A mixed hybrid recommender system for given names

Rafael Glauber[1], Angelo Loula[1], and João B. Rocha-Junior[2]

[1] Intelligent and Cognitive Systems Lab (LASIC)
[2] Advanced Data Management Research Group (ADaM)
State University of Feira de Santana (UEFS)
Feira de Santana, Bahia, Brazil
{rafaelglauber, angelocl, joao}@ecomp.uefs.br

Abstract. Recommender systems are data filtering systems that suggest data items of interest by predicting user preferences. In this paper, we describe the recommender system developed by the team named *uefs.br* for the offline competition of the *15th ECML PKDD Discovery Challenge 2013* on building a recommendation system for given names. The proposed system is a hybrid recommender system that applied a content-based approach, a collaborative filtering approach and a popularity approach. The final recommendation is composed by the results of these three different approaches, in which parameters where optimized according to experiments conducted in datasets built from train data.

1 Introduction

Choosing a given name can be a hard task, considering the large amount of available names and the diverse personal preferences and cultural contexts [4]. The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML PKDD 2013, organized a Discovery Challenge on building a recommendation system for given names. The challenge has two phases, an offline competition, for predicting search activity for users based on data from the nameling.net website, and an online competition where teams would integrate their recommender system into the nameling.net website.

Here we describe the recommender system developed for the offline competition by the team named *uefs.br* (initially called *sertão*). The uefs.br team was ranked first in the last public leaderboard of the offline competition, with a score of 0,0491. The proposed system was a hybrid recommender system that applied a content-based approach, a collaborative filtering approach and a popularity approach.

In summary, the main contributions of this paper concern proposing a popularity-based, content-based and collaborative-based recommender system; proposing a mixed hybrid recommender system based on recommendations of the previous systems; and evaluating the performance of the hybrid recommender system using different settings.

The rest of this paper is organized as follows. In the next section, we present an overview on recommender systems. Section 3 states the proposed offline challenge task and how recommendations where evaluated. Then, we describe our proposed hybrid recommender system in Section 4. In Section 5, experimental results are presented along with parameters settings and the datasets used, and the last section gives final remarks on the proposed solution.

2 Recommender Systems

A Recommender System (RS) is a filtering system that suggests items of possible interest for a given user [6]. The problem of recommendation can be seen as the problem of estimating the user rating for items not yet rated by the user. Items predicted with high rating for a given user can be offered by the system as a recommendation.

There are two main approaches for recommendation, Content-based and Collaborative Filtering. Different recommendation approaches can be combined in a third approach, a Hybrid [1] one.

Content-based Approach. The Content-based (CB) approach [5] recommends new items according to their content similarity with items of previous user interest. It has its basis in Information Retrieval (IR) and Machine Learning (ML) [6, 3, 8]. This approach employs, for example, content representation and comparison techniques from IR besides classification algorithms from ML to represent items previously assessed by the user and compare with other items, in order to recommend similar items.

Collaborative Filtering Approach. Instead of comparing items, the Collaborative Filtering (CF) approach [7] compares users based on their item interests, and recommends new items that were of interest to such similar users. This technique, called the "automation of word-of-mouth" by Shardanand and Maes [9], has a filtering principle in which the workload of identifying relevant content (or quality from the perspective of an individual user) is partitioned among system users, who record their personal rating for retrieved items.

Hybrid Approach. The previous approaches, and any other approach, can be combined in order to employ complementary aspects of each other [1, 2]. Recommender systems that employ more than one approach are named a hybrid recommender systems. The main challenge of this approach is how to combine approaches, for example, by combining items suggested by a collaborative filtering approach and a content-based approach, in order to suggest items similar to those previously "recommended" by the active users and items of interest that are "recommended" by users with similar taste.

3 Problem Statement

The offline competition phase from *15th ECML PKDD Discovery Challenge 2013* defined the task of predicting future searches for users of the nameling.net

website. The nameling.net website is a search engine and a recommendation system for given names, based on data observations from the social web [4]. As the user enters the website, he enters a given name and gets a browsable list of "relevant" names, called "namelings". Each name listed, including the searched one, has details associated with it, including Wikipedia articles (categories), popularity in Twitter and Wikipedia and names commonly co-occuring.

