 gives the geometrical intuition leading to BeWARE.Item. Again, setting $\alpha = 0$ leads to the same selection as ALS-WR. The regularization (on line 4) can be modified. This algorithm has no straightforward interpretation in terms of LinUCB.

7 Experimental Investigation

In this section we evaluate empirically our family of algorithms on artificial data, and on real datasets. The BeWARE algorithms are compared to:

- greedy approaches (denoted Greedy.ALS and Greedy.ALS-WR) that always choose the item with the largest current estimated value (respectively given a decomposition obtained by ALS, or by ALS-WR),

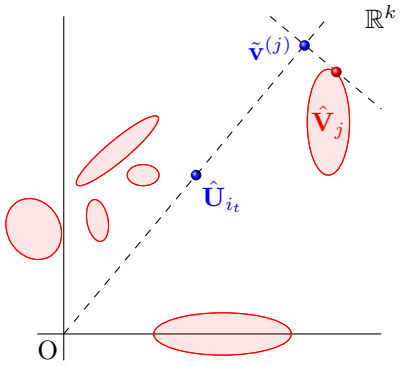
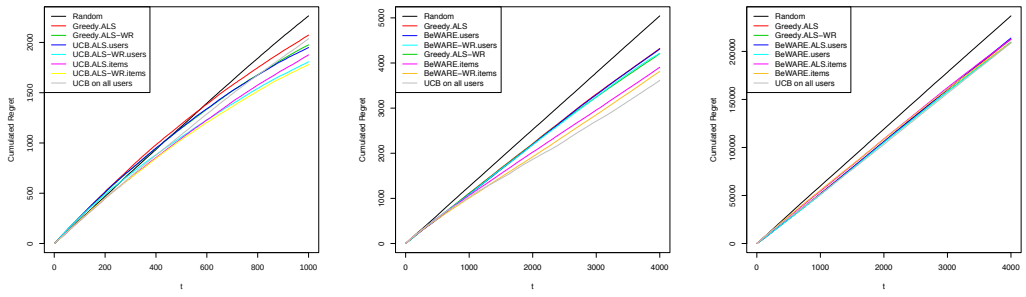


Figure 2: This figure illustrates the use of the upper confidence ellipsoid for item selection in the context of new items. The setting is similar to Fig. 1 except that the vector associated to the user is known (blue dot) while the items vectors live in confidence ellipsoids. The optimistic recommendation system recommends the item maximizing the scalar product $\langle \hat{\mathbf{U}}_{i_t}, \tilde{\mathbf{v}}^{(j)} \rangle$.

Algorithm 2 BeWARE.Item: for a user i_t, selects and recommends one of the new items to this user.

Input: i_t, λ, α
Input/Output: $\mathbf{R}$, $\mathcal{S}$

1: $(\hat{\mathbf{U}}, \hat{\mathbf{V}}) \leftarrow \text{MatrixFactorization}(\mathbf{R})$
2: $\forall j \notin \mathcal{J}(i_t),\ \mathbf{B}(j) \leftarrow (\hat{\mathbf{U}}_{\mathcal{I}(j)})^T \hat{\mathbf{U}}_{\mathcal{I}(j)} + \lambda \cdot \#\mathcal{I}(j) \cdot \mathbf{Id}$
3: $j_t \leftarrow \underset{j \notin \mathcal{J}(i_t)}{\text{argmax}}\, \hat{\mathbf{U}}_{i_t}.\hat{\mathbf{V}}_j^T + \alpha \sqrt{\hat{\mathbf{U}}_{i_t} \mathbf{B}(j)^{-1} \hat{\mathbf{U}}_{i_t}^T}$
4: Recommend item j_t and receive rating $r_t = r_{i_t, j_t}$
5: Update $\mathbf{R}$, and $\mathcal{S}$



(a) Artificial dataset. (b) Netflix dataset. (c) Yahoo!Music dataset.

Figure 3: Cumulated regret (the lower, the better) for a set of 100 new items and 200 users with no prior information. Figures are averaged over 20 runs (for Netflix and artificial data, $k = 5,\ \lambda = 0.05,\ \alpha = 0.12$ whereas for Yahoo!Music, $k = 8,\ \lambda = 0.2,\ \alpha = 0.05$). On the artificial dataset (a), BeWARE.items is better than the other strategies in terms of regret. On the Netflix dataset (b), `UCB on all users` is the best approach and BeWARE.items is the second best. On the Yahoo!Music dataset (c), BeWARE.items, Greedy.ALS-WR and UCB all 3 lead to similar performances.

