Tackling Multi-member Problem in IPTV Recommendation with Virtual Users

Peng Zhou[1, a]

[1]Department of Computer Science and Engineering,Shanghai Jiao Tong University, Shanghai 200240, China

[a]peterchou139@sjtu.edu.cn

Keywords:IPTV, recommendation, multi-member, virtual users, content-based recommendation, collaborative filtering

Abstract: With the popularity of the Internet, Internet Protocol Television (IPTV) has been replacing traditional TV and recommendation systems can help IPTV users to choose programs that may interest them. Compared with other recommendations (e.g., movies), IPTV recommendation suffers from a multi-member problem — more than one family members in a household share TV. Because family members can have different tastes, this problem makes recommendations less accurate for IPTV.

We propose a virtual user approach to solve this problem. We virtualize two virtual users in a household — Kid and Adult. We have integrated virtual users into both content-based and collaborative filtering recommendation approaches. Experimental results on a real IPTV dataset show that the proposed approaches perform better than the recommendation algorithms without virtual users.

Introduction

The Internet has permeated every aspect of our life with personal computers, mobile phones, and pads. Traditional TV, without exception, has been superseded by Internet Protocol Television (IPTV) for its richer content and more interactivity. Because of much richer content, IPTV users often face information overload and it becomes increasingly difficult for users to choose programs to watch. Thus, recommending interesting TV programs for users is crucial for providing better user experience.

Generally, recommendation approaches are classified into two categories, content-based approaches and collaborative filtering [3]. The content-based approaches recommend items similar to the ones that a given user has preferred in the past. In contrast, collaborative filtering recommends items that users having similar tastes with the given user have preferred. Previous work [4,13] on IPTV recommendation has adopted a hybrid approach by combining content-based approach with collaborative filtering.

Compared with other types of recommendation (e.g., movie recommendation of Netflix [1]), IPTV recommendation is particularly challenging for a multimember problem. That is, in a household there are more than one viewers and different family members may have different tastes. Because IPTV data usually only contain the interactions between TV and its users (which does not distinguish different family members), it is difficult for recommendation algorithms to take the subtle difference within tastes into account and to make predictions accordingly. Thus, the multi-member problem makes recommendation less accurate for IPTV.

In this paper, we propose a virtual user approach to solve the multi-member problem. Instead of representing each IPTV user as a single entity, we use a combination of multiple virtual users to represent a single household. In this way, different virtual users capture various tastes among family members, resulting in better recommendation performance. The contribution of this paper includes the introduction of virtual users to a content-based recommendation algorithm and a collaborative filtering algorithm, and our evaluation results on a real IPTV dataset demonstrating the effectiveness of the proposed virtual users.

The rest of the paper is organized as follows. Section 2 summarizes related work. Section 3 illustrates virtual users and how we integrate virtual users into a content-based recommendation algorithm and a collaborative filtering algorithm. Section 4 evaluates the performance of our scheme with real IPTV data. Finally, Section 5 concludes our work.

Related Work

With the increasing popularity of IPTV, its recommendation has become a hot research topic. Barragans-Martı nez et al. mixed the results from a content-based recommendation and an item-based collaborative filtering [4]. Specifically, they enhanced collaborative filtering with singular value decomposition [7] to alleviate both scalability and data sparsity problems, achieving higher recommendation accuracy. Park et al. [13] also proposed a hybrid approach. Rather than recommending a fixed ratio of programs from content-based recommendation and collaborative filtering, Park et al. recommended top-five ranked IPTV programs to users according to the predicted programs' preferences. Finally, Kim et al. [9] proposed a recommendation algorithm using ontology and k-means algorithm. Different from the above studies, this paper targets the multi-member problem in IPTV recommendation and proposes a virtual user based solution. We have integrated virtual users into both content-based and collaborative filtering recommendation algorithms.

Content-based recommendation can be divided into two steps, i.e., obtaining item representation and making recommendation. Typically, item representation is often formulated via vector space model [16]. After obtaining item representation, various algorithms are used to make recommendation, e.g., naive Bayes, Rocchio, decision trees, nearest neighbor, and linear classifiers [11,15,8]. In this paper, we integrate virtual users into a nearest neighbor based recommendation algorithm [2,6].