Data captured from the nameling.net website was provided for offline competition, regarding logged interactions from the website users, called activities. Every user activity is supplied with the (obfuscated) user ID, the type of activity, the associated name and a time stamp. The types of activities could be ENTER_SEARCH, when the user entered a name into the search field, LINK_SEARCH, when the user clicked a link on a result page, LINK_CATEGORY_SEARCH, when the user clicked on a wikipedia category related with a name, resulting in a list of all name in the same category, NAME_DETAILS, when the user gets details associated with a name, or ADD_FAVORITE, when the user adds a name to his list of favorite names.

A subset of users is selected as test users, and for each user in this subset, the last two ENTER_SEARCH activities were removed from his activity history. Each competition team is challenged to build a recommender system that provides an ordered list of names (up to 1000 names) for each test user that should include the names related to the two omitted activities. These two names are present in a provided list of nameling.net known names, but they may not occur as ENTER_SEARCH or ADD_FAVORITE in the given user provided history. This procedure provides two datasets, a secret evaluation dataset, with withhold names for test users, and a public challenge dataset, with all remaining user activities.

To evaluate the recommended list of names for each test user, the Mean Average Precision at position 1000 (MAP@1000) was used as the challenge score. Given a user i and his first omitted name found at position n_1 and the second one at position n_2, the Average Precision $AveP@1000_i$ and $MAP@1000$ for N users were given by:

$$AveP@1000_i = \frac{1}{2}.\left(\frac{1}{n_1} + \frac{2}{n_2}\right)$$

$$MAP@1000 = \frac{1}{N}.\sum_{i=1}^{N} AveP@1000_i$$

If one of the omitted names was not in the ordered list, n_2 was admitted to be at position 1001, and if both were not in the list, n_1 was set to 1001 and n_2, 1002.

4 Proposed Solution

To address the offline competition task, we built a hybrid recommender system that combines recommendations coming from different recommender systems, a

content-based one, a collaborative filtering one and a popularity based one. In Section 4.1, we present the popularity-based approach that recommends names based on the popularity of the names in the collection. In Section 4.2, we describe our collaborative filtering approach that retrieves given names from users with similar name interests. In Section 4.3, we describe our content-based approach that recommends new names based on phonetic string matching with names in the last activities of the given user. Finally, in Section 4.4, we present our hybrid approach that combines the previous techniques for recommending names.

4.1 Popularity-based approach

The popularity-based approach recommends popular (frequent) names. The idea is ranking names based on frequency in the collection and suggesting the same result ordered list to every test user, but removing from the recommendation list names that have previously being associated with an ENTER_SEARCH or ADD_FAVORITE activity of the given user.

In order to determine name popularity, we parse the public challenge dataset and, for each name we count the number of users that have such a name in their history of activities. If a name appears more than once in the activities of a given user, only one occurrence of the name is counted.

The big advantage of this approach is its simplicity and capacity for filling the list of one thousand names. The big disadvantage is that it is not customized for each user.

4.2 Collaborative Filtering approach

The collaborative filtering system produces a customized list of recommended names for each test user coming from names in the list of activity of similar users (neighbors). To determine neighbors for a given test user, a similarity measure is computed between this test user and all other users. Those users with the similarity measure above a given threshold are considered neighbors of the test user. The list of names recommended to the given test user is composed by the names appearing in the neighbors' list of activities and that are not in the given test user list as an ENTER_SEARCH and ADD_FAVORITE activity.

Computing the similarity between two users (neighborhood similarity). The neighborhood similarity measure between two users is computed taking into account only the valid (known) names associated with ENTER_SEARCH and LINK_SEARCH activities for both users, defining a user profile. Profiles are represented as a n-dimensional binary vector with each dimension representing a name among the n valid names. If a name occurs in a profile, the profile vector has value 1 in that dimension, otherwise, it has value 0. As neighborhood similarity measure, we use the well-known *cosine* similarity [3] between profile vectors of a test user T and another user U:

$$similarity(T, U) = cos(\theta) = \frac{\sum_{i=1}^{n} T_i \cdot U_i}{\sqrt{\sum_{i=1}^{n} T_i^2} \cdot \sqrt{\sum_{i=1}^{n} U_i^2}} \quad (1)$$

where n is number of valid names (vector size), T_i and U_i are the i-th dimension value in vector T and U, respectively.

Selecting the neighbors. The neighbors of a test user T (test user) are those users U whose cosine similarity is greater or equal a given *Similarity Threshold*.

Selecting the names. After selecting neighbors, all names present in these neighbors profiles compose a list of candidate names for recommendation. For each name, the score is the sum of the cosine similarity of the neighbors that contain the name. This score is used to rank the candidate names and order the list. The final list of names recommended by the Collaborative Filtering Approach is limited in the hybrid system (*Collaborative Filtering Limit*).

