Exploiting Item Co-utility to Improve Collaborative Filtering Recommendations

A. Bessa
Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil.
E-mail: aline.bessa@nyu.edu

R.L.T. Santos
Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil.
E-mail: rodrygo@dcc.ufmg.br

A. Veloso
Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil.
E-mail: adrianov@dcc.ufmg.br

N. Ziviani
Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil.
E-mail: nivio@dcc.ufmg.br; Kunumi, Belo Horizonte, Brazil. E-mail: nivio@kunumi.com

In this article we study the extent to which the interplay between recommended items affect recommendation effectiveness. We introduce and formalize the concept of co-utility as the property that any pair of recommended items has of being useful to a user, and exploit it to improve collaborative filtering recommendations. We present different techniques to estimate co-utility probabilities, all of them independent of content information, and compare them with each other. We use these probabilities, as well as normalized predicted ratings, in an instance of an $\mathcal{NP}$-hard problem termed the Max-Sum Dispersion Problem (MSDP). A solution to MSDP hence corresponds to a set of items for recommendation. We study one heuristic and one exact solution to MSDP and perform comparisons among them. We also contrast our solutions (the best heuristic to MSDP) to different baselines by comparing the ratings users give to different recommendations. We obtain expressive gains in the utility of recommendations and our solutions also recommend higher-rated items to the majority of users. Finally, we show that our co-utility solutions are scalable in practice and do not harm recommendations' diversity.

Received June 23, 2016; revised December 19, 2016; accepted December 21, 2016

• Published online 6 July 2017 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/asi.23853

Introduction

People from widely varying backgrounds are inundated with options that lead to a situation known as information overload, where the presence of too much information interferes with decision-making processes. To circumvent it, content providers and electronic retailers have to identify a small yet effective amount of information that matches users' expectations. In this scenario, recommender systems have become tools of paramount importance, providing personalized recommendations that intend to suit user needs in a satisfactory way.

The dominant type of recommender systems is known as collaborative filtering. It makes predictions about the interests of a user by gathering taste information from many other users, and works as follows: i) prediction step—keeps track of users' known preferences and processes them to predict items that may be interesting to other users; ii) recommendation step—selects predicted items, optionally ranks them, and recommends them to users (Adomavicius & Tuzhilin, 2005). In the prediction step, scores are independently assigned to items by taking users' historical data into account (Ricci, Rokach, Shapira, & Kantor, 2011). The higher the score, the higher the estimated compatibility between the item and the user. It is thus intuitive that recommending the highest scored items should result in the highest accuracy. Nonetheless, accurate recommendations are not necessarily useful ones, because other dimensions or properties associated with the recommended items may affect recommendation effectiveness. Examples of dimensions or properties that are typically taken into account during the recommendation step include diversity or competition (Xiong, Wang, Wenkui Ding, & Liu, 2012; Zhang & Hurley, 2008).

In this article we focus on the co-utility property, which is the property that any pair of items has of being useful to a user—two items are co-useful with respect to a user if s/he considers both of them useful. The motivation behind using this property comes from the Theory of Choice, which indicates that preference among items depends not only on the items' specific features, but also on the presented alternatives (Tversky, 1972). In our case, the selection of an item is based not only on its independently predicted rating, but also on how likely it is to be co-useful with other selected items. More specifically, for each pair of items we compute their probabilities of being co-useful and use this information in methods designed to generate recommendations. Some of the contributions of this work include: i) a definition of co-utility and methods to estimate co-utility probabilities; ii) an objective function that combines predicted values and co-utility probabilities, its reduction to a popular Facility Location Analysis problem (Borodin, Lee, & Ye, 2012), and algorithms to tackle it; iii) a comparison between the usefulness of our method and different baselines; and iv) an analysis of the scalability, diversity, and optimality of recommendations produced by our method.

In the remainder of this article, "Background and Related Work" section positions this work in the related literature. "Combining Individual and Pairwise Scores" section introduces our approach to estimate co-utility probabilities, whereas "Algorithms" section presents algorithms to tackle the optimization problem involved in exploiting co-utility. "Experimental Setup" and "Experimental Results" sections discuss the setup and results of the empirical evaluation of our approach. Lastly, "Conclusions and Future Work" section concludes the article.

Background and Related Work

Collaborative filtering is traditionally concerned with predicting the feedback a user would give to an item. Several predictors have been studied for collaborative filtering, which can be broadly grouped into two classes: *memory-based* and *model-based* (Breese, Heckerman, & Kadie, 1998). Memory-based predictors operate over the entire database to compute similarities between users or items, usually by applying distance metrics such as the cosine distance, and then they produce predictions. In contrast, model-based predictors use the database to learn models, and then use the learned models for predictions. Instead of producing a numeric prediction for a given user-item pair, collaborative filtering can be tackled as a ranking task, often referred to as Top-*n* recommendation (Cremonesi, Koren, & Turrin, 2010), in which the goal is to produce a list of items to be recommended for a given user.

Dependency-Agnostic Recommendation

Traditionally, the items included in a recommendation list are selected independently from one another, for example, based on their individually estimated recommendation scores. When explicit feedback is available, the state-of-the-art dependency-agnostic approaches for Top-*n* recommendation are NNCosNgbr (Non-normalized Cosine Neighborhood) and PureSVD (Pure Singular Value Decomposition) (Cremonesi et al., 2010). The NNCosNgbr algorithm is memory-based and works on the concept of neighborhood, computing predictions according to the feedback given to similar items or users—in this article, we focus on similar items. The algorithm computes similarities between items with the adjusted cosine similarity, and also takes biases into consideration, which are related to how users rate items. Item biases include the fact that certain items tend to receive better feedback than others. Similarly, user biases include the tendency that certain users have of giving better feedback than others. In contrast, the PureSVD algorithm is model-based and works on latent factors, that is, users and items are modeled as vectors in the same space (Cremonesi et al., 2010). PureSVD factorizes a matrix filled up with numerical feedback given by users to items, and then predicts the score of user u for item i via the inner-product between their corresponding vectors.

In this article, competitive dependency-agnostic Top-*n* recommenders are important for two reasons. First, the optimization problem we tackle uses individual item scores that correspond to the predictions generated by such recommenders. Second, our approach extends Top-*n* recommendations by addressing dependencies among items—that is, by employing dependency-aware algorithms—and thus we use the studied predictors for Top-*n* recommendations as baselines.

Dependency-Aware Recommendation

Attempts to abandon the assumption that items are independent date back from information retrieval studies in the 1980s. By that time, researchers started questioning the Probability Ranking Principle (PRP), according to which documents should be retrieved in decreasing order of their predicted probabilities of relevance (Robertson, 1977). Bookstein (1983), for instance, presented decision-theoretic ranking models that take document interactions into account iteratively. Later on, researchers started to focus on diversity-based reranking, and they also had to address relations among items to reduce intersimilarities (Santos, Macdonald, & Ounis, 2015). In particular, Carbonell and Goldstein (1998) proposed the concept of Maximal Marginal Relevance (MMR) to diminish redundancy while maintaining query relevance. MMR is a criterion that has been widely adopted in search and recommendation contexts (Carbonell & Goldstein, 1998; Santos, Macdonald, & Ounis, 2010; Vargas & Castells, 2011; Zuccon, Azzopardi, Zhang, & Wang, 2012). It consists of a ranking formula that, as well as our method, takes the individual relevance of items and relations among them both into account. Given the wide scope of applications for MMR, there are different ways of implementing it, but generally, at each iteration MMR returns the highest-valued item with respect to a tradeoff between relevance and diversity.

In the context of recommender systems, several works exploited relations among items to improve diversity (Ribeiro et al., 2014). Zhang and Hurley (2008) modeled the competing goals maximizing relevance and diversity as a binary optimization problem, relaxed to a trust-region problem. Wang (2009) presented a document ranking paradigm inspired by the Modern Portfolio Theory in finance (Elton, Gruber, Brown, & Goetzmann, 2009), where both the mean relevance of predictions and their variance are taken into account. In that context, variance works as a measure of risk. Based on this mean-variance principle, they devised a document ranking algorithm, abbreviated henceforth as MVA. Zuccon et al. (2012) showed how Facility Location Analysis, taken from Operations Research, works as a generalization for state-of-the-art retrieval models for diversification in search. They treated the Top-*n* search results as facilities that should be dispersed as far as possible from each other, and implemented MMR by using Kullback–Leibler divergence as the distance metric for pairs of items.

Relations among items other than diversity have also been exploited in the search and recommendation literature. Tversky (1972) proposed a model according to which preference among items is influenced by the presented alternatives. The model, called Elimination By Aspects (EBA), states that a consumer chooses among options by sets of aspects, eliminating items that do not satisfy such aspects. A variation of EBA for commerce search was proposed by Sheffet, Mishra, and Ieong, (2012), who introduced the Random Shopper Model, where each item feature is a Markov network over the items to be ranked, and the goal is to find a weighting of the features that best reflects their importance. Relatedly, Xiong et al. (2012) observed that the Click-Through Rate (CTR) of an ad is often influenced by the other ads showed alongside. They designed a continuous conditional random field for click prediction focusing on how similarities influence items' CTRs. Weston and Blitzer (2012) also incorporated interitem similarity during ranking to improve recall. They used a latent structured model to learn the structure of the ranked list while assigning scores to items, merging prediction, and recommendation steps. Hansen and Golbeck (2009) addressed the task of recommending collections of items—music lists and mix tapes, for example. This task is different from the one we tackle, given that in their problem each recommended item is actually a collection of items (mix tapes, for instance). In spite of that, they also considered relations between items as an aspect that contributes to the overall value of a collection. In particular, they modeled the value of individual items, co-occurrence interaction effects, and order effects including placement and arrangement of items.

In this article we adopt Wang (2009) and Zuccon et al. (2012)'s techniques as baselines. The former is close to ours because it exploits correlations between documents, via variance, in a collaborative filtering scenario, even though its focus is on diversity. The latter relates to our work because they also use Facility Location Analysis as a framework, although also focused on diversity. Given that our method and theirs share the same theoretical framework, we think it is appropriate to compare them. We do not compare our method with that of Weston and Blitzer (2012) because what they present is an improvement over a specific class of latent factor models, whereas our method is also suitable for memory-based approaches. As for Sheffet et al. (2012), we discarded it because it requires information about item features, and therefore it is not a pure collaborative filtering method.

Max-Sum Dispersion Problem (MSDP)

The exploration of relationships among items is becoming popular in the Recommender Systems literature. Some works, including Zuccon et al. (2012) and Vieira et al. (2011), consider the setting where they are given a set of candidate items I and a set valuation function f defined on every subset of I. For any subset $R \subseteq I$, the overall objective is a linear combination of $f(R)$ and the sum of dissimilarities induced by the items in R. The goal is to find a subset R with a given cardinality constraint—for example, $|R|=5$ if five items must be selected out of I—that maximizes the overall objective (Borodin et al., 2012). Our objective, as discussed in "Combining Individual and Pairwise Scores" section, is similar to this. Our valuation function is the sum of predicted ratings for items in R and we combine it with the sum of co-utility probabilities induced in R.

These objectives map into a well-known Facility Location Analysis problem: the weighted version of the MSDP. MSDP is a well-studied problem in Operations Research (Gollapudi & Sharma, 2009). A common scenario is the placement of facilities in a given area in such a way that the distances between them, as well as their individual relevances, are maximized. Analytical models for MSDP assume that an area is represented by a set $V=\{v_1,\ldots,v_k\}$ of k vertices with a metric distance between every pair of vertices. The objective is to locate $n \leq k$ facilities such that some function of distances between facilities, combined with individual relevances, is maximized. MSDP is known to be $\mathcal{NP}$-hard, but it admits approximation algorithms in some cases. As we show in "Combining Individual and Pairwise Scores" section, approximations are not admitted in our case.

Combining Individual and Pairwise Scores

In this work we propose to exploit two fundamental sources of evidence in order to select which items should be recommended to a user: i) individual scores ϕ, that correspond to ratings predicted by any Top-*n* recommender, and ii) pairwise scores θ that quantify co-utility probabilities among items. Scores ϕ and θ are always real values, and they are combined in a bi-criteria optimization problem. In "Pairwise Scores" section we present techniques to compute pairwise scores θ and in "Combining Scores Using MSDP" section we present techniques to combine individual and pairwise scores using MSDP.

Pairwise Scores

In this section we address different techniques for estimating pairwise scores θ. The pairwise score θ_{ij} represents the probability of items i and j being co-useful to any user. If we consider E_{ij} as a random variable that represents the event *"Items i and j are co-useful to l_{ij} users,"* and assume that E_{ij} follows a binomial distribution, then its probability mass function is given by:

$$f(l_{ij};f_{ij},\theta_{ij}) = \binom{l_{ij}}{f_{ij}} \theta_{ij}^{l_{ij}} (1-\theta_{ij})^{f_{ij}-l_{ij}}, \tag{1}$$

where f_{ij} is the number of users that gave feedback to both i and j.

To estimate θ_{ij}, we employed the estimators Maximum Likelihood and Empirical Bayes (Bishop, 2006). Maximum Likelihood gives the maximum of $f(l_{ij};f_{ij},\theta_{ij})$ by using the point where its derivative is zero and its second derivative is negative. Assuming that $f(l_{ij};f_{ij},\theta_{ij}) \neq 0$, the derivation of Maximum Likelihood leads to:

$$\theta_{ij} = \frac{l_{ij}}{f_{ij}} \in [0,1]. \tag{2}$$

Maximum Likelihood is simple but it is not always suitable when pairs of items have poor support. This is very common in recommender systems, as users give feedback to a very small fraction of items. Empirical Bayes has the advantage of being more robust when not much data are available. An estimate score with Empirical Bayes for the number of users l_{ij} that liked both items i and j, with probability of co-utility θ_{ij} for items i and j, can be derived by combining a conjugate prior for the binomial distribution as a prior distribution on θ_{ij} and a beta-binomial distribution for the marginal distribution of l_{ij}. In our case, to estimate scores with Empirical Bayes we follow the rationale exemplified in Casella (1992) for the coin tossing problem,[1] whose modeling is adequate for the estimation of θ_{ij}. Consider the following prior distribution on θ_{ij}:

$$\pi(\theta_{ij}) = 6\theta_{ij}(1-\theta_{ij}), \tag{3}$$

which is symmetric around $\frac{1}{2}$, indicating that we have no prior opinion as to which side of $\frac{1}{2}$ a specific θ_{ij} lies. We do not assume anything about how co-useful items i and j are because this can vary significantly from one pair of items to another. This prior is a conjugate prior density, which greatly simplifies the ensuing calculations. We calculate the distribution of θ_{ij} given l_{ij} as:

$$\pi(\theta_{ij}|l_{ij}) = \frac{\Gamma(f_{ij}+4)}{\Gamma(l_{ij}+2)\Gamma(f_{ij}-l_{ij}+2)} \times \theta_{ij}^{l_{ij}+1} (1-\theta_{ij})^{f_{ij}-l_{ij}+1}. \tag{4}$$

Details about this derivation can be found in Casella (1992).

This posterior distribution contains all information necessary for Bayesian inference (Casella, 1992). If a point estimate of θ_{ij} is needed, a Bayes point estimator is given by the mean of $\pi(\theta_{ij}|l_{ij})$, which is what we effectively use in this paper:

$$E(\theta_{ij}|l_{ij}) = \int_0^1 \theta_{ij}\pi(\theta_{ij}|l_{ij})d\theta_{ij} = \frac{f_{ij}}{f_{ij}+4} \times \frac{l_{ij}}{f_{ij}} + \left(1 - \frac{f_{ij}}{f_{ij}+4} \times \frac{1}{2}\right). \tag{5}$$

To compute θ_{ij} with either Maximum Likelihood or Empirical Bayes, we assume that the random variable E_{ij} is the same for any user u and therefore θ_{ij} is independent of the user in question. Another important consideration is that, ideally, E_{ij} should only imply that i and j were co-useful if they were presented in the same recommendation. Unfortunately, the data sets used in our experiments do not include such information. Hence, we compute θ_{ij} by considering f_{ij} and l_{ij} regardless of temporality. To give an example, if a user liked *Titanic* in November 2012 and *Matrix* in June 2011, we consider that they were co-useful to her/him even though s/he was not presented with them simultaneously. It is important to stress that this is a limitation of our experimental setup, but not of our model. Note as well that our model does not take into account how co-utility varies from one user to another. This is a simplification that turned the model much more scalable and easy to implement, as there were fewer random variables with values to be estimated.

It is crucial to point out that scores θ differ from collaborative filtering item-to-item similarities. In particular, these similarities take all feedback into account. For instance, if a set of common users rated two items negatively, this contributes to their cosine similarity as much as positive ratings would. In the case of scores θ, what is measured is co-utility—not similarity—and only feedback attesting that items were actually useful—for example, rated positively—is taken into consideration.

Another critical distinction between scores θ and item-to-item similarities has to do with their scope. Item-to-item similarities are computed between two sets of items in the prediction step: i) items that are part of the user's historical data, and ii) items to which the user has not given feedback yet. The idea is to retrieve candidates for recommendation that are likely to match the user's taste. In this step, no relation among the retrieved candidates is taken into account. Scores θ, on the other hand, capture the co-utility probabilities of pairs of retrieved candidates.

Combining Scores Using MSDP

In this section we present a formulation to MSDP in order to combine individual and pairwise scores to select n items out of k for recommendation. Our maximization problem is therefore posed as selecting a set of items $R=\{i_1,\ldots,i_n\}$ that maximizes the following function:

[1] In the coin tossing problem, a coin is tossed n times and the unknown probability of a head is p. The estimator is designed to estimate the observed number y of heads. In our case, l_{ij} can be interpreted as y, p as θ_{ij} and f_{ij} as n.

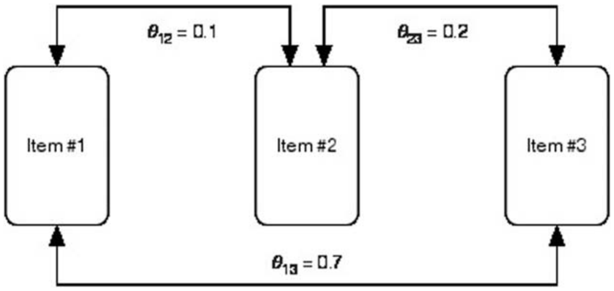

FIG. 1. The triangle inequality is not satisfied, as $\theta_{13} \geq \theta_{12}+\theta_{23}$.

$$\frac{1}{|R|}\sum_{i_j \in R}\phi_{i_j}+\frac{1}{|R|^2}\sum_{(i_a,i_b)\in R^2}\theta_{i_a i_b}, \qquad (6)$$

where the normalization in both summations is important to keep their contributions fair. Scores ϕ and θ are also normalized to the interval $[0, 1]$.

Structurally, this problem is an instance of MSDP. As previously mentioned, MSDP is a Facility Location Analysis problem, which is $\mathcal{NP}$-hard. When pairwise scores θ satisfy the triangle inequality, MSDP admits a two-approximation algorithm (Borodin et al., 2012). On the other hand, it was demonstrated that if the triangle inequality is not satisfied, there is no polynomial time approximation algorithm to MSDP unless $\mathcal{P}=\mathcal{NP}$ (Ravi, Rosenkrantz, & Tayi, 1994). The co-utility probabilities that we exploit in this work, namely, pairwise scores θ, do not satisfy the triangle inequality, as illustrated in Figure 1. Hence none of the algorithms we analyze in this work have bounds on solution quality.

There are different ways of obtaining an exact solution to this optimization problem. For instance, one can trivially enumerate all n-combinations of a set with k items and choose the one that sums up to the highest value. It is also possible to use integer programming to solve it. We describe how we obtain exact solutions to MSDP in “Algorithms” section.

Algorithms

To tackle MSDP under a practical viewpoint, we studied a suboptimal, polynomial algorithm that is widely related to this problem. We also study an integer programming approach to MSDP. This problem cannot be solved efficiently by exact algorithms, albeit it is important to understand how it can be optimally solved.

Greedy

A popular heuristic to MSDP, here referred to as Greedy, was proposed by Borodin et al. (2012). Greedy is popular because, when pairwise scores θ satisfy the triangle inequality, it is a two-approximation algorithm to MSDP. It runs fast and yields acceptable solutions in practice, even when scores θ do not satisfy the triangle inequality. Greedy is shown in Algorithm 1. I is a set of items, I_ϕ corresponds to their individual scores, and I_θ corresponds to their pairwise scores. The output R is a set with n selected items, where $n \leq k$. Greedy starts by selecting the item that has the best individual score i in line 1. All other $n-1$ selected items are chosen in a way that maximizes the equation in line 5, where the maximized set is comprised by all items R that were already chosen and the new item itself.

Algorithm 1 Greedy

```
Input: $I=\{i_1,\ldots,i_k\}$, $I_\phi=\{\phi_{i_1},\ldots,\phi_{i_k}\}$, $I_\theta=\{\theta_{i_1 i_2}, \theta_{i_1 i_3},\ldots,\theta_{i_{k-1} i_k}\}$, and $1 \leq N \leq |I|$
Output: Selected items $R$
1: $i \Leftarrow \mathrm{argmax}_{i \in I}\phi_i$
2: $R \Leftarrow \{i\}$
3: $I \Leftarrow I \setminus \{i\}$
4: while $|R| < N$ do
5:     $j \Leftarrow \mathrm{argmax}_{j \in I}\frac{\phi_j}{2}+\frac{1}{|R|}\sum_{k \in R}\theta_{jk}$
6:     $R \Leftarrow R \cup \{j\}$
7:     $I \Leftarrow I \setminus \{j\}$
8: end while
9: return $R$
```

Exact Solution

Since MSDP is $\mathcal{NP}$-hard, it can only be solved efficiently by suboptimal algorithms. Despite that, it is important to understand how to model an exact algorithm to MSDP, especially if comparisons between optimal and suboptimal solutions are of interest. We decided to model MSDP under the integer programming paradigm because of the rather fast exact solvers available. The parameters to model our integer programming problem are a set of items $I=\{i_1,\ldots,i_k\}$, their corresponding individual scores $I_\phi=\{\phi_{i_1},\ldots,\phi_{i_k}\}$, the pairwise scores for all combinations of items in I, $I_\theta=\{\theta_{i_1 i_2},\ldots,\theta_{i_{k-1} i_k}\}$, and the number of items for selection n. We come up with binary variables $Y=\{y_1,\ldots,y_k\}$ to represent which items are selected ($y_j=1$ if and only if i_j is selected), and rewrite MSDP as:

$$\begin{array}{ll} \text{maximize} & \frac{1}{|I|}\sum_{j \in I} y_j\phi_j+\frac{1}{|I|^2}\sum_{j \in I}\sum_{k \in I | k \neq j} y_j y_k \theta_{jk}, \\ \text{subject to} & y_i \in \{0,1\} \quad \forall i, \\ & \sum_{y_i \in Y} y_i=N. \end{array} \qquad (7)$$

To frame this program in the integer programming paradigm, we have to linearize MSDP’s products $y_j y_k$ as variables $x_{jk}=y_j y_k\ \forall j, \forall k$. Considering that y_j and y_k are binary variables, we have the following constraints for variables x_{jk}:

$$\begin{aligned} x_{jk} &\leq y_j \\ x_{jk} &\leq y_k \\ x_{jk} &\geq y_j+y_k-1. \end{aligned} \qquad (8)$$

TABLE 1. Statistics of the data sets used in our investigations.

Feature	MovieLens-100K	MovieLens-1M	Jester-1
Domain	Movies	Movies	Jokes
Number of users	943	6,040	24,983
Number of items	1,682	3,900	100
Number of ratings	100,000	1,000,209	1,810,455
Minimum ratings/user	20	20	36
Sparsity rate	0.937	0.958	0.275
Ratings range	[1,5]	[1,5]	[-10,10]
Mean rating value	3.588	3.703	1.877

That being stated, we rewrite our problem as:

$$\begin{aligned} \text{maximize} \quad & \frac{1}{|I|}\sum_{j\in I} y_j\phi_j + \frac{1}{|I|^2}\sum_{j\in I}\sum_{k\in I|k\neq j} x_{jk}\theta_{jk} \\ \text{subject to} \quad & y_i \in \{0,1\} \quad \forall i \\ & x_{jk} \leq y_j \\ & x_{jk} \leq y_k \\ & x_{jk} \geq y_j + y_k - 1 \\ & \sum_{y_i \in Y} y_i = N. \end{aligned} \tag{9}$$

The algorithms addressed in this section are compatible with any recommender system where it is possible to estimate individual scores ϕ and pairwise scores θ for candidate items. Therefore, these algorithms are *a priori* compatible with systems that employ both matrix factorization techniques and fingerprinting methods.

Experimental Setup

In this section we discuss the experimental setup that supports our investigations in "Experimental Results" section. In particular, we aim to answer the following research questions:

- **Q1.** How useful are our produced recommendations?
- **Q2.** How diverse are our produced recommendations?
- **Q3.** How far from optimal are our produced recommendations?
- **Q4.** How scalable is our method?

Studied Data sets

For the experiments described in "Experimental Results" section we used three data sets: MovieLens-100K,[2] MovieLens-1M,[3] and Jester-1.[4] We worked with the MovieLens data sets and Jester-1 due to their popularity in the collaborative filtering literature. In this section we present a characterization of these data sets in order to facilitate posterior experiment analyses.

Table 1 summarizes some of the data sets' main features. We can see that the ratings in the MovieLens data sets are discretized and vary from 1 to 5, and users in Jester-1 can assign any real number from −10 to 10 to any joke. Another key difference is that the MovieLens data sets are significantly sparser than Jester-1. In the former, users rated at least 20 movies, whereas in the latter feedback was given to at least 36 jokes. Considering that there are only 100 jokes in Jester-1, this value corresponds to a minimum of 36%.

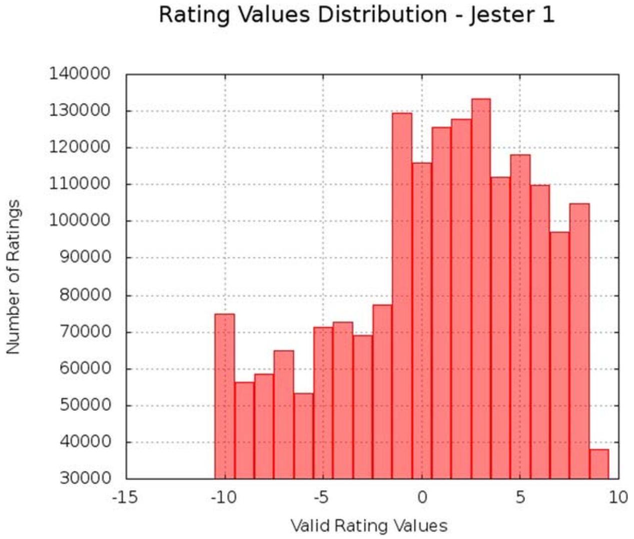

FIG. 2. Distributions of ratings for Jester-1. Each bar consists of ratings in intervals [-10.00; −9.00), [−9.00; −8.00), ..., [8.00, 9.00), [9.00; 10.00]. [Color figure can be viewed at wileyonlinelibrary.com]

As for the mean rating value, Table 1 indicates that MovieLens users tend to give average-to-good ratings to movies. This reveals that users prefer to manifest their tastes by rating movies they find enjoyable. As for Jester-1, the mean rating is more neutral. With respect to the MovieLens data sets, we adopted 4 as a threshold for high ratings, as in Ricci et al. (2011). As for Jester-1, we decided to adopt 5.00 as a threshold for high ratings because this value is considerably higher than its mean rating and than the interval $[2.00; 3.00)$, which is associated with most ratings in this data set, as illustrated by Figure 2. It is important to choose a value that is above interval $[2.00; 3.00)$ because one could claim that most users were neutral with respect to the jokes they rated.[5]

Recommendation Baselines

The baselines that we use can be divided into two categories: dependency-agnostic and dependency-aware. Dependency-agnostic baselines do not assume interdependencies among items for recommendation, generate simple Top-*n* recommendations, and in this work they are associated with predictors PureSVD and NNCosNgbr. Individual scores are predicted for items and, in a dependency-agnostic fashion, the Top-*n* ones are recommended. Dependency-aware baselines take individual item scores and relations among them

[2] http://www.grouplens.org/system/files/ml-100k.zip

[3] http://www.grouplens.org/system/files/ml-1m.zip

[4] http://goldberg.berkeley.edu/jester-data/jester-data-1.zip

[5] We believe that we could alternatively have used a value higher than 5.00 as well, but given the extent of our experiments we only used one threshold value for each data set, and 5.00 was a reasonable choice.

into consideration. In this work, these baselines are MVA and MMR, introduced in "Dependency-Aware Recommendation" section. MVA and MMR exploit relations among items as a means of improving diversity in recommendation lists, while keeping relevance.

For all studied methods, individual scores were predicted by PureSVD with 50 latent factors and NNCosNgbr. Greedy's pairwise scores were calculated with the Empirical Bayes estimator. We set MVA's parameter α, that works as a risk regulator, as 0.05 after a grid search involving values that ranged from -5 to 5, that is, MVA is slightly risk-lower in our experiments and prompted the best MVA's results. As to MVA's covariance matrix, we computed it by considering, for every pair of items, the ratings they received by a common set of users. Finally, MMR uses a term regulator λ to balance the contribution of items dissimilarities in the generation of final scores. We used $\lambda = 1$ because it prompted the best results in a grid search over values from 0.01 to 1. The computation of dissimilarities between items' rating distributions was made by using Kullback–Leibler divergence.[6]

Validation Metrics

To validate our work, we use the explicit feedback users give over items as a utility metric: the better the feedback, the more useful the recommendations are (Breese et al., 1998; Liu & Belkin, 2015). For example, in movie ratings, where a 5-star movie is considered an excellent movie, we can assume that recommending a 5-star movie is more useful than recommending a 4-star one (Ricci et al., 2011). Considering that we are focused on recommendations' utility, and that we use ratings as a utility metric, we compare different algorithms by contrasting the ratings their selected items receive.

We applied cross-validation in all experiments, and randomly partitioned the data sets into training and test data. Consequently, we ignored rating timestamps, whenever they were present, while splitting the data. Cross-validation is interesting in our case because we only analyze three data sets, and by crossing training and data partitions we increase the number of different scenarios on which we run experiments. Considering that recommendation lists are generated over items in the test data, to which we know the actual ratings, our experiments simulate scenarios where users would rate all recommended items. Other works that opt for cross-validation are Vargas and Castells (2011) and Sarwar, Karypis, Konstan, and Riedl, (2001).

For each data set, the training data are explored by predictors PureSVD and NNCosNgbr to generate individual scores ϕ. The training data are also used to estimate pairwise scores θ and other pairwise information required by the baselines.

Finally, for all experiments, recommendation lists have sizes $|R| = 5, 10, 20$ because these are popular values in the related literature. We report means of values per fold and, additionally, means of ratings in some experiments. In this work we also make comparisons under a diversity perspective. In our scope, diversity is defined as the opposite of similarity, and although this is not the focus of our work, we briefly investigate whether our method hurts recommendations' diversity. The diversity metric we apply, intra-list distance (ILD), was proposed by Zhang and Hurley (2008) and works as follows:

$$\mathrm{ILD} = \frac{2}{|R|(|R|-1)} \sum_{i_k, i_l \in R, l<k} 1 - \mathrm{sim}(i_k, i_l), \qquad (10)$$

where R is comprised by all selected items and $\mathrm{sim}(i_k, i_l)$ is a generic similarity measurement for items i_k and i_l. Further discussions on how we computed items' similarities and performed experiments with ILD are presented in "Experimental Results" section.

Experimental Results

In this section we present experiments and results that address and answer research questions Q1, Q2, Q3, and Q4 presented in "Experimental Setup" section. In "Recommendation Usefulness" section we compare Greedy with dependency-agnostic and dependency-aware baselines in order to understand how useful Greedy's recommendations are. In "Relating Co-Utility and Diversity" section we investigate how diverse Greedy's recommendations are. In "Recommendation Optimality" section we compare recommendations obtained with Greedy and recommendations that correspond to exact solutions to MSDP. Finally, In "Analyzing the Scalability of Our Method" section, we discuss Greedy's scalability.

Recommendation Usefulness

In this section we compare Greedy with different baselines in order to answer research question Q1: How useful are our produced recommendations? We use dependency-agnostic, Top-*n* baselines with predictors PureSVD and NNCosNgbr and dependency-aware baselines MVA and MMR. It is important to mention that Greedy, MVA, and MMR use individual *and* pairwise scores, and in all cases individual scores were generated by using either PureSVD or NNCosNgbr.

For all experiments reported in the tables, we assessed the significance of statistical equivalences and differences by applying paired *t*-tests with a 95% confidence interval. The paired *t*-tests were applied over distributions of mean predicted ratings (one mean per user in the test fold) and lowest predicted ratings (one per user in the test fold). The lowest predicted ratings could be approximated by a Gaussian distribution in all data sets and means of ratings, considering that the sample sizes for the paired *t*-tests were always at least 200, could also be approximated by a Gaussian distribution according to the Central Limit Theorem (Hastie, Tibshirani, & Friedman, 2001). It is important to keep in mind that, for all experiments, very distinct reported average

[6] The use of rating distributions is particularly indispensable when a strict collaborative filtering schema has to be adopted, or when no information about the items is available. This is the scenario assumed for most of our experiments.

TABLE 2. Mean average rating for recommendation lists of size $n = 5$, 10, and 20.

	@5	@10	@20
MovieLens-100K			
PureSVD	3.924 ± 0.030	3.837 ± 0.023	3.738 ± 0.019
+MVA	3.923 ± 0.030	3.830 ± 0.023	3.720 ± 0.019
+MMR	3.884 ± 0.030	3.799 ± 0.023	3.698 ± 0.019
+Greedy	3.987 ± 0.029	3.881 ± 0.023	3.768 ± 0.019
NNCosNgbr	3.821 ± 0.031	3.775 ± 0.024	3.691 ± 0.019
+MVA	3.833 ± 0.031	3.768 ± 0.024	3.677 ± 0.019
+MMR	3.801 ± 0.031	3.760 ± 0.024	3.677 ± 0.019
+Greedy	3.896 ± 0.031	3.818 ± 0.023	3.732 ± 0.019
MovieLens-1M			
PureSVD	4.127 ± 0.011	4.004 ± 0.008	3.908 ± 0.007
+MVA	4.128 ± 0.011	4.013 ± 0.008	3.901 ± 0.007
+MMR	4.120 ± 0.011	4.012 ± 0.008	3.900 ± 0.007
+Greedy	4.187 ± 0.011	4.065 ± 0.008	3.941 ± 0.007
NNCosNgbr	4.027 ± 0.011	3.928 ± 0.009	3.836 ± 0.007
+MVA	4.027 ± 0.011	3.927 ± 0.009	3.824 ± 0.007
+MMR	4.022 ± 0.011	3.926 ± 0.009	3.832 ± 0.007
+Greedy	4.082 ± 0.011	3.988 ± 0.009	3.902 ± 0.007
Jester-1			
PureSVD	1.292 ± 0.028	1.031 ± 0.021	0.892 ± 0.017
+MVA	1.092 ± 0.029	0.950 ± 0.021	0.864 ± 0.017
+MMR	1.292 ± 0.028	1.031 ± 0.021	0.892 ± 0.017
+Greedy	1.688 ± 0.028	1.323 ± 0.021	0.923 ± 0.017
NNCosNgbr	2.312 ± 0.027	1.529 ± 0.020	0.939 ± 0.017
+MVA	1.146 ± 0.029	1.338 ± 0.021	0.859 ± 0.017
+MMR	2.312 ± 0.027	1.559 ± 0.020	0.939 ± 0.017
+Greedy	2.356 ± 0.027	1.578 ± 0.021	0.939 ± 0.017

Reported results are averages across test folds in a 5-fold cross-validation (at a time, 80% of the data set was used for training and the remaining 20% for testing, and the partitions are chosen at random).

TABLE 3. Lowest average rating for recommendation lists of size $n = 5$, 10, and 20.

	@5	@10	@20
MovieLens-100K			
PureSVD	2.881 ± 0.062	2.404 ± 0.060	2.100 ± 0.059
+MVA	2.870 ± 0.064	2.405 ± 0.061	2.075 ± 0.059
+MMR	2.798 ± 0.063	2.339 ± 0.061	2.037 ± 0.059
+Greedy	3.003 ± 0.065	2.481 ± 0.062	2.123 ± 0.060
NNCosNgbr	2.670 ± 0.066	2.303 ± 0.061	2.035 ± 0.059
+MVA	2.711 ± 0.065	2.290 ± 0.061	2.016 ± 0.059
+MMR	2.663 ± 0.066	2.272 ± 0.061	2.011 ± 0.059
+Greedy	2.812 ± 0.067	2.359 ± 0.061	2.065 ± 0.059
MovieLens-1M			
PureSVD	3.127 ± 0.025	2.624 ± 0.024	2.223 ± 0.023
+MVA	3.129 ± 0.025	2.608 ± 0.024	2.208 ± 0.023
+MMR	3.114 ± 0.025	2.612 ± 0.024	2.209 ± 0.023
+Greedy	3.217 ± 0.025	2.683 ± 0.025	2.263 ± 0.024
NNCosNgbr	2.963 ± 0.025	2.472 ± 0.024	2.117 ± 0.023
+MVA	2.962 ± 0.025	2.457 ± 0.024	2.087 ± 0.023
+MMR	2.953 ± 0.025	2.466 ± 0.024	2.111 ± 0.023
+Greedy	3.057 ± 0.025	2.557 ± 0.024	2.201 ± 0.023
Jester-1			
PureSVD	−3.766 ± 0.053	−5.589 ± 0.046	−6.318 ± 0.041
+MVA	−4.295 ± 0.052	−5.918 ± 0.044	−6.349 ± 0.041
+MMR	−3.766 ± 0.053	−5.590 ± 0.044	−6.319 ± 0.041
+Greedy	−3.270 ± 0.054	−5.231 ± 0.046	−6.285 ± 0.041
NNCosNgbr	−2.178 ± 0.057	−4.777 ± 0.049	−6.257 ± 0.041
+MVA	−4.287 ± 0.056	−5.937 ± 0.048	−6.350 ± 0.041
+MMR	−2.179 ± 0.057	−4.778 ± 0.049	−6.257 ± 0.041
+Greedy	−2.200 ± 0.057	−4.789 ± 0.049	−6.261 ± 0.041

Reported results are averages across test folds in a 5-fold cross-validation (at a time, 80% of the data set was used for training and the remaining 20% for testing, and the partitions are chosen at random).

values are not necessarily statistically different according to a *t*-test. This is usually the case, but if there are some abruptly low or high values in the samples, the averages will likely reflect it, while the *t*-test will remain resilient.

In Tables 2, 3, and 5, rows PureSVD and NNCosNgbr indicate the use of these methods as dependency-agnostic baselines (Top-*n* baselines). +MVA, +MMR, and +Greedy's results are under either PureSVD or NNCosNgbr, depending on which of these two techniques was used for the prediction of their individual scores.

Global analysis. To begin our analysis on Greedy's usefulness, we computed mean ratings obtained with all baselines for recommendation lists with sizes $R = 5$, 10, 20. The results are summarized in Table 2, which presents strong evidence that the exploitation of co-utility alone yields better recommendations than those obtained with competitive dependency-agnostic, Top-*n* baselines. In all cases, either Greedy led to superior mean ratings or it was statistically equivalent to the corresponding Top-*n* results. As to what concerns MVA and MMR, the results indicate that Greedy is likely to recommend items that receive better feedback from users. MVA's mean ratings were particularly low with respect to Jester-1, and the results yielded by MMR were very close to those obtained with Top-*n* baselines. In Table 2, paired *t*-tests were applied to contrast the performance of each method against Greedy. *p*-values are extremely low (ranging from 10^{-11} to 10^{-59}), with exception to NNCosNgbr, +MMR, and +Greedy in the Jester-1 data set.

Note that in Table 2 there are only three underlined values that were statistically equivalent to Greedy when considering the methods NNCosNgBr, +MVA, and +MMR. All other results have shown statistically significant differences. This coheres with the fact that we were taking mean ratings for several users into account—in each test folder of the cross-validation there were 20% of users on average for all data sets, and all of them rated at least 20 items. The samples used in the paired *t*-test were thus significantly large, which helps the paired *t*-test *learn* differences in predictions on a more precise level. Larger data sets such as MovieLens-1M have even more samples for the *t*-tests, which explains why so many values were statistically different for this data set.

Worst-case analysis. It has been suggested that it is worse to recommend an item the user dislikes than to not recommend an item s/he likes (Hansen & Golbeck, 2009). In order to continue our analysis, we exploit this idea by assuming that low ratings are given to disliked items and compare the lowest ratings obtained with different baselines and Greedy. Instead of focusing on all recommended items, this experiment concerns only the worst-rated item in each recommendation.

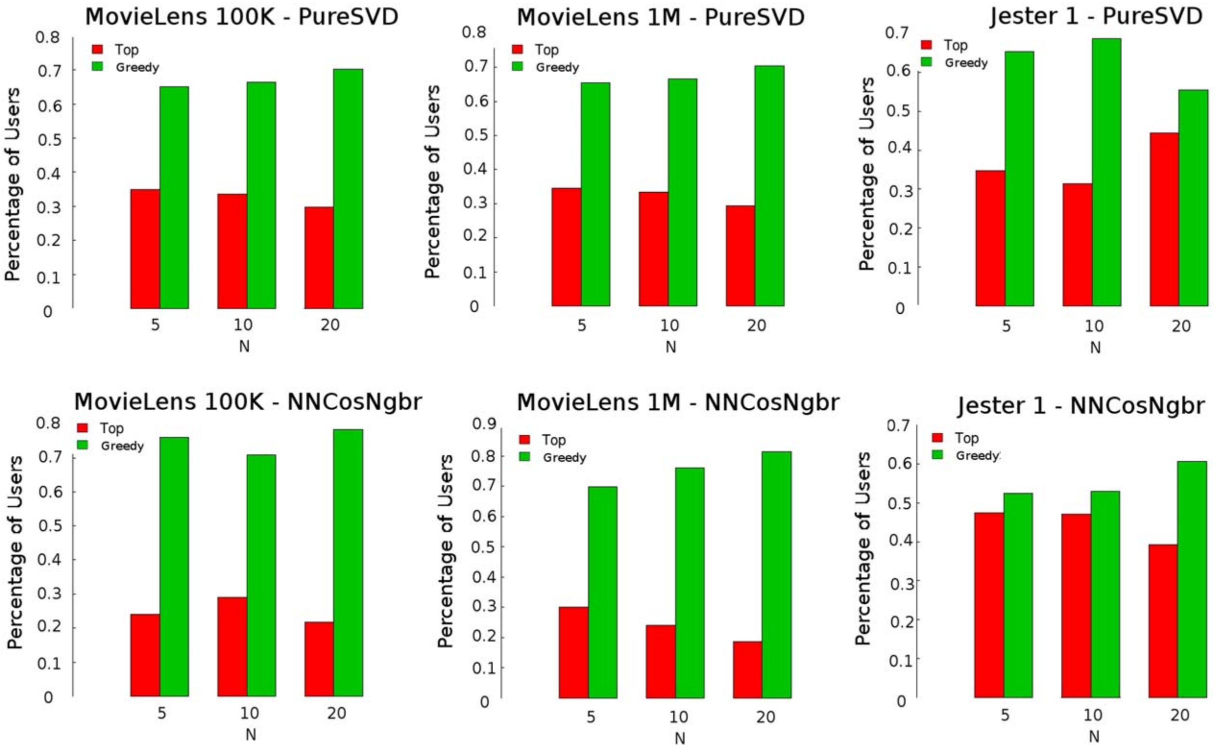

FIG. 3. Percentages of users to which Top-*n* and Greedy have won over each other, in terms of highest mean rating given to generated recommendations. [Color figure can be viewed at wileyonlinelibrary.com]

Results in Table 3 indicate that the worst item recommended by Greedy tends to be better rated than the corresponding one for Top-*n*. In all cases, Greedy leads to superior or statistically equivalent lowest ratings, when compared with both baselines. Underlined results are statistically equivalent to Greedy, and all other results were statistically different, especially for the data set MovieLens-1M. This is a consequence of the large amount of samples used in the *t*-tests (*p*-values ranged from 10^{-5} to 10^{-14}). With respect to the dependency-aware baselines, the worst item recommended by Greedy tends to be better rated than the corresponding ones for MVA and MMR. Once again, values obtained with MMR were somewhat similar to those prompted by Top-*n*. MVA performed better than MMR with respect to the MovieLens data sets and the opposite was noticed with respect to Jester-1. It is interesting to observe that recommendations generated for Jester-1 with NNCosNgbr were particularly similar for Greedy and all baselines, with equivalent mean ratings for almost all comparisons. As for Jester-1, NNCosNgbr yields somewhat high lowest ratings in scenarios of low sparsity, very close to the lowest ratings produced by Greedy.

Win-loss analysis. Figures 3, 4, and 5 lead to a succinct Win-Loss analysis for Greedy, which generates the highest mean ratings, to ~65% of users when compared to Top-*n* baselines. The percentages associated with Greedy tend to increase as *n* grows, which suggests that our method brings gain to more users when more recommendations are generated. Greedy wins over MVA for ~65% of users, and it is more effective when the adopted predictor is NNCosNgbr. With respect to MMR, Greedy outperforms it for ~65% of users as well. Nonetheless, the results varied more for MMR: in the graph associated with Jester-1 and NNCosNgbr, in particular, it won over Greedy for ~48% of users. The percentages associated with Greedy tended to increase as *n* grew as well, as shown in Figure 3.

It is important to compare the results presented in Tables 2 and 3 to further our understanding of the Win-Loss results, in particular for the Jester-1 data set. As indicated in Table 2, the differences between average mean ratings tend to be lower when *n* increases for Jester-1 when we contrast Greedy with the baselines. This can also be observed in Table 3, and although we did not perform an analysis for average highest mean ratings, we believe it would have followed a similar pattern. This likely closer gap between highest ratings for Jester-1 when *n* increases somewhat reflects the Win-Loss analysis: as they become more similar, the methods yield more similar Win-Loss proportions. Note that differences between average mean ratings and average lowest mean ratings in Tables 2 and 3 do not change much when *n* increases for the MovieLens data sets, which probably leads to a more uniform increase in differences for highest ratings and wins/losses between Greedy and the studied baselines.

Recalling Q1, the results indicate that Greedy consistently generates recommendations that are more useful than those produced by the studied baselines. Greedy was particularly better than MVA. In general, absolute gains were

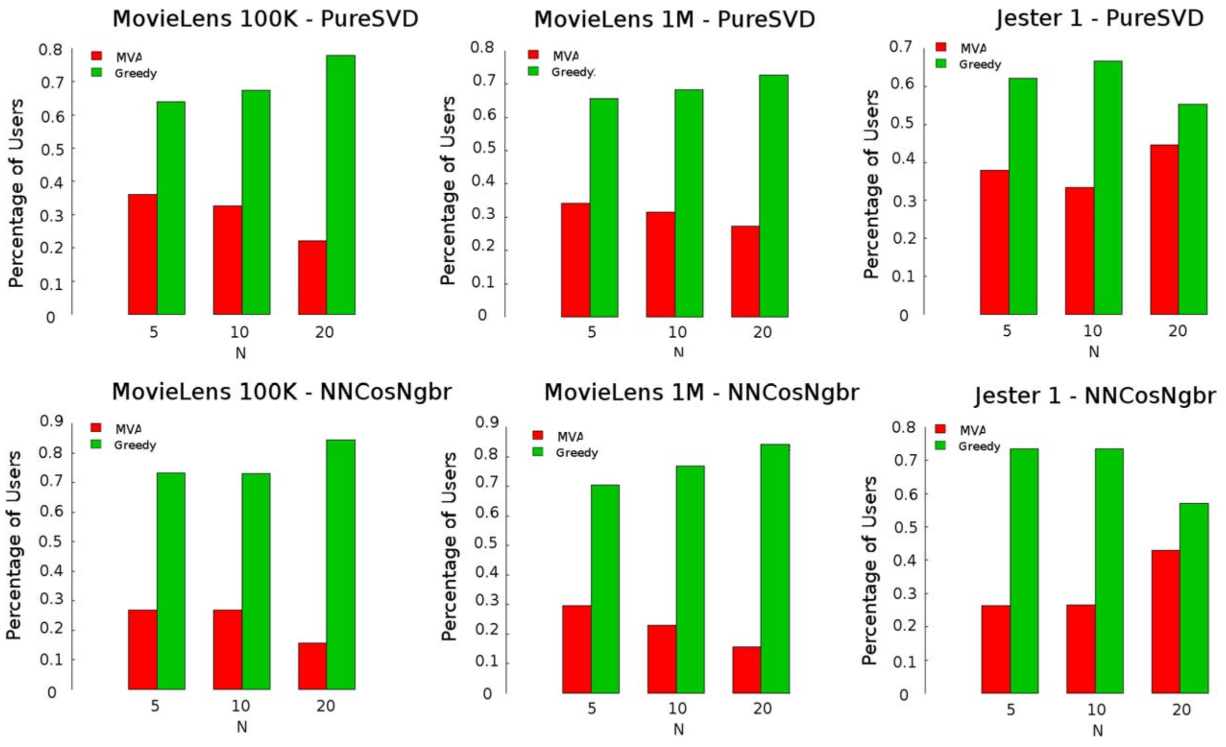

FIG. 4. Percentages of users to which MVA and Greedy have won over each other, in terms of highest mean rating given to generated recommendations. [Color figure can be viewed at wileyonlinelibrary.com]

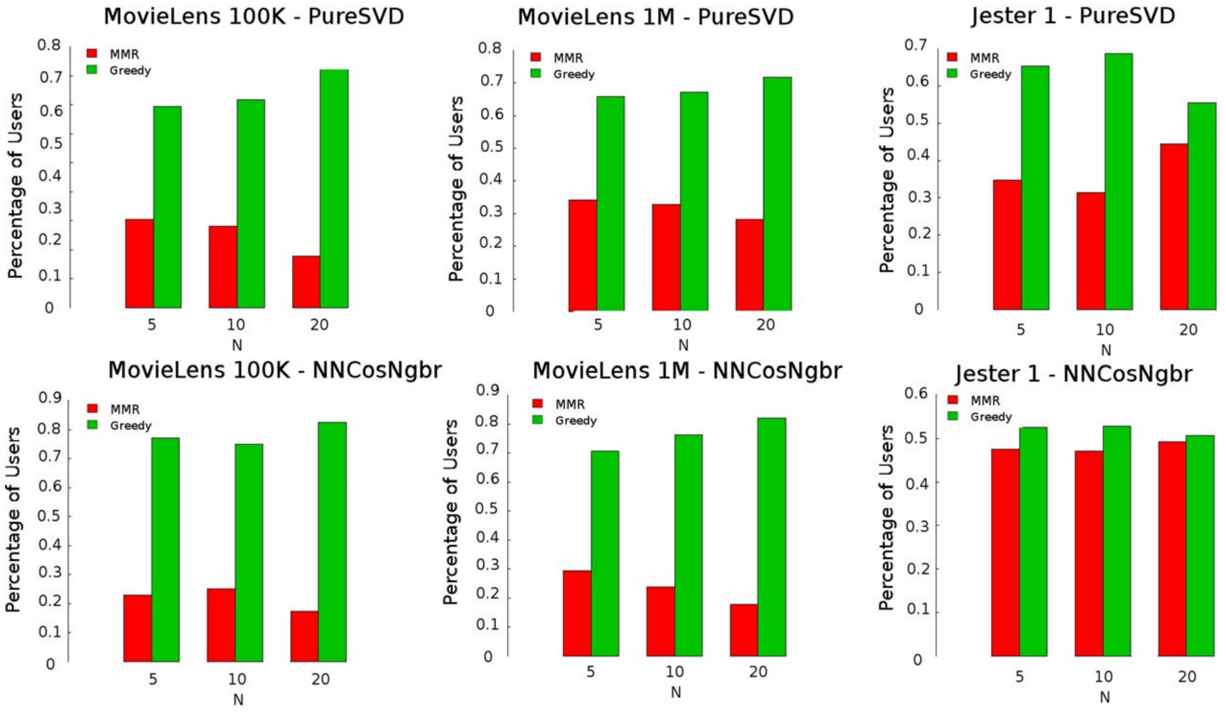

FIG. 5. Percentages of users to which MMR and Greedy have won over each other, in terms of highest mean rating given to generated recommendations. [Color figure can be viewed at wileyonlinelibrary.com]

much higher for the Jester-1 data set. Despite that, the gains obtained with Greedy were consistent for both data sets even when they were small.

Relating Co-Utility and Diversity

In this section we address research question Q2: How diverse are our produced recommendations? In order to do

TABLE 4. Top 10 pairs of movies with highest co-utility probabilities, computed with Empirical Bayes. Alongside these pairs, we list the genres each pair has in common.

Pairs of movies		Common genres
Seven Samurai	*Sanjuro*	Action
The Boat	Sanjuro	Action
GoodFellas	Sanjuro	None
Casablanca	Sanjuro	None
The Wrong Trousers	A Close Shave	Animation/Comedy
The Great Escape	Sanjuro	Adventure
The Wrong Trousers	Wallace & Gromit	Animation
Yojimbo	The Bridge on the River Kwai	Drama
Yojimbo	A Clockwork Orange	None
When We Were Kings	Star Wars Episode IV	None

TABLE 5. Lowest average rating for recommendation lists of size $n = 5$, 10, and 20.

	MovieLens-1M		
	@5	@10	@20
PureSVD	0.8305	0.8392	0.8444
+MMR	0.8309	0.8395	0.8445
+Greedy	0.8302	0.8392	0.8444
NNCosNgbr	0.8360	0.8429	0.8458
+MMR	0.8365	0.8428	0.8458
+Greedy	0.8358	0.8429	0.8458

Reported results are averages across test folds in a 5-fold cross-validation (at a time, 80% of the data set was used for training and the remaining 20% for testing, and the partitions are chosen at random).

so, we investigate whether items that are co-useful have similar content. We also compare the level of diversity in recommendations generated by Greedy, Top-*n*, and MMR. We opted to contrast Greedy with Top-*n* because it is dependency-agnostic, and thus we can analyze whether the pairwise scores θ would hamper the diversity of recommendations generated by predictors PureSVD and NNCosNgbr. As for MMR, we wanted to understand how its results differ from those prompted by Greedy and Top-*n*—two methods that do not focus on diversity. Finally, we analyze if pairwise scores θ—that is, co-utility probabilities—correlate with the cosine similarity.

There are several methods to measure recommendations' diversity (Vargas & Castells, 2011; Ricci et al., 2011). In our scenario, it is important to choose a method that explores item content, as this information was not used by any of the studied algorithms. The use of content dissimilarities allows more impartial comparisons among these algorithms—especially with respect to MMR, as it already embeds rating distributions' dissimilarities in its optimization.

An advantage of exploring content instead of rating distributions is the interpretability of diversity results. For instance, stating that two books are different because their genres and authors are not the same is more interpretable than affirming it because their ratings are not alike. Most experiments in this subsection are exclusive to the MovieLens-1M data set because it has a large amount of content information with which we can compute movie dissimilarities, in contrast to the Jester-1 data set. To have an intuition about whether co-useful items share the same genres, we listed the pairs of movies with highest co-utility probabilities alongside their genres in common in Table 4.

Table 4 indicates that movies that are highly co-useful to users are not necessarily similar in terms of genres. In particular, 4 out of 10 pairs of movies have no genre in common. Although there may be other sources of similarities that we did not consider, such as actors in common, these results strengthen the hypothesis that co-utility does not imply similarity. For instance, *When We Were Kings* is a documentary released in the 1990s, whereas *Star Wars Episode 4* is a Sci-Fi/Adventure movie from 1977. It is important to note that most of these movies are very popular and received high ratings in websites such as IMDb[7] and Rotten Tomatoes.[8]

To further our understanding on how co-utility may relate to diversity, we aggregated the diversity levels of recommendations generated with Greedy, Top-*n*, and MMR in Table 5. To compute movie dissimilarities, we calculated Jaccard's coefficient over movies' corresponding genres. To compare the diversity levels of Greedy, Top-*n*, and MMR, we used the ILD metric described in "Validation Metrics" section.

We performed paired *t*-tests for each Top-*n*/Greedy and MMR/Greedy pair in Table 5 with a 95% confidence interval, and none of the results were statistically different. Recalling Q2, the results indicate that Greedy is not likely to hurt recommendations' diversity when compared to Top-*n* and may generate as diversified recommendations as the MMR algorithm implemented with the Kullback–Leibler divergence.

Recommendation Optimality

In this section we used experiments that aim to answer the research question Q3: How far are our produced recommendations from those obtained with an optimal solution to MSDP? To contrast Greedy and optimal solutions obtained with an optimizer, namely, Exact, we divided the data sets into 10 folds with approximately the same size randomly. In all cases, one of the folds was chosen for test and the others were used for training. We did not perform cross-validation in this experiment due to time constraints. Also, because of time constraints, experiments with the MovieLens data sets only use predictor PureSVD, and experiments with Jester-1 only use predictor NNCosNgbr.[9]

To compute the exact solutions and perform comparisons that make sense in practical, real-time scenarios, we adopted a timeout of 20 seconds. We discarded all exact solutions that would take more than that, and only compared optimal

[7] http://www.imdb.com/

[8] http://www.rottentomatoes.com/

[9] In preliminary experiments, predictor PureSVD has yielded the best results for MovieLens-1M and NNCosNgbr has yielded the best results for Jester-1.

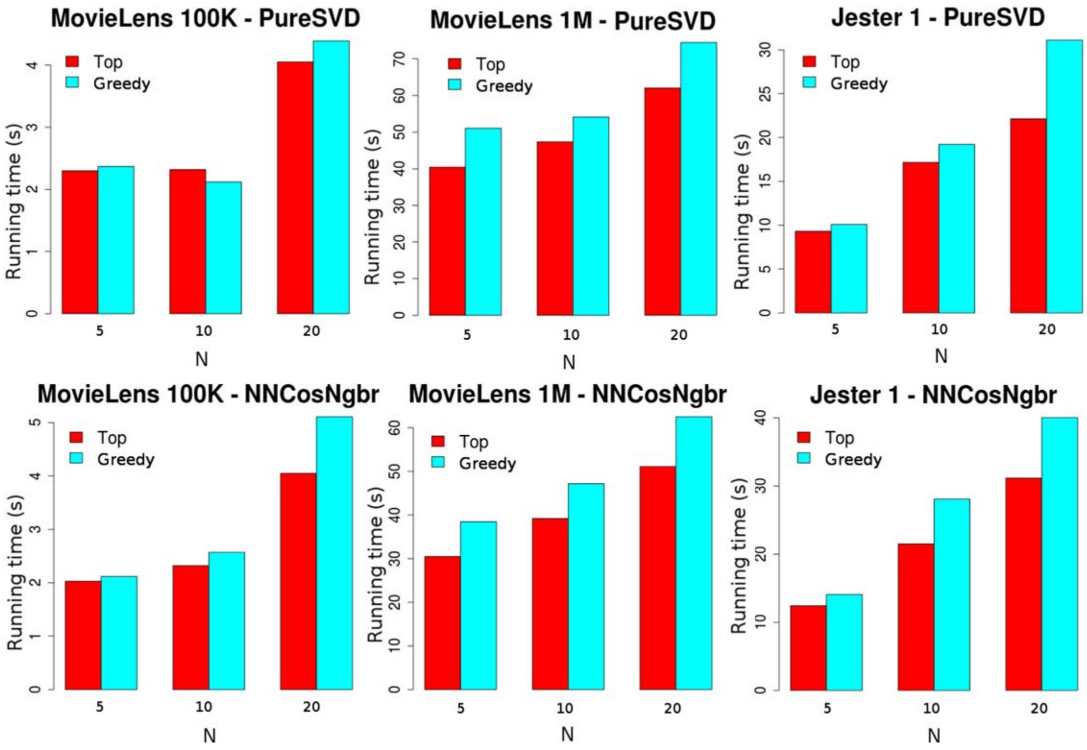

FIG. 6. Mean running times per validation fold, in seconds, for different combinations of data sets and predictors, with $N = 5$, 10, 20. Reported results are averages across test folds in a 5-fold cross-validation (at a time, 80% of the data set was used for training and the remaining 20% for testing, and the partitions are chosen at random). [Color figure can be viewed at wileyonlinelibrary.com]

solutions obtained below this time threshold with their corresponding suboptimal ones. Solutions that take more than 20 seconds to compute corresponded to less than 15% of all cases. We performed a paired *t*-test with a 95% confidence interval over the mean ratings obtained with Greedy and Exact and all results were statistically equivalent.

Recalling Q3, the results indicate that, in practice, Greedy is a good approach to MSDP. A case where Greedy and Exact lead to different solutions works as follows. Let us consider a set of candidate items $I=\{i_1, i_2, i_3\}$ with corresponding sets of scores $I_\phi=\{\phi_{i_1}=0.9, \phi_{i_2}=0.85, \phi_{i_3}=0.85\}$ and $I_\theta=\{\theta_{i_1 i_2}=0.7, \theta_{i_1 i_3}=0.6, \theta_{i_2 i_3}=0.9\}$. If we want to select $N=2$ items out of I, Exact will select i_2 and i_3, whereas Greedy will select i_1 and i_2. The Greedy choice of starting the selection by choosing the item with the highest score ϕ does not necessarily lead to the optimal solution, as the example illustrates.

Analyzing the Scalability of Our Method

In this section we examine research question Q4: How scalable is our method? Although Greedy is polynomial and rather fast, there are some easy and important optimizations that make it scalable and competitive in practice. It is important, for example, to precompute and store all pairwise scores θ in a hash table as a preprocessing step. This offline computation speeds up the generation of solutions to MSDP by avoiding redundant computations of pairwise scores. Another improvement involves the use of memorization to reuse partial summations. Figure 6 illustrates the mean computation time per validation fold for each data set, varying n and the predictor algorithm. All experiments were performed with a Pentium Dual-Core 2.0GHz with 2GB RAM. We decided to compare Greedy with Top-n because, from all studied methods, Top-n is the fastest one in practice.[10]

The results in Figure 6 correspond to the mean aggregated running time for the generation of all recommendation lists concerning a validation fold. These results provide an answer to Q4: yes, Greedy is quite scalable in practice.

For higher values of n, the time difference between Greedy and Top-n could increase, but such analysis is not useful in real-world scenarios because n values are not high in practice (Ricci et al., 2011). Therefore, for realistic values of n (i.e., most recommendation applications use lists with fewer than 50 items), Greedy scales well and its mean running times per validation fold are only slightly worse than those obtained with Top-n. In spite of that, the time difference for generating a single recommendation list with all methods is irrelevant. Given that in real-world systems recommendation lists are generated one at a time via the interaction with users, Greedy is a feasible alternative.

[10] We used a Top-n implementation with time complexity $O(K \log N)$.

Conclusions and Future Work

In this article we investigated how co-utility probabilities can be estimated and exploited in order to improve the utility of recommended items. To this end, we modeled the interplay of individual predictions and co-utility probabilities as a linear combination. Afterwards, we posed the task of finding the best subset of candidate recommendations, which mapped trivially into the Max-Sum Dispersion Problem (MSDP). We implemented a scalable, Greedy heuristic to MSDP, and evaluated it using three publicly available recommendation data sets.

To demonstrate the usefulness of our approach, we have shown that it performs consistently better than two state-of-the-art dependency-agnostic recommendation baselines. Moreover, by contrasting our approach to dependency-aware, diversity-oriented baselines, we have shown that the exploitation of co-utility probabilities does not necessarily hurt recommendations' diversity. Furthermore, by contrasting our Greedy heuristic to an optimal solution to MSDP, we have shown that our produced recommendations are not statistically different than those obtained via an exact optimization. Finally, by comparing the running times of our Greedy heuristic and the dependency-agnostic baselines, we have demonstrated the scalability of our approach and its applicability for real-world recommendation scenarios.

Throughout this article, we showed that co-utility probabilities are important evidence for recommender systems. Hence, in the future we intend to develop Learning to Rank algorithms that embed co-utility estimates as learning features. We also want to extend our method to hybrid recommenders, by exploiting content information in order to compute co-utility probabilities. Finally, we believe that there is still room for improvement in the choice of θ_{ij}, since different persons/users may have different probability of items. Thus, we plan to study approaches to set θ_{ij} values according to the probability of items related to each user.

Acknowledgments

We thank the partial support given by the Brazilian National Institute of Science and Technology for the Web (grant MCT-CNPq 573871/2008-6), Project Models, Algorithms and Systems for the Web (grant FAPEMIG/PRONEX/MASWeb APQ-01400-14), and the authors' individual grants and scholarships from CNPq and Kunumi.

References

Adomavicius, G., & Tuzhilin, A. (2005). Towards the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. Transactions on Knowledge and Data Engineering, 17, 734–749.

Bishop, C.M. (2006). Pattern recognition and machine learning. New York: Springer.

Bookstein, A. (1983). Information retrieval: A sequential learning process. Journal of the American Society for Information Science, 34, 331–342.

Borodin, A., Lee, H.C., & Ye, Y. (2012). Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of the 31st Symposium on Principles of Database Systems*, Scottsdale, USA (pp. 155–166).

Breese, J.S., Heckerman, D., & Kadie, C. (1998). Empirical analysis of predictive algorithm for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, Madison, USA (pp. 43–52).

Carbonell, J., & Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st ACM SIGIR Conference on Research and Development in Information Retrieval*, Melbourne, Australia (pp. 335–336).

Casella, G. (1992). Illustrating Empirical Bayes methods. Chemometrics and Intelligent Laboratory Systems, 16, 107–125.

Cremonesi, P., Koren, Y., & Turrin, R. (2010). Performance of recommender algorithms on Top-n recommendation tasks. In *Proceedings of the 4th ACM Conference on Recommender Systems*, Barcelona, Spain (pp. 39–46).

Elton, E.J., Gruber, M.J., Brown, S.J., & Goetzmann, W.N. (2009). Modern portfolio theory and investment analysis. Hoboken, NJ: Wiley.

Gollapudi, S., & Sharma, A. (2009). An axiomatic approach for result diversification. In *Proceedings of the 18th International Conference on World Wide Web*, Madrid, Spain (pp. 381–390).

Hansen, D.L., & Golbeck, J. (2009). Mixing it up: Recommending collections of items. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems*, Boston, USA (pp. 1217–1226).

Hastie, T., Tibshirani, R., & Friedman, J. (2001). The elements of statistical learning. New York: Springer.

Liu, J., & Belkin, N.J. (2015). Personalizing information retrieval for multi-session tasks: Examining the roles of task stage, task type, and topic knowledge on the interpretation of dwell time as an indicator of document usefulness. Journal of the American Society for Information Science and Technology, 66, 58–81.

Ravi, S., Rosenkrantz, D., & Tayi, G. (1994). Heuristic and special case algorithms for dispersion problems. Operations Research, 42, 299–310.

Ribeiro, M.T., Ziviani, N., de Moura, E.S., Hata, I., Lacerda, A., & Veloso, A. (2014). Multiobjective pareto-efficient approaches for recommender systems. ACM Transactions on Information Science and Technology, 5, 53:1–53:20.

Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (Eds). (2011). Recommender systems handbook. New York: Springer.

Robertson, S.E. (1977). The probability ranking principle in information retrieval. Journal of Documentation, 33, 294–304.

Santos, R.L.T., Macdonald, C., & Ounis, I. (2010). Exploiting query reformulations for web search result diversification. In *Proceedings of the 19th International Conference on World Wide Web*, Raleigh, USA (pp. 881–890).

Santos, R.L.T., Macdonald, C., & Ounis, I. (2015). Search result diversification. Foundations and Trends in Information Retrieval, 9, 1–90.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International World Wide Web Conference*, Hong Kong, Hong Kong (pp. 285–295).

Sheffet, O., Mishra, N., & Ieong, S. (2012). Predicting consumer behavior in commerce search. In *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland (pp. 1767–1774).

Tversky, A. (1972). Elimination by aspects: A theory of choice. Psychological Review, 79, 281–299.

Vargas, S., & Castells, P. (2011). Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the 5th ACM Conference on Recommender Systems*, Chicago, USA (pp. 109–116).

Vieira, M.R., Razente, H.L., Barioni, M.C.N., Hadjieleftheriou, M., Srivastava, D., Jr, Caetano, T., & Tsotras, V.J. (2011). On query result diversification. In *Proceedings of the 27th International Conference on Data Engineering*, Hannover, Germany (pp. 1163–1174).

Wang, J. (2009). Mean-variance analysis: A new document ranking theory in information retrieval. In *Proceedings of the 31st European Conference on Information Retrieval*, Toulouse, France (pp. 4–16).

Weston, J., & Blitzer, J. (2012). Latent structured ranking. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, Catalina Island, USA (pp. 903–913).

Xiong, C., T., Wang, Wenkui Ding, Y.S., & Liu, T.Y. (2012). Relational click prediction for sponsored search. In *Proceedings of the 5th International Conference on Web Search and Web Data Mining*, Seattle, USA (pp. 493–502).

Zhang, M., & Hurley, N. (2008). Avoiding monotony: Improving the diversity of recommendation lists. In *Proceedings of the 2nd ACM Conference on Recommender Systems*, Lausanne, Switzerland (pp. 123–130).

Zuccon, G., Azzopardi, L., Zhang, D., & Wang, J. (2012). Top-k retrieval using facility location analysis. In *Proceedings of the 34th European Conference on Information Retrieval*, Barcelona, Spain (pp. 305–316).