- the UCB1 approach Auer, Cesa-Bianchi, and Fischer 2002 (denoted UCB.on.all.users) that consider each reward r_{i_t,j_t} as an independent realization of a distribution ν_{j_t}. In other words, UCB.on.all.users recommends an item without taking into account the information on the user requesting the recommendation.

On the one hand, the comparison to greedy approaches highlights the needs of exploration to have an optimal algorithm in the online context. On the other hand, the comparison to UCB.on.all.users is there to assess the benefit of personalizing recommendations.

7.1 Experimental Setting

For each dataset, algorithms start with an empty $\mathbf{R}$ matrix of 100 items and 200 users. Then, the evaluation goes like this:

1. select a user uniformly at random among those who have not yet rated all the items,
2. request his favorite item among those he has not yet rated,
3. compute the immediate regret (the difference of rating between the best not yet selected item and the one selected according to $\widetilde{\mathbf{R}}^*$ for this user),
4. iterate until all users have rated all items.

The difficulty with real datasets is that the ground truth is unknown, and actually, only a very small fraction of ratings is known. This makes the evaluation of algorithms uneasy. To overcome these difficulties, we also provide a comparison of the algorithms considering an artificial problem based on a ground truth matrix $\mathbf{R}^*$ considering m users and n items. This matrix is generated as in Chatterjee 2012. Each item belongs to either one of k genres, and each user belongs to either

one of l types. For each item j of genre a and each user i of type b, $r^*_{i,j} = p_{a,b}$ is the ground truth rating of item j by user i, where $p_{a,b}$ is drawn uniformly at random in the set $\{1, 2, 3, 4, 5\}$. The observed rating $r_{i,j}$ is a noisy value of $r^*_{i,j}$: $r_{i,j} = r^*_{i,j} + \mathcal{N}(0, 0.5)$.

We also consider real datasets, the NetFlix dataset Bennett, Lanning, and Netflix 2007 and the Yahoo!Music dataset Dror et al. 2011. Of course, the major issue with real data is that there is no dataset with a complete matrix, which means we do no longer have access to the ground truth $\mathbf{R}^*$, which makes the evaluation of algorithms more complex. This issue is usually solved in the bandit literature by using a method based on reject sampling Li et al. 2011. For a well constructed dataset, this kind of estimators has no bias and a known bound on the decrease of the error rate Langford, Strehl, and Wortman 2008.

For all the algorithms, we restrict the possible choices for a user at time-step t to the items with a known rating in the dataset. However, a minimum amount of ratings per user is needed to be able to have a meaningful comparison of the algorithms (otherwise, a random strategy is the only reasonable one). As a consequence, with both datasets, we focus on the 5000 heaviest users for the top $\sim$250 movies/songs. This leads to a matrix $\widetilde{\mathbf{R}}^*$ with only 10% to 20% of missing ratings. We insist on the fact that this is necessary for performance evaluation of the algorithms; obviously, this is not required to use the algorithms on a live RS.

For people used to work on full recommendation dataset the experiment can seem small. But one has to keep in mind several points:

- Each value in the matrix corresponds to one possible observation. After each observation we are allowed to update our recommender policy. This means that for 4000 observations we need to perform 4000 matrix decompositions.
- To evaluate precisely Beware, we would need the rating of any user on any item (because Beware may choose any of the items for the current user). In the dataset many of the ratings are unknown so using part of the matrix with many unknown ratings would introduce a bias in the evaluation.

We would like to advertize this experimental methodology which has a unique feature: indeed, this methodology allows us to turn any matrix –or tensor– of ratings into an online problem which can be used to test bandit recommendation algorithms. This is of interest because there is currently no standard dataset to evaluate bandits algorithms. To be able to evaluate offline any bandit algorithm on real data, one has to collect data using a random uniform strategy and use a replay like methodology Langford, Strehl, and Wortman 2008. To the best of our knowledge, the very few datasets with desired properties are provided by Yahoo Webscope program (R6 dataset) as used in the challenge Mary et al. 2012. These datasets are only available to academics which restrain their use. So, it is very interesting to be able to use a more generally available rating matrix (such as the Netflix dataset) to evaluate an online policy. We think that this methodology is an other contribution of this paper. A similar trick has already been used in reinforcement learning to turn a turn a reinforcement learning into a supervised classification task Lagoudakis and Parr 2003.