The advantage of this approach is that it suggests names from users of similar taste, producing a customized list of names for each user. The limitation is that it may not find enough neighbors (or no neighbors at all) and it may not be able to provide a list of one thousand names.

4.3 Content-based approach

Our content-based approach checks the names in the last activities of the test user in order to recommend similar names. The recommended names are selected from a list of candidate names, which are determined comparing the names in the last activities with the list of valid names using an algorithm for *phonetic string matching* [10]. This algorithm can be used to suggest spelling variations of a given name. For example, given the name "johane" and a list of valid names, this procedure can suggest "johan", "johanie", and "johanne". In our proposed system, we have employed the Soundex algorithm [10], but any other phonetic string matching algorithm could have been employed.

Soundex employs a parameter to define the minimum string similarity between two names and only names above this string similarity are considered. Therefore, if we increase the parameter, less names are returned by content-based recommendation, once only strictly more similar names are recommended. The number of names provided by this approach depends not only on the string similarity parameter but also on the specific name under consideration. Since names are compared to the list of valid names, some names may have no similar names (empty set), while others have more than 10 similar ones. Besides, we also need to order the list of similar names and, to that end, we rank the suggested names by popularity (Section 4.1), hence we can select the most popular names suggested by the content-based approach.

The content-based approach is divided into two phases. In the first phase, we find the candidate names and in the second phase, we select the best names for recommendation.

Finding candidate names (first phase). In order to find the candidate names, we employ three techniques.

- **First.** We check if the user's last activity is an ENTER_SEARCH activity associated with a valid name (a name in known names list), and insert all similar names (using Soundex) in the candidates names list.

- **Second.** We check if the user's last activity is a LINK_SEARCH, inserting it as candidate. This name, when it appears, is always the first candidate.
- **Third.** We check the last four activities of a test user, looking for invalid names (names not in known names list) associated with ENTER_SEARCH activities. The invalid names are then split using the following delimiters: " -#+,". The aim is to transform invalid names such as "stine#" in valid names such as "stine" or invalid names such as "Christina Alexandra" in two valid names "Christina" and "Alexandra". The valid names are added to the candidate names list. If, after name split, the result names are still invalid, we search for similar valid names using Soundex, which are also added as candidate names.

Selecting the best candidate names (second phase). The list of candidate names produced in the first phase can contain many names. Therefore, in order to select the best names, we rank the list of candidates by popularity (Section 4.1). In a pure content-based approach, we can select up to one thousand names; while in our hybrid approach, we limited the number of names according to a parameter *Content-based Limit*.

The main limitation of this approach is that, in most cases, it does not fill the list of one thousand names. Besides, it does not recommend names to users with last activities associated with names with great phonetic difference to all other known names. The big advantage is exploiting the names in the user's last activities for recommendation, once the challenge task is to predict two subsequent names in the user profile, which are highly temporally dependent on the last activities.

4.4 Hybrid approach

All the approaches proposed above have strong and weak points and, therefore, they may not present the best results when considered individually. However, if the list of recommended names is built by combining names obtained from all the previous approaches, we can have a hybrid recommender system with better results. In our proposal, the final list of recommended names is composed by the resulting list of recommended names from each of the three different approaches, characterized as a mixed hybrid system [1].

In order to find the best way to combine the results from the different approaches, we have tried different combinations of the previous approaches. Figure 1 presents the best combination found, where the two first items are obtained using the content-based approach, the next names (up to three hundred) are obtained using the collaborative filtering approach, and the remaining names are obtained using the popularity-based approach.

The rationale behind the hybrid approach is putting first the names related to the last few activities of the user (content-based approach), which is a short list that exploits high scoring positions in the MAP score, but leaves a lot of subsequent score positions for the following approach. Then, collaborative filtering fills the 300 following names in the recommendation list, which is still a

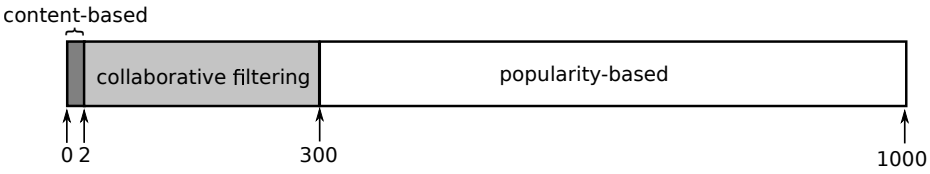


Fig. 1. Hybrid approach and its final combination.

user customized recommendation. Finally, to complete the list of 1000 names, since the previous two may not do so, we add popular names with overall high probability of being of interest.

5 Experimental Evaluation

An experimental evaluation was conducted to compare the different approaches proposed and the impact of parameters in the hybrid recommender system. In each experiment, we vary one parameter, while the others are fixed, therefore, showing the impact of one parameter in system results.

5.1 Settings

Our hybrid recommender system combines different approaches for building a recommendation list. Table 1 presents the parameters employed in the experimental evaluation, the default values are presented in bold.

Parameters	**Values**
Content-based limit	1, **2**, 3, 4, 5
Collaborative filtering limit	100, 200, **300**, 400, 500
Soundex parameter for valid names	0.93, 0.94, 0.95, **0.96**, 0.97
Soundex parameter for invalid names	0.89, 0.90, **0.91**, 0.92, 0.93
Similarity threshold	0.09, 0.10, **0.11**, 0.12, 0.13
Dataset	1, 2, 3

Table 1. Parameter settings used in the experiments.

Content-based Limit (ContentLimit) is the parameter that controls the maximum number of names recommended by the content-based approach.

Collaborative Filtering Limit (CollaborativeLimit) is the parameter that controls the maximum number of names recommended by the collaborative filtering approach.

Soundex Parameter for Valid Names (SoundexValid) and Soundex Parameter for Invalid Names (SoundexInvalid) defines minimum string similarity required for phonetically similarity between two names.

Similarity Threshold (SimThreshold) defines minimum cosine similarity required for a user to be considered neighbor of the test user (active user).

5.2 Datasets

In order to evaluate the impact of each parameter in our recommender system, we have created three evaluation datasets from the public challenge dataset. The datasets were created using the provided script to generate test users, then splitting the result test users dataset into three smaller datasets and combined with users from the public challenge dataset, keeping similar characteristics to the original dataset . In each new dataset, only test users have their profile changed (the last two ENTER_SEARCH activities was removed), while the profile of the other users are kept the same as in the public challenge dataset.

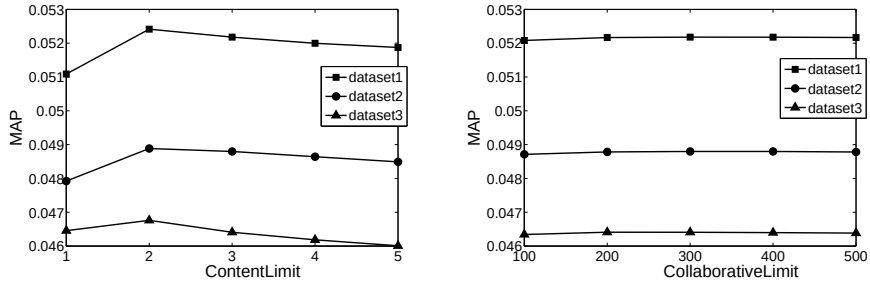
Properties	**Dataset 0**	**Dataset 1**	**Dataset 2**	**Dataset 3**
#valid names	17457	17326	17309	17291
#invalid names	14638	14232	14277	14175
#test users	4140	4139	4141	4728
avg #valid names per user	4.2643	4.1013	4.1017	4.0765
avg #invalid names per user	0.3087	0.2985	0.3000	0.2976
avg #users per valid name	14.8046	14.3463	14.3619	14.2884
avg #users per invalid name	1.2781	1.2714	1.2734	1.2725
max #valid names per user	1476	1476	1476	1476
max #invalid names per user	61	60	61	61
max #users per valid name	2263	2183	2189	2181
max #users per invalid name	57	54	55	56

Table 2. Datasets characteristics.

Table 2 shows the characteristics of the public challenge dataset (Dataset 0) and the three new datasets created (Datasets 1, 2, and 3). These characteristics were extracted from all activities, except LINK_CATEGORY_SEARCH. Each new dataset has a distinct set of test users. The valid names are those that are present in the provided list of known names, while the invalid names are ENTER_SEARCH activities whose names are unknown names.

5.3 Defining the number of items for each approach

As an initial step, we study the maximum number of names taken from each approach in our hybrid recommender system (Figure 2). The list of names recommended by the hybrid recommender system is composed by items obtained using the content-based approach, followed by items obtained using the collaborative filtering approach and, filling the rest of the list, with items obtained using the popularity-based approach. First, we study the limits of the content-based approach, then we study the limits of the collaborative-based approach.



(a) Content-based limit (b) Collaborative filtering limit

Fig. 2. Number of items from the (a) content-based and (b) collaborative filtering approaches used in the hybrid recommender system.

Content-based limit (ContentLimit). Figure 2(a) presents the MAP obtained with our hybrid recommender system, while the *ContentLimit* varies[3]. The best result was obtained using only two items from the content-based approach. Besides the small number of items, the impact of the content-based approach in our recommender system is significant, since the items recommended are the two first items in the list (Section 3).

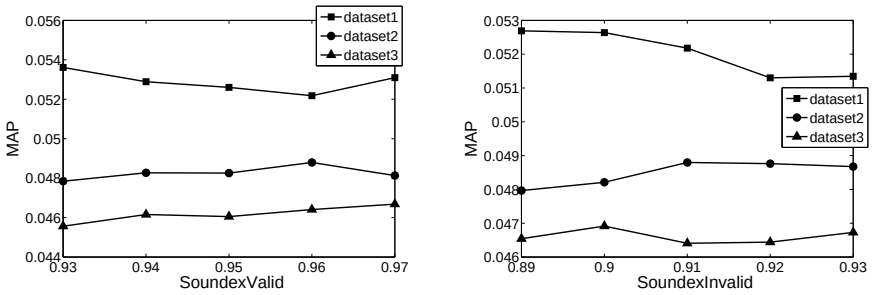
Collaborative filtering limit (CollaborativeLimit). Figure 2(b) presents the MAP obtained with our hybrid recommender system, while the *CollaborativeLimit* varies. The *CollaborativeLimit* includes the items obtained using the content-based approach. Therefore, when $CollaborativeLimit = 300$, it means that the two first items are obtained using the content-based approach, while the other 298 items are obtained using the collaborative-based approach. Results show a really small difference in MAP for variations of *CollaborativeLimit*, but a slightly better MAP for the value 300, which is the reason for using this value. This small difference also evidences that the collaborative filtering and the popularity approach almost compensates each other in MAP score, when more names from one approach and less from the other are included.

5.4 Calibrating the phonetic string matching algorithm

The impact of the phonetic string matching algorithm (Soundex) in our hybrid recommender system was also evaluated. The Soundex algorithm compares phonetic similar names and allows to locate known names similar to valid and invalid names (Section 4.3). Therefore, we have configured Soundex differently, depending whether the name is valid or invalid.

Figure 3(a) presents the MAP while varying the phonetic string similarity threshold for valid names, while Figure 3(b) shows the MAP for invalid names. Although there is no clear convergence of this parameters in all three datasets, we have chosen Dataset 2 as our benchmark as it approaches better results obtained (weekly) from the secret evaluation dataset and it also represents a

[3] In all figures, we employ the default values presented in Table 1, while varying one single parameter.



(a) Similarity for valid name (b) Similarity for invalid name

Fig. 3. String similarity threshold parameter in Soundex algorithm.

balance between the higher results obtained for Dataset 1 and lower results for Dataset 3. Therefore, we employed the value 0.96 for valid names and 0.91, which maximizes the MAP for Dataset 2.

5.5 Neighborhood selection

Another evaluation concerned the minimum similarity required for considering a user to be a neighbor of a test user in the collaborative filtering approach (Section 4.2). Only users whose similarity to a given test user is above this similarity threshold are considered neighbors, who can recommend names to the test user. Figure 4 presents the impact on the MAP for the similarity threshold varying from 0.09 to 0.13. The best results are obtained when this value is set to 0.11.

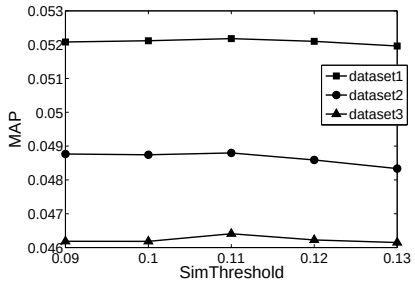


Fig. 4. The lowest degree of similarity required to be neighbor.

5.6 Comparing the different approaches individually

To evaluate the potential of each recommendation approach, we studied each one as an isolated recommender system. Figure 5 presents the MAP score for each of the three approaches, when they are considered individually. In these experiments, we fill the list with items recommended by a single approach, without limiting the number of items suggested. For example, for the content-based approach, we list all names that can be recommended by this approach without limiting to only two names.

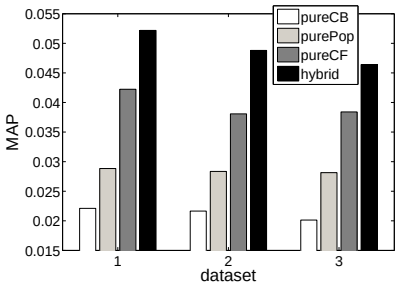


Fig. 5. Comparing approaches as isolated recommenders.

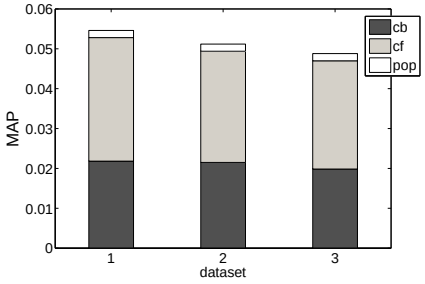


Fig. 6. Impact of each approach in the hybrid recommendation.

When considered individually as a pure recommender system, the collaborative filtering approach (pureCF) presents the best results, while the content-based approach (pureCB) is the one that presents the worst results, with popularity-based (purePop) having intermediate results. The content-based approach does not recommend many names, therefore, most lists of recommended names using this approach have much less than one thousand items (or even no recommendation at all). But pureCB has a considerable MAP score for such a limited list of names. Besides the good results obtained with the pure collaborative-based approach, the hybrid is able to combine characteristics from each approach producing the overall best results. These experiments shows the efficacy of our hybrid recommender system in aggregating recommendations from the different approaches.

5.7 Studying the impact of each approach to the hybrid system

In order to assess the individual contribution from each recommendation approach to the hybrid approach, we evaluate the MAP score composition from each one in the final recommendation list. Figure 6 presents the results obtained using the hybrid approach, showing the contribution of each approach in the result (cf:collaborative filtering, cb:content-based and pop:popularity-based). In this experiment, we employ the default values presented in Table 1.

The first thing to notice is the contribution of the content-based approach that provides only two items at most, but has a significant impact in the results obtained by the hybrid approach. The main reason for this high impact is that the two items recommended by this approach are put in the top of the list, positions highly scored, and this items are based on the content of the last user activities.

In accordance with the results presented in the previous section, collaborative filtering is the approach with the highest impact in our hybrid recommender system. The names suggested by this approach are also in the top of the list (two positions after the content-based approach) and capture the diversity of names suggested by similar users, ensuring good results for the hybrid approach.

Although with a small contribution, the popularity-based approach has its importance. The popularity-based approach fills a great part of the recommendation list (the tail of the list) that would not be filled properly by any of the other approaches studied, ensuring that all recommended lists have one thousand names.

6 Final Remarks

The task of recommending given names is very hard. Even though the list of recommend names is quite long, with up to 1000 names, it is still difficult to predict given names of interest for many users, even with a hybrid system that combines efforts from different approaches. There are many cultural factors and behavior (or even fashion!) that may influence the choice of a name. A deeper understanding of such factors, particularly with a much larger and representative dataset, can be a fruitful track to build more efficient recommender systems for given names.

Acknowledgments. The authors would like to thank Ivo Romário Lima and Matheus Cardoso Silva for their collaboration during system development.

References

1. R. Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.
2. J. Liu, P. Dolan, and E. Pedersen. Personalized news recommendation based on click behavior. In *Proceedings of the 15th international conference on Intelligent user interfaces*, pages 31–40. ACM, 2010.
3. C. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
4. F. Mitzlaff and G. Stumme. Namelings - discover given name relatedness based on data from the social web. In *Proceedings of the International Conference on Social Informatics (SocInfo)*, pages 531–534, 2012.
5. M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer, 2007.
6. F. Ricci, L. Rokach, and B. Shapira. Introduction to recommender systems handbook. In F. Ricci, L. Rokach, B. Shapira, P. B. Kantor, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, chapter 1, pages 1–35. Springer, Boston, MA, 2011.
7. J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer, 2007.
8. F. Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002.
9. U. Shardanand and P. Maes. Social information filtering: algorithms for automating "word of mouth". In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '95, pages 210–217, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.
10. J. Zobel and P. Dart. Phonetic string matching: lessons from information retrieval. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information retrieval*, pages 166–172, 1996.