Collaborative filtering can be classified into two categories: (i) memory-based algorithms that store all ratings, items and users in memory, and (ii) model-based algorithms that create a summary of ratings patterns offline [17,8]. The Netflix Prize competition [1] that began in October 2006 has promoted the development of collaborative filtering. For the first time, the research community gained access to a large-scale industrial data set of more than 100 million movie ratings and attracted lots of researchers to the domain [10]. Finally, the "BellKor's Pragmatic Chaos" team won the Netflix Grand Prize [10,1] using a collaborative filtering algorithm based on Regularized Singular Value Decomposition (RSVD) [14] (referred as SVD in [10]). This RSVD model is popular among Netflix competitors and we choose to enhance the model with virtual users in our work.

Design

This section first introduces virtual users. Then we discuss the integration of virtual users into a content-based recommendation algorithm and a collaborative filtering algorithm.

Virtual Users: Because a single household often has multiple viewers, we introduce virtual users to represent different family members. Specifically, we virtualize two virtual users in a household — Kid (corresponds to children in the household) and Adult (corresponds to parents and grandparents in the household). The IPTV dataset we used have predefined categories of all programs and we assign kid category to virtual user Kid and assign other categories to virtual user Adult.

Enhancing Content-Based Recommendation with Virtual Users: The content-based recommendation algorithms usually represent an item as a vector. Then the vector space is powered by Latent Semantic Analysis (LSA) [5] to reduce dimensions in the item representations. Finally, we use nearest neighbor to make recommendation for virtual users for its simplicity and stability [6]. These steps are discussed in the following.

Item Representation. In our data set, each item (program) is associated with some metadata, such as the title and description of the program, actor names, director names, and country names. We tokenize these strings, remove stop words, and compute a vector of TF-IDF [12] values of tokens for titles and descriptions. For actor, director, and country names, each distinct name is represented as a feature in a name vector, with one indicating the name is in the metadata. Then, we combine the TF-IDF vector and the name vector into a single vector for the item. Finally, we obtain a term-item matrix, where each column corresponds to the vector of an item.

Dimension Reduction with LSA. We conduct LSA on the term-item matrix to reduce noise and to reduce dimensions for items [5]. Specifically, LSA is performed with singular value decomposition. Given a $m \times n$ term-item matrix $\mathbf{A}$ (m terms and n items). $\mathbf{A}$ can be decomposed into three matrices, $\mathbf{U}$ ($m \times f$), $\mathbf{\Sigma}$ ($f \times f$), $\mathbf{V}$ ($n \times f$):

$$\mathbf{A} \approx \widehat{\mathbf{A}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \qquad (1)$$

where f ($f \ll m$) is the number of latent semantic features of items, $\widehat{\mathbf{A}}$ is the best rank-f approximation of $\mathbf{A}$ in terms of Frobenius norm. $\mathbf{U}$ and $\mathbf{V}$ are column orthonormal matrices and $\mathbf{\Sigma}$ is a diagonal matrix. The best value of f is determined by cross-validation.

Now we get a new vector space that has f dimensions, which is called latent semantic space. In the latent semantic space, the items i' s coordinates is given by the i row $\mathbf{v}_i$ of matrix $\mathbf{V\Sigma}$ [5].

Recommendation with Nearest Neighbor. For a given virtual user u_s, we predict the virtual user' s rating for item i as follows. First, we compute the similarity between item i and items the virtual user u_s watched in the past. We use cosine distance to compute the similarity $sim(i, j)$ between item i and item j:

$$sim(i, j) = \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_i\|\|\mathbf{v}_j\|}. \qquad (2)$$

Then we sort the items by similarity in the descending order and pick the top k items. These k items are denoted as set K_i .The predicted rating of item i is given by:

$$\hat{r}_i = \frac{\sum_{j \in K_i} sim(i, j) r_j}{\sum_{j \in K_i} sim(i, j)}, \qquad (3)$$

Where r_j is rating of item j.

For other virtual users in the household, we repeat the process above. The final recommended items are those with top predicted ratings among all virtual users.

Enhancing Collaborative Filtering with Virtual Users: Instead of using traditional item-based and user-based collaborative filtering, we adopt the RSVD model for its better performance in making predictions [10]. Each item i is associated with a vector $\mathbf{q}_i \in \mathbb{R}^f$ and an item bias b_i, and each user u is associated with a vector $\mathbf{p}_u \in \mathbb{R}^f$ and a user bias b_u. $\mathbf{q}_i^T \mathbf{p}_u$ captures the interaction between user u and item i. f is the number of latent factors and μ is the average value over all ratings. The predicted rating is given by the model:

$$\hat{r}_{ui} = \mu + b_i + b_u + \mathbf{q}_i^T \mathbf{p}_u. \qquad (4)$$

The above model can be trained with a stochastic gradient descent method (see [10] for details).

We propose a new model called V-RSVD, integrating virtual users into the RSVD model. In our model, there are multiple virtual users, where each virtual user u_s is associated with a vector $\mathbf{p}_{u_s} \in \mathbb{R}^f$ and a user bias b_{u_s}. $\mathbf{q}_i^T \mathbf{p}_{u_s}$ captures the interaction between virtual user u_s and item i. μ_s is the average value over all ratings assigned to all the virtual users. The V-RSVD model is denoted by:

$$\hat{r}_{u_s i} = \mu_s + b_i + b_{u_s} + \mathbf{q}_i^T \mathbf{p}_{u_s}. \qquad (5)$$

In order to train the model, we need to minimize the regularized squared error:

$$\min_{b_*,\mathbf{q}_*,\mathbf{p}_*} \sum_{(u_s,i)\in R} (r_{u_s i} - \mu_s - b_i - b_{u_s} - \mathbf{q}_i^T \mathbf{p}_{u_s})^2 + \lambda(b_i^2 + b_{u_s}^2 + \|\mathbf{q}_i\|^2 + \|\mathbf{p}_{u_s}\|^2), \tag{6}$$

Where R is set of the (u_s, i) pairs for which $r_{u_s i}$ is known and λ is the regularized parameter.

The above minimization can be solved with stochastic gradient descent over all ratings. For each known rating $r_{u_s i}$, we have a corresponding rating $\hat{r}_{u_s i}$ predicted by V-RSVD model. For simplicity, we denote $r_{u_s i} - \hat{r}_{u_s i}$ as $e_{u_s i}$. For a given training case $r_{u_s i}$, the parameters are updated as:

$$\begin{aligned}
b_i &= b_i + l(e_{u_s i} - \lambda b_i),\\
b_{u_s} &= b_{u_s} + l(e_{u_s i} - \lambda b_{u_s}),\\
\mathbf{q}_i &= \mathbf{q}_i + l(e_{u_s i}\mathbf{p}_{u_s} - \lambda \mathbf{q}_i),\\
\mathbf{p}_{u_s} &= \mathbf{p}_{u_s} + l(e_{u_s i}\mathbf{q}_i - \lambda \mathbf{p}_{u_s}),
\end{aligned} \tag{7}$$

Where l is the learning rate and we set it as 0.002 in our experiments.

Finally, the recommended items are those with top predicted ratings among all virtual users.

Evaluation

Dataset and Settings: The dataset used in our study is from a provincial-level broadcasting corporation of China. It consists of more than 1.6 billion watching records of over 600,000 IPTV VOD programs by over 2.4 million users from January 1st, 2014 to November 30th, 2014. The dataset also contains metadata about the programs, including title, description, actor names, director names and country names. Every program has a type attribute and Table 1 lists all 16 different program types. For each user, we assign programs of type kid to the virtual user Kid and designate others to the virtual user Adult.

Table 1. Program types and virtual users

Virtual user	Type
Kid	Kid
Adult	Series
	Documentary
	News
	Music
	Sport
	Movie
	Law
	Military
	Finance
	Entertainment
	Life
	Fashion
	Game
	History
	Culture

Because there are no explicit ratings in the dataset, we first convert viewing record into numeric values. Let T (minutes) denote the overall time of the program and t (minutes) be the actual watching time by the user. If the actual playing time is less than five minutes, that watching record is discarded. Otherwise, the estimated rating r is given by:

$$r = 1 + 4\frac{t}{T}, \quad 5 \leqslant t \leqslant T. \tag{8}$$

We use Root Mean Square Error (RMSE) to evaluate the performance of recommendation algorithms. The RMSE value is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{|R_{\text{test}}|}\sum_{(u_s,i)\in R_{\text{test}}}(r_{u_s i} - \hat{r}_{u_s i})^2}, \tag{9}$$

Where R_{test} is set of the (u_s, i) pairs for which $r_{u_s i}$ is in the test data set. Smaller RMSE values mean better performance on recommendations. In the experiments, we divide the dataset into two parts, a training set and a testing set. The percentages of training set are 90, 70, 50, and 30, respectively.

We compare the performance of the following recommendation approaches:

CB: Content-based approach without virtual users. This approach can be considered as only one virtual user for a household, where all ratings are assigned to a single virtual user.

V-CB: Content-based approach integrated with two virtual users, Kid and Adult.

CF: Collaborative filtering approach using the RSVD model [10].

V-CF: Collaborative filtering approach using the proposed V-RSVD model with two virtual users, Kid and Adult.

In the V-RSVD model, the learning rate l is set to 0.002, and the regularized parameter λ is set to 0.02.

Overall Performance Comparison: Table 2 shows the performance comparison between CB and V-CB and Table 3 shows the performance comparison between CF and V-CF. All experiments are performed with the best parameters, i.e., f and k in CB and V-CB and f in CF and V-CF. From both tables, we can observe the approaches integrated with virtual users (V-CB and V-CF) perform better than their counterparts without virtual users (CB and CF), because virtual users can better capture family members’ tastes and make recommendations accordingly. This is evident from the observation that both virtual user Kid and Adult perform better than their non-virtualized counterparts in CB and CF. The virtual user Kid performs better than virtual user Adult, because our current model cannot distinguish adult members in the family. That is, recommendation for the virtual user Adult still suffers from the multi-member problem, which we leave as future work.

Table 2 and Table 3 also show that collaborative filtering approaches always perform better than content-based ones. This is partly due to incomplete metadata information in our dataset. For instance, a program may not have information about its actors and directors. Unlike collaborative filtering, content-based approaches depend on such metadata of programs. Consequently, the lack of the metadata information lowers the accuracy of content-based approaches.

Table 2. Comparison of recommendation accuracy between CB and V-CB.

Train	Virtual user	CB	V-CB	Improvement
90%	Kid	1.0311	1.0192	1.15%
	Adult	1.1157	1.1152	0.04%
	All	1.0910	1.0873	0.34%
70%	Kid	1.0375	1.0225	1.45%
	Adult	1.1169	1.1168	0.01%
	All	1.0933	1.0889	0.40%
50%	Kid	1.0418	1.0321	0.93%
	Adult	1.1225	1.1212	0.12%
	All	1.0997	1.0961	0.33%
30%	Kid	1.0663	1.0593	0.66%
	Adult	1.1418	1.1420	-0.02%
	All	1.1182	1.1162	0.18%

Table 3. Comparison of recommendation accuracy between CF and V-CF.

Train	Virtual user	CF	V-CF	Improvement
90%	Kid	0.9576	0.9490	0.90%
	Adult	1.0481	1.0480	0.01%
	All	1.0218	1.0193	0.24%
70%	Kid	0.9618	0.9544	0.77%
	Adult	1.0504	1.0500	0.04%
	All	1.0242	1.0218	0.23%
50%	Kid	0.9881	0.9792	0.91%
	Adult	1.04800	1.04798	0.002%
	All	1.0310	1.0285	0.24%
30%	Kid	1.0018	0.9959	0.59%
	Adult	1.0624	1.0617	0.07%
	All	1.0434	1.0411	0.22%

Impact of Parameters f and k in V-CB: In this experiment, we investigate the impact of parameters f (the number of latent semantic features) and k (the number of the neighbors) in V-CB. The best parameters are different when the percentages of training set are different. For simplicity, we only show the impact of the parameters when the percentage of training set is 90%. We try different combinations of $f \in \{10, 20, 50, 80, 120, 150, 200\}$ and $k \in \{10, 20, 50, 80, 120, 150\}$.

Figure 1 shows the impact of the number of latent features f on recommendations. We can observe that generally RMSE decreases when f is increasing, because more features result in more accurate latent models. When the value of f becomes large (close to 200), RMSE increases slightly. This is due to small noise introduced into the model by the newly added features.

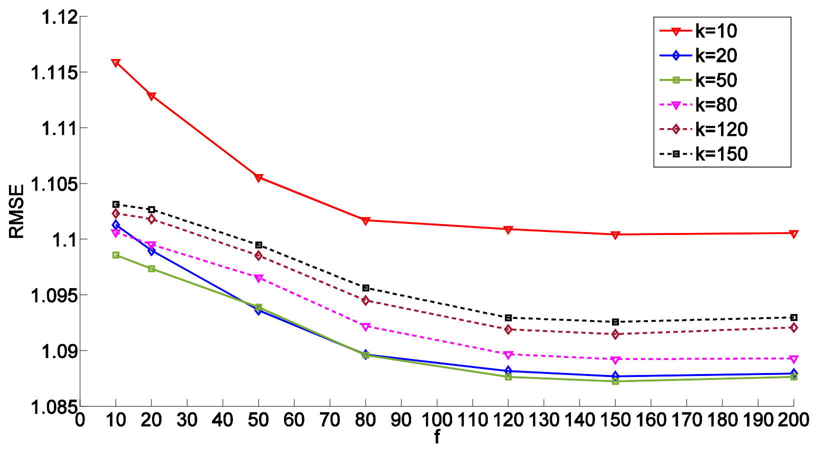

Fig. 1. The impact of the number of latent factors f on recommendation accuracy in V-CB.

Figure 2 shows the impact of the number of the neighbors k on recommendations. We can observe that initially when the value of k is increasing RMSE decreases, because more similar items are added to the prediction model. However, when k is larger than 50, RMSE increases due to more dissimilar items added to the model.

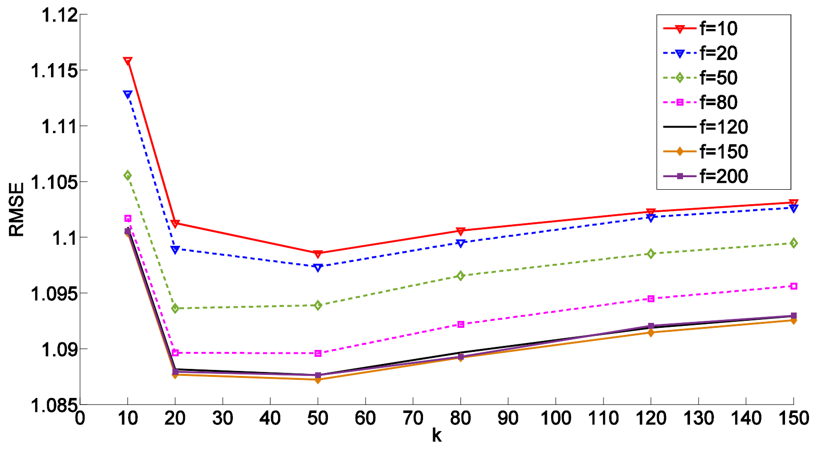

Fig. 2. The impact of the number of neighbors k on recommendation accuracy in V-CB.

Impact of Parameter f in V-CF: This experiment studies the impact of the number of latent factors f on recommendations in V-CF. Similar to V-CB, the best parameters are different when the percentages of training set are different. For simplicity, we only show the results when the percentage of training set is 90%.

Figure 3 shows the RMSE values for different values of f. We can observe that the best performance is achieved when value of f is 50. When the value of f is small, RMSE is high. When the number of latent factors is smaller, $\mathbf{q}_i^T\mathbf{p}_{u_s}$ cannot capture the interaction between virtual user u_s and item i well, because the V-RSVD model has high bias by omitting useful latent factors. On the other hand, RMSE goes up with larger f values. In this case, the V-RSVD model tends to overfit the training data, resulting in lower performance in recommendations.

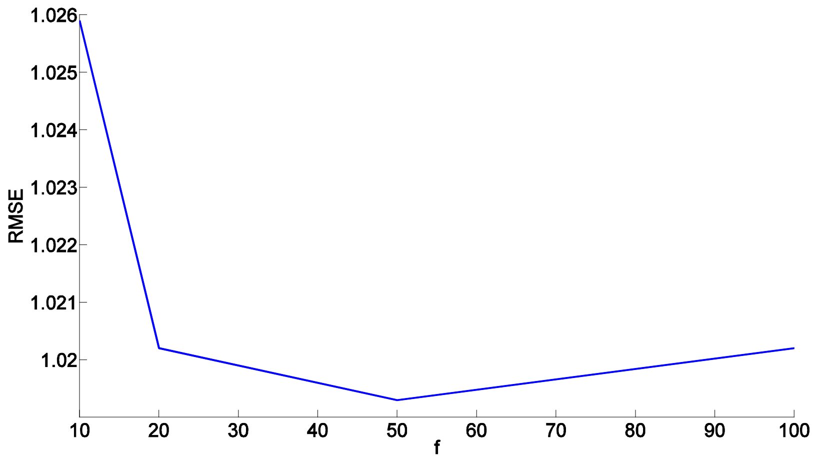

Fig. 3. The impact of the number of latent factors f on recommendation accuracy in V-CF.

Conclusions and Future Work

In this paper, we introduce virtual users to solve the multi-member problem in IPTV recommendation. We integrate virtual users into two types of recommendation algorithms. Experimental results on a real IPTV dataset demonstrate that virtual users improve the accuracy of both content-based recommendation and collaborative filtering approaches. Compared with virtual user Adult, virtual user Kid is more effective in improving the predictions.

The virtual user Adult may correspond to more than one family members (e.g., parents and grandparents). We can’t distinguish individual family members among them. In the family, the male members may prefer certain types of programs (e.g., news and military types) while the females may prefer life type of programs. In future work, we plan to investigate using more virtual users to represent those members. Another direction is to integrate virtual users into hybrid recommendation approaches.

References

[1] Netflix Prize: Home. http://www.netflixprize.com

[2] Altman, N.S.: An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician 46(3), 175 – 185 (1992), http://www.tandfonline.com/doi/abs/10.1080/00031305.1992.10475879

[3] Balabanovic, M., Shoham, Y.: Fab: Content-based, collaborative recommendation.Commun. ACM 40(3), 66 – 72 (Mar 1997), http://doi.acm.org/10.1145/245108. 245124

[4] Barragans-Martı nez, A.B., Costa-Montenegro, E., Burguillo, J.C., Rey-Lopez, M.,Mikic-Fonte, F.A., Peleteiro, A.: A hybrid content-based and item-based collaborative filtering approach to recommend tv programs enhanced with singular value decomposition. Inf. Sci. 180(22), 4290 – 4311 (Nov 2010), http://dx.doi.org/10.1016/j.ins.2010.07.024

[5] Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Harshman, R.: Indexing bylatent semantic analysis. Journal of the American society for information science 41(6), 391 – 407 (1990)

[6] Desrosiers, C., Karypis, G.: A comprehensive survey of neighborhood-based recommendation methods. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.) Recommender Systems Handbook, pp. 107 – 144. Springer US (2011), http://dx. doi.org/10.1007/978-0-387-85820-3_4

[7] Golub, G.H., Van Loan, C.F.: Matrix computations. Johns Hopkins University Press (1996)

[8] Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems: An Introduction. Cambridge University Press, New York, NY, USA, 1st edn. (2010)

[9] Kim, J., Kwon, E., Cho, Y., Kang, S.: Recommendation system of iptv tv program using ontology and k-means clustering. In: Kim, T.h., Adeli, H., Robles, R., Balitanas, M. (eds.) Ubiquitous Computing and Multimedia Applications, Communications in Computer and Information Science, vol. 151, pp. 123 – 128. Springer Berlin Heidelberg (2011), http://dx.doi.org/10.1007/978-3-642-20998-7_16

[10] Koren, Y., Bell, R.: Advances in Collaborative Filtering. In: Ricci, F., Rokach, L.,Shapira, B., Kantor, P.B. (eds.) Recommender Systems Handbook, pp. 145 – 186. Springer (2011)

[11] Lops, P., de Gemmis, M., Semeraro, G.: Content-based recommender systems: State of the art and trends. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (eds.) Recommender Systems Handbook, pp. 73 – 105. Springer US (2011), http://dx.doi.org/10.1007/978-0-387-85820-3_3

[12] Manning, C.D., Raghavan, P., Schu ¨ tze, H.: Introduction to Information Retrieval.Cambridge University Press, New York, NY, USA (2008)

[13] Park, K., Choi, J., Lee, D.: A single-scaled hybrid filtering method for iptv program recommendation. Int. J. of Circuits, Systems and Signal Processing 4(4), 161 – 168 (2010)

[14] Paterek, A.: Improving regularized singular value decomposition for collaborative filtering. In: Proceedings of KDD cup and workshop. vol. 2007, pp. 5 – 8 (2007)

[15] Pazzani, M., Billsus, D.: Content-based recommendation systems. In: Brusilovsky,P., Kobsa, A., Nejdl, W. (eds.) The Adaptive Web, Lecture Notes in Computer Science, vol. 4321, pp. 325 – 341. Springer Berlin Heidelberg (2007), http://dx. doi.org/10.1007/978-3-540-72079-9_10

[16] Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM 18(11), 613 – 620 (Nov 1975), http://doi.acm.org/10.1145/361219.361220

[17] Schafer, J., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) The Adaptive Web, Lecture Notes in Computer Science, vol. 4321, pp. 291 – 324. Springer Berlin Heidelberg (2007), http://dx.doi.org/10.1007/978-3-540-72079-9_9