7.2 Experimental Results

Figures 3(a) and 3(b) show that given a fixed factorization method, BeWARE strategies improve the results on the Greedy-one. Looking more closely at the results, BeWARE based on items uncertainty performs better than BeWARE based on users uncertainty, and BeWARE.users is the only BeWARE strategy beaten by its greedy counterpart (Greedy.ALS-WR) on the Netflix dataset. These results demonstrate that an online strategy has to care about exploration to tend towards optimality.

While UCB.on.all.users is almost the worst approach over Artificial data (Fig. 3(a)), it surprisingly performs better than all other approaches over Netflix dataset. We feel that this difference is strongly related to the preprocessing of the Netflix dataset we have done to be able to follow the experimental protocol (and have an evaluation at all). By focusing on the top $\sim$250 movies, we keep blockbusters that are appreciated by everyone. With that particular subset of movies, there is no need to adapt the recommendation user per user. As a consequence, UCB.on.all.users suffers a smaller regret than other strategies, as it considers users as n independent realizations of the same distribution. It is worth noting that UCB.on.all.users regret would increase with the number of items while the regret of BeWARE scales with the dimensionality of the factorization, which makes BeWARE a better candidates for real applications with much more items to deal with.

Last, on Fig. 3(c) all approaches suffer the same regret.

7.3 Discussion

In a real setting, BeWARE.Item has a desirable property: it tends to favor new items with regards to older ones because they have less feedback than the others, hence larger confidence bound. So the algorithm gives them a boost which is exactly what a webstore is willing — if a webstore accepts new products this is because he feels the new one are potentially better than the old ones. Moreover it will allow the recommender policy to use at its best the novelty effect for the new items. This natural attraction of users with regards to new items can be very strong as it has been shown by the Exploration & Exploitation challenge at ICML'2012 which was won by a context free algorithm Mary et al. 2012.

The computational cost of the BeWARE methods is the same as doing an additional step of alternate least squares; moreover some intermediate calculations of the QR factorization can be re-used to speed up the computation. So the total cost of BeWARE.Item is almost the same as ALS-WR. Even better, while the online setting requires to recompute the factorization at each time-step, this factorization sightly changes from one iteration to the next one. As a consequence, only a few ALS-WR iterations are needed to update the factorization. Overall the computational cost stays reasonable even in a real application.

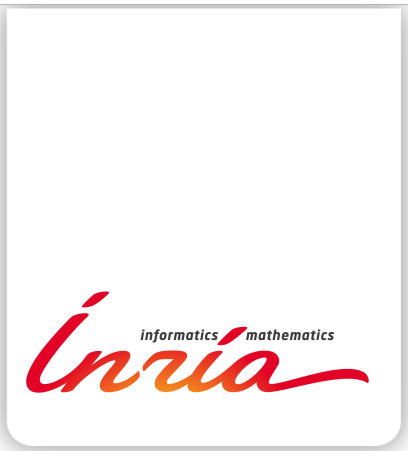
8 Conclusion and Future Work

In this paper, we introduced the idea of using bandit algorithm as a principled, and effective way to solve the cold start problem in recommendation systems. We think this contribution is conceptually rich, and opens ways to many different studies. We showed on large, publicly available datasets that this approach is also effective, leading to efficient algorithms able to work online, under the expected computational constraints of such systems. Furthermore, the algorithms are quite easy to implement.

Many extensions are currently under study. First, we work on extending these algorithms to use contextual information about users, and items. This will require combining the similarity measure with confidence bounds; this might be translated into a Bayesian prior. We also want to analyze regret bound for large enough number of items and users. This part can be tricky as LinUCB still does not have a full formal analysis, though some insights are available in Abbasi-yadkori, Pal, and Szepesvari 2011.

An other important point is to work on the recommendation of several items at once and get feedback only for the best one. There is some work in the non contextual bandits on this point, one could try to translate it in our framework Cesa-Bianchi and Lugosi 2012.

Finally, we plan to combine confidence ellipsoid about both users and items — this is not a straightforward sum of the bounds. However, we feel that such a combination has low odds to provide better results for real application, but it is interesting from a theoretical perspective, and should lead to even better results on artificial problems.



RESEARCH CENTRE
LILLE – NORD EUROPE

Parc scientifique de la Haute-Borne
40 avenue Halley - Bât A - Park Plaza
59650 Villeneuve d'Ascq

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399