PCF: PROJECTION-BASED COLLABORATIVE FILTERING

Ibrahim Yakut, Huseyin Polat
Computer Engineering Department, Anadolu University, 26470 Eskisehir, Turkey

Mehmet Koc
Electrical and Electronics Engineering Department, Anadolu University, 26470 Eskisehir, Turkey

Keywords: Collaborative filtering, e-Commerce, projection, Scalability and Accuracy.

Abstract: Collaborative filtering (CF) systems are effective solutions for information overload problem while contributing web personalization. Different memory-based algorithms operating over entire data set have been utilized for CF purposes. However, they suffer from scalability, sparsity, and cold start problems. In this study, in order to overcome such problems, we propose a new approach based on projection matrix resulted from principal component analysis (PCA). We analyze the proposed scheme computationally; and show that it guarantees scalability while getting rid of sparsity and cold start problems. To evaluate the overall performance of the scheme, we perform experiments using two well-known real data sets. The results demonstrate that our scheme is able to provide accurate predictions efficiently. After analyzing the outcomes, we present some suggestions.

1 INTRODUCTION

Collaborative filtering (CF) methods have been proposed to overcome information overload problem caused by the huge amount of data available on the Internet. CF systems facilitate web personalization needed by e-commerce companies (Riedl, 2001). Such systems support responding for items whose contents cannot be easily analyzed by computational processes (Herlocker et al., 1999).

CF schemes first collect customers' preferences. They then construct an $n \times m$ user-item matrix including users' ratings, where n and m represent number of users and items, respectively. Then, when an active user (a) who is looking for a recommendation for a target item (q) asks a prediction (p_{aq}) from an e-commerce site utilizing CF systems, a's neighborhood is formed. Finally, the weighted average of her neighbors' ratings of q is then presented as a referral. Active users who seek predictions for target items, which they think to buy or taste, request recommendations from CF systems. Besides providing referrals for single items called *prediction*, such schemes also produce a list of items that might be liked by active users called *top-N recommendation*.

Conventional memory-based schemes operate over the entire user database, where entire process is conducted online; and that causes scalability problems in commercial recommender systems. Online computation time exponentially increases with increasing n and/or m. Furthermore, they are directly affected by sparse data. Finally, they suffer from cold start problem in which available data are not enough to produce predictions. This happens especially in the new established e-commerce companies. Such vendors are not able to offer referrals at all because in memory-based schemes, there must be enough commonly rated products between users and sufficient ratings for target items to estimate referrals. Therefore, the coverage is very low. To resolve such problems or overcome the shortcomings of memory-based approaches, model-based techniques have been proposed (Su and Khoshgoftaar, 2009).

In this study, we propose a model-based approach using principal component analysis (PCA). The proposed method is a solution for such limitations that memory-based approaches have. It can overcome the scalability problem by performing CF tasks off-line as much as possible. Sparsity problem can be resolved by filling the empty cells with non-personalized votes. So, even if there is a limited number of users' data, the proposed scheme can offer predictions for all items.

Yakut I., Polat H. and Koc M..
PCF: PROJECTION-BASED COLLABORATIVE FILTERING.
DOI: 10.5220/0003071204080413
In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval* (KDIR-2010), pages 408-413
ISBN: 978-989-8425-28-7

2 RELATED WORKS

”Collaborative filtering” concept first appeared in the *Tapestry* recommender system (Goldberg et al., 1992). Herlocker et al. (Herlocker et al., 1999) present an algorithmic framework for performing CF and evaluate the memory-based algorithms experimentally. Su and Khoshgoftaar (Su and Khoshgoftaar, 2009) present a detailed survey about CF consisting of its background, proposed algorithms, and their contributions and limitations.

To solve scalability and sparsity issues in memory-based systems, many model-based algorithms have been proposed. In some of such model-based schemes, dimensionