Improving the Recommendation of Given Names by Using Contextual Information

Marcos Aurélio Domingues[1], Ricardo Marcondes Marcacini[2], Solange Oliveira Rezende[1], and Gustavo E. A. P. A. Batista[1]

[1] Institute of Mathematics and Computer Science – University of São Paulo
Av. Trabalhador São-Carlense, 400, Cx. Postal 668, 13560-970, São Carlos, SP, Brazil
mad@icmc.usp.br, solange@icmc.usp.br, gbatista@icmc.usp.br
http://labic.icmc.usp.br
[2] Federal University of Mato Grosso do Sul
Três Lagoas, MS, Brazil
ricardo.marcacini@ufms.br

Abstract. The people who have to choose a given name, know how challenging it is to find a suitable name that fits the social context, the language, the cultural background, and especially, the personal taste. For example, future parents end up browsing through several lists of given names in order to choose a name for their unborn child. A recommender system can help a person in this task by recommending given names which are of interest to the user. In this paper, we exploit contextual information (e.g., time and location) in two state-of-the-art recommender systems for the task of recommending given names. The empirical results have shown that we can improve the recommendation of given names by using contextual information.

Key words: Given Names, Recommender Systems, Item-based Collaborative Filtering, Association Rules, Contextual Information

1 Introduction

The task of finding a suitable name for an unborn child is not so easy. Future parents usually face many books or web sites, listing given names, in order to find a suitable name for their child. Here, a suitable name is represented by a given name which satisfies a set of factors such as the social context, the language, the cultural background, and especially, the personal taste. Although this task is relevant in practice, little research has been performed on the analysis and application of interrelations among given names from a recommender system perspective. A recommender system is an information filtering technology which can be used to output an ordered list of items (e.g., given names) that are likely to be of interest to the user [1, 2].

In different scenarios, recommender systems are subject to scientific research, as, for example, recommending products [3], mobile applications [4], places of interest [5], movies [6], music [7]. In this paper, we exploit two state-of-the-art

recommender systems for the task of recommending given names. In addition, we also incorporate contextual information (e.g., time and location) in such systems in order to capture tendencies in choosing given names and, thus, to improve the recommendations.

The paper is organized as follows: In Section 2 we present some work related to the recommendation of given names. In Section 3 we describe the two state-of-the-art recommender systems used in this work and an approach to incorporate contextual information in these systems. We present our empirical evaluation in Section 4. We discuss the data set, the pre-processing of the data, the experimental setup and evaluation metric, and the empirical results. In Section 5 we show our contribution to the *15th Discovery Challenge: Recommending Given Names*. In Section 6 we present final remarks.

2 Related Work

In this section, we present some work related to the recommendation of given names. A search engine and recommender system for given name is described in [8]. This system, called *Nameling*[3], can be used by users to find a suitable given name. For example, the user enters a given name and obtains a browsable list of names.

In [9], the authors analyze the co-occurence of names from the *Nameling* system. They show how basic approaches from the field of social network analysis and information retrieval can be applied for discovering relations among names. In [10], a recommendation method for given names, based on co-occurence within Wikipedia[4], is proposed. The method, called preferential PageRank, is a modification of the well known PageRank algorithm [11]. There, the preferential PageRank method is evaluated by using a data set from the *Nameling* system.

An empirical evaluation comparing the preferential PageRank method against some state-of-the-art recommender systems, for the task of recommending given names, is presented in [12]. By using a data set from the *Nameling* system, the authors compare the user-based collaborative filtering [13], the item-based collaborative filtering [14], the weighted matrix factorization method [15], the most popular recommendation approach (that recommends the most popular names), and the random approach (that recommends names randomly) against the preferential PageRank method [10]. The results show that the preferential PageRank method provides good performance in terms of prediction accuracy as well as runtime complexity.

Our contribution for the *Nameling* system consists in exploiting the contextual information (i.e., the month and the city) contained in the data from the system to capture tendencies in choosing given names, and, thus, to improve the recommendations. Our proposal will be described in the next sections.

[3] `http://nameling.net/`
[4] `http://www.wikipedia.org/`

3 Recommender Systems

A recommender system for the web is an information filtering technology which can be used to predict preference ratings of items (e.g., movies, music, books, news, images, web pages, given names, etc) not currently rated by the user [16], and/or to output a set of items/recommendations that are likely to be of interest to the user [1, 2].

We focus our work on the task of selecting the top-N items/recommendations which are of interest to a user. We formalize this task as follows:

> Let p be the number of users $U = \{u_1, u_2, ..., u_p\}$ and q the number of all possible items that can be recommended $I = \{i_1, i_2, ..., i_q\}$. Now, let j be the number of historical sessions in a web site $S = \{s_1, s_2, ..., s_j\}$. Each session $s = \langle u, I_s \rangle$ is a tuple defined by a user $u \in U$ and a set of accessed items $I_s \subseteq I$. The set S is used to build a top-N recommendation model M.
> Given an active session s_a defined by an active user u_a and a set of observable items $O \subset I$, the recommendation model M uses the set O to identify the interest of the user u_a and recommend N items from the set of items/recommendations R, such that $R \subset I$ and $R \cap O = \varnothing$, that are believed to be the top preferences of the user u_a.

In this section, we present two state-of-the-art recommender systems: Item-based Collaborative Filtering and Association Rules based. In this work we use these systems for recommending given names. In addition, we present an approach which is used to incorporate contextual information in the two state-of-the-art systems in order to generate context-aware recommendations.

3.1 Item-Based Collaborative Filtering

The Item-based Collaborative Filtering technique analyzes items to identify relations among them [17]. Here, the recommendation model M is a matrix representing the similarities between all the pairs of items, according to a given similarity metric. An abstract representation of a similarity matrix is shown in Table 1. Each item $i \in I$ is an accessed item, for example, a given name.

Table 1. Item-item similarity matrix

	i_1	i_2	$\cdots$	i_q
i_1	1	$sim(i_1, i_2)$	$\cdots$	$sim(i_1, i_q)$
i_2	$sim(i_2, i_1)$	1	$\cdots$	$sim(i_2, i_q)$
$\cdots$	$\cdots$	$\cdots$	1	$\cdots$
i_q	$sim(i_q, i_1)$	$sim(i_q, i_2)$	$\cdots$	1

According to [17], the properties of the model and consequently the effectiveness of this recommendation algorithm depend on the method used to calculate

the similarity among the items. To calculate the similarity between pairs of items, for example, i_1 and i_2, we first isolate the users who have rated both of these items, and then, we apply a metric on the ratings to compute the similarity $sim(i_1, i_2)$ between i_1 and i_2. Metrics to measure the similarity between pairs of items are cosine angle, Pearson's correlation and adjusted cosine angle. In this paper, we use the cosine angle metric, defined as

$$sim(i_1, i_2) = cos(\overrightarrow{i_1}, \overrightarrow{i_2}) = \frac{\overrightarrow{i_1} . \overrightarrow{i_2}}{||\overrightarrow{i_1}|| * ||\overrightarrow{i_2}||}, \tag{1}$$

where $\overrightarrow{i_1}$ and $\overrightarrow{i_2}$ are rating vectors with as many positions as existing users in the set U. The operator "." denotes the dot-product of the two vectors. In our case, the rating vectors are binary. The value 1 means that the users accessed the respective item. The value 0 has the opposite meaning.

Once we obtain the recommendation model, we can generate the recommendations. Given an active session s_a containing a user u_a and its set of observable items $O \subseteq I$, the model generates the N recommendations as follows. First, we identify the set of candidate items for recommendation C by selecting from the model all items $i \notin O$. Then, for each candidate item $c \in C$, we calculate its similarity to the set O as

$$sim_{c,O} = \frac{\sum_{i \in K_c \cap O} sim(c, i)}{\sum_{i \in K_c} sim(c, i)}, \tag{2}$$

where K_c is a set with the k most similar items (the nearest neighbors) to the candidate item c.

Finally, we select the top-N candidate items with the highest similarity to the set O and recommend them to the user u_a.

3.2 Association Rules Based

A recommendation model M based on association rules is a set of rules. Each rule m has the form $m : X \rightarrow Y$, where $X \subseteq I$ and $Y \subseteq I$ are sets of items and $X \cap Y = \varnothing$. Here, we generate association rules with one single item in the consequent of the rule [18], i.e., Y is a singleton set. Each association rule is characterized by two metrics: support and confidence [19].

The support of a rule in a set of sessions S is defined as

$$support(X \rightarrow Y) = \frac{|X \cup Y|}{|S|}, \tag{3}$$

where $|X \cup Y|$ is the number of sessions in S that contain all items in $X \cup Y$ and $|S|$ is the number of sessions in S.

The confidence of a rule is the proportion of the number of sessions which contain $X \cup Y$ with respect to number of sessions that contain X, and can be formulated as

$$confidence(X \rightarrow Y) = \frac{|X \cup Y|}{|X|}. \quad (4)$$

Discovering all association rules from a set of sessions S consists in generating all rules whose support and confidence are greater than or equal to the corresponding minimal thresholds, called *minsup* and *minconf*. The classical algorithm for discovering association rules is Apriori [19].

To build the recommendation model M using association rules, the set of sessions S is used as input to an association rules algorithm to generate a set of rules. Once we have the model, we can make recommendations, R, to a new session. Given an active session s_a containing a user u_a and its set of observable items O, we build the set R as follows [18]:

$$R = \{consequent(m) | m \in M \text{ and } antecedent(m) \subseteq O \text{ and } consequent(m) \notin O\}. \quad (5)$$

To obtain the top-N recommendations, we select from R the N distinct recommendations corresponding to the rules with the highest confidence values.

3.3 The Weight Post-Filtering Approach

There are many definitions of context in the literature depending on the field of application and the available customer data [2]. For example, in [20], context is defined as any information that can be used to characterize an item. In this paper, we use time and location (i.e., month and city) as context to identify tendencies in the choice of a given name to improve the recommendations.

To incorporate contextual information in the previous recommender systems, we have extended the Weight Post-Filtering (PoF) approach, proposed in [21], for the task of item recommendation. The original approach was proposed for rating prediction [21].

The Weight PoF approach first ignores the contextual information in the data (in our case, the month and the city from each access) and applies a traditional algorithm (e.g., Item-based Collaborative Filtering or Association Rules based) to build the recommendation model. Then, it computes the probabilities of user's access items under a given context. The probability $P_c(u, i)$ that a user u accesses an item i under the context c can be computed as follows

$$P_c(u, i) = \frac{Num_c(u, i)}{Num(u, i)}, \quad (6)$$

where $Num_c(u, i)$ is the number of users that, like the user u, also accessed the item i under the context c; and $Num(u, i)$ is the total number of users that accessed the item i.

The score of the items computed by using the previous recommender systems are multiplied by the probabilities $P_c(u, i)$, incorporating context into the

recommendations and improving the performance of the recommender systems. Finally, the items are reordered and the top-N items are recommended to the user.

4 Empirical Evaluation

In this section, we empirically evaluate the recommender systems, presented in Section 3, in the task of recommending given names.

4.1 Data Set

The empirical evaluation is carried out using an usage data set from *Nameling*. According to [8], *Nameling* is a search engine and a recommender system for given names. In this system, the user enters a given name and obtains a browsable list of recommended names, called "*namelings*".

The data set is derived from the Nameling query logs, ranging from March 6th, 2012 to February 12th, 2013. It contains 515,848 accesses from 60,922 users to 20,714 different items (i.e., given names). There are five types of accesses/activities[5]:

1. **ENTER_SEARCH:** The user enters a name directly into the Nameling's search field;
2. **LINK_SEARCH:** The user follows a link on some result page;
3. **LINK_CATEGORY_SEARCH:** Wherever available, names are categorized according to the corresponding Wikipedia articles;
4. **NAME_DETAILS:** Users can get some detailed information for a name;
5. **ADD_FAVORITE:** Users can maintain a list of favorite names.

Additionally, for each access there are a timestamp and a proxy for the user's geographic location (i.e., country code, province, city, latitude and longitude) which is obtained by using the MaxMind's GeoLite City data base[6].

As part of the data set, there is also a list of known names[7] containing all names which are currently known in the *Nameling* web site. As we will see in Section 4.3, all names that occur in the evaluation data set are contained in this list of names.

4.2 Pre-processing of the Data Set

Before running the experiments, we pre-processed the data set by replacing invalid names and removing singleton sessions, as described below:

[5] We use the terms access and activity interchangeably.
[6] `http://www.maxmind.com/`
[7] `http://www.kde.cs.uni-kassel.de/nameling/dumps`

Replacing invalid names: In real-world data sets, it is common to find several variations of a name, for example, spelling variations due to typographical errors (like "Richard" and "Ricahrd") and differences in punctuation marks (like "O'Reilly" and "O Reilly"). Considering the existence of a reference list with valid names, we can use string comparison measures to replace an invalid name by the nearest valid name, thus believing that we are correcting a name typed incorrectly. Thus, in the data pre-processing step, we use the list of known names, described in the previous section, and apply the Jaro-Winkler measure [22] for detection and replacement of invalid names. For this purpose, it is necessary first to define the Jaro (j) measure between two strings w_1 e w_2 (Equation 7):

$$j(w_1, w_2) = \begin{cases} 0 & \text{if } h = 0 \\ \frac{1}{3}\left(\frac{h}{|w_1|} + \frac{h}{|w_2|} + \frac{h-t}{h}\right) & \text{otherwise,} \end{cases} \tag{7}$$

where h is the number of matching characters and t represents the number of transpositions. The matching between two characters c_1 and c_2, with $c_1 \in w_1$ and $c_2 \in w_2$ occurs when $c_1 = c_2$ and they are not further than $\frac{\max(|w_1|,|w_2|)}{2} - 1$. The number of transpositions is obtained by considering different orders for matching characters. The Jaro-Winkler (jw) is based on the Jaro measure, according to Equation 8, in which $l(w_1, w_2)$ represents the length of common prefix at the start of the string up to a maximum of 4 characters.

$$jw(w_1, w_2) = j(w_1, w_2) + \frac{l(w_1, w_2)}{4} * (1 - j(w_1, w_2)) . \tag{8}$$

The Jaro-Winkler measure was selected for this work because it shows better performance in studies involving name-matching [23].

Removing singleton sessions: For different reasons, users often access only one item on a web site and then leave it. The use of these sessions containing a singleton access by a recommender system can affect its accuracy negatively [24]. For example, singleton sessions will never count for the item-item similarity in the Item-based Collaborative Filtering technique. Thus, we have removed the singleton sessions from the data set.

After pre-processing the data, we obtained a set with 510,705 accesses from 55,779 users to 20,318 different names.

4.3 Experimental Setup and Evaluation Metric

To carry out the experiments, we use a Perl script[8] to split the data set in training and test sets.

The script selects some users with at least five different names for the test set. Then, for each test user, it withholds the last two entered names for evaluation.

[8] `http://www.kde.cs.uni-kassel.de/nameling/dumps/process_activitylog.pl`

To withhold the last two entered names, the script uses the following rules. For each test user, the script selects for evaluation the last two names which had directly been entered into the Nameling's search field (i.e., ENTER_SEARCH access) and which are also contained in the list of known names. The script only considers those names which were not previously added as a favorite name by the test user (i.e., ADD_FAVORITE access). Finally, the script removes the accesses after the names for evaluation and keeps in the test set only users with at least three accesses. The remaining users in the data set are used as training set.

To evaluate the recommender systems, we compute the metric Mean Average Precision (MAP) [25]. For each test user, the metric takes the left out evaluation names and compute the precision at the respective position in the ordered list of recommended names. These precision values are first averaged per test user and than in total to obtain the final score. Here, we use a Perl script[9] to compute the MAP@1000 which means that only the first 1,000 positions of a list of recommendations are considered.

With respect to the recommendation algorithms, we use the Item-based Collaborative Filtering and the Association Rules based, which were described in Section 3. In the Item-based Collaborative Filtering, the top-N recommendations are generated based on their 1, 5, 10, 15 and 20 most similar items (i.e., the 1, 5, 10, 15 and 20 nearest neighbors). In the Association Rules based algorithm, the recommendation models are built using a minimum support value determined to generate around 10,000; 50,000 and 100,000 rules. The minimum confidence values are defined as being the support value of the one thousandth most frequent item in the training set. This allows the generation of at least 1,000 rules without antecedent that can be used by default, in the case that no other rule applies. Here, as the left out names for evaluation are only names which had been directly entered into the Nameling's search field (i.e., ENTER_SEARCH access), we have selected only this type of access from the training set to build the recommendation models.

Finally, we use the month and the city from the accesses as contextual information in the Weight PoF approach, as described in Section 3.3. Such information can capture tendencies of names in a given city, in a given month. The month is obtained from the timestamp of the access. We obtain the city by using the proxy for the user's geographic location provided with the data set.

4.4 Empirical Results

We start by comparing the Item-based Collaborative Filtering technique (CF) against its contextual version that makes use of the Weight PoF approach (CF-PoF). In Table 2, we see that the values of MAP@1000 decrease when we increase the number of neighbors. This fact occurs because when we increase the number of neighbors, less similar items are used to generate the recommendations. Comparing the CF-PoF against the CF, we see an improvement of the

[9] `http://www.kde.cs.uni-kassel.de/nameling/dumps/evaluate_recommender.pl`

recommendations by using the contextual information. The CF-PoF algorithm provides gains of MAP@1000 ranging from 6.9% to 22.6%.

Table 2. Comparing the MAP@1000 values between CF and CF-PoF algorithms

K-neighbors	CF	CF-PoF
K = 1	0.0092	0.0099
K = 5	0.0038	0.0042
K = 10	0.0033	0.0039
K = 15	0.0031	0.0038
K = 20	0.0029	0.0031

In Table 3, we can see that the difference between the Association Rules based algorithm (AR) and its, respective, contextual version (AR-PoF) is quite small. In this case, the AR-PoF algorithm provides gains of MAP@1000 around 2.4%.

Table 3. Comparing the MAP@1000 values between AR and AR-PoF algorithms

Number of Rules	AR	AR-PoF
10,000	0.0337	0.0343
50,000	0.0314	0.0321
100,000	0.0290	0.0299

We also compare the results between both Tables 2 and 3. We see that the AR-PoF recommender system with 10,000 rules provides the best value for MAP@1000, i.e., 0.0343. If we compare this value against the one provided by the best recommender system in Table 2, the CF-PoF with K = 1, we see a gain of 246.5%. Besides, our Association Rules based algorithms are quite fast. We measured the computational time to build the recommendation model and generate the 1,000 recommendations. In our experimental scenario, we used an Intel Core i7 Ivy Bridge with a CPU clock rate of 3.4 GHZ, 32 GB of main memory, and running the Debian Linux operating system. To build a recommendation model, the algorithms take around 26 seconds (10,000 rules) to 2 minutes (100,000 rules). Here, the top-1000 recommendations are generated in approximately 1 second.

5 The 15th Discovery Challenge: Recommending Given Names

After analyzing the results presented in Section 4.4, we applied our best scenario to the data set from the *15th Discovery Challenge: Recommending Given Names*.

We pre-processed the data set, selected only names entered into the Nameling's search field, and then ran the algorithm which provided the best MAP@1000, i.e., the AR-PoF algorithm with about 10,000 association rules. With this scenario, our **Labic** team obtained a score of 0.0379 in the final leaderboard.

6 Final Remarks

Although the task of recommending given names is relevant in practice, little research has been performed on the perspective of recommender systems. In this paper, we exploited two state-of-the-art recommender systems in the task of recommending given names. In addition, we also incorporated contextual information in such systems to capture tendencies in choosing given names and, thus, to improve the recommendations. Although the gains obtained by using the Weight PoF approach are small, the results of our empirical evaluation present evidences that we can improve the recommendation of given names by using contextual information.

There are some directions to be explored in future research. For example, other pre-processing tasks can be applied on the data set in order to improve the quality of the data. We can also try other context-aware recommender systems in the task of recommending given names [26, 27]. On the other hand, we can also combine the two state-of-the-art recommenders, presented in this paper, in a hybrid algorithm.

Acknowledgments. This work was supported by the grants 2010/20564-8, 2011/19850-9, 2012/13830-9, 2012/07295-3, São Paulo Research Foundation (FAPESP).

References

1. Resnick, P., Varian, H.R.: Recommender systems. Communications of the ACM **40**(3) (1997) 56–58
2. Ricci, F., Rokach, L., Shapira, B., Kantor, P.B., eds.: Recommender Systems Handbook. Springer (2011)
3. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing **7**(1) (2003) 76–80
4. Karatzoglou, A., Baltrunas, L., Church, K., Böhmer, M.: Climbing the app wall: enabling mobile app discovery through context-aware recommendations. In: Proceedings of the 21st ACM international conference on Information and knowledge management. CIKM '12, New York, NY, USA, ACM (2012) 2527–2530
5. Baltrunas, L., Ludwig, B., Peer, S., Ricci, F.: Context-aware places of interest recommendations for mobile users. In Marcus, A., ed.: Design, User Experience, and Usability. Theory, Methods, Tools and Practice. Volume 6769 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2011) 531–540
6. Ko, S.K., Choi, S.M., Eom, H.S., Cha, J.W., Cho, H., Kim, L., Han, Y.S.: A smart movie recommendation system. In Smith, M., Salvendy, G., eds.: Human Interface and the Management of Information. Interacting with Information. Volume 6771 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2011) 558–566

7. Domingues, M.A., Gouyon, F., Jorge, A.M., Leal, J.P., Vinagre, J., Lemos, L., Sordo, M.: Combining usage and content in an online recommendation system for music in the long tail. International Journal of Multimedia Information Retrieval **2**(1) (2013) 3–13
8. Mitzlaff, F., Stumme, G.: Namelings: discover given name relatedness based on data from the social web. In: Proceedings of the 4th international conference on Social Informatics. SocInfo'12, Berlin, Heidelberg, Springer-Verlag (2012) 531–534
9. Mitzlaff, F., Stumme, G.: Onomastics 2.0 - the power of social co-occurrences (2013)
10. Mitzlaff, F., Stumme, G.: Relatedness of given names. Human Journal **1**(4) (2012) 205–217
11. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems **30**(1-7) (1998) 107–117
12. Mitzlaff, F., Stumme, G.: Recommending given names (2013)
13. Su, X., Khoshgoftaar, T.: A survey of collaborative filtering techniques. Advances in Artificial Intelligence (2009)
14. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: WWW'01: Proceedings of the Tenth International Conference on World Wide Web, New York, NY, USA, ACM (2001) 285–295
15. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Eighth IEEE International Conference on Data Mining (ICDM'08). (2008) 263–272
16. Breese, J.S., Heckerman, D., Kadie, C.M.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence. (1998) 43–52
17. Deshpande, M., Karypis, G.: Item-based top-n recommendation algorithms. ACM Transaction on Information System **22**(1) (2004) 143–177
18. Jorge, A.M., Alves, M.A., Azevedo, P.J.: Recommendation with association rules: A web mining application. In: Proceedings of Information Society (IS-2002): Data Mining and Warehouses, Ljubljana, Slovenia (2002)
19. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of Twentieth International Conference on Very Large Data Bases. (1994) 487–499
20. Dey, A.K.: Understanding and using context. Personal Ubiquitous Computing **5**(1) (2001) 4–7
21. Panniello, U., Gorgoglione, M.: Incorporating context into recommender systems: an empirical comparison of context-based approaches. Electronic Commerce Research **12**(1) (2012) 1–30
22. Winkler, W.E.: Methods for evaluating and creating data quality. Information Systems **29**(7) (2004) 531–550
23. Christen, P.: A comparison of personal name matching: Techniques and practical issues. In: Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops, Washington, DC, USA, IEEE Computer Society (2006) 290–294
24. Domingues, M.A., Soares, C., Jorge, A.M.: An empyrical study on the impact of singleton web accesses on the accuracy of recommender systems. In: Proceedings of the SBIA 2008 First Workshop on Web and Text Intelligence (WTI 08), Salvador, Bahia, Brazil (2008) 43–50
25. Voorhees, E., Harman, D., of Standards, N.I., (US), T.: TREC: Experiment and evaluation in information retrieval. Volume 63. MIT Press Cambridge (2005)

26. Adomavicius, G., Sankaranarayanan, R., Sen, S., Tuzhilin, A.: Incorporating contextual information in recommender systems using a multidimensional approach. ACM Transactions on Information Systems **23**(1) (2005) 103–145
27. Domingues, M.A., Jorge, A.M., Soares, C.: Dimensions as virtual items: Improving the predictive ability of top-n recommender systems. Information Processing & Management **49**(3) (2013) 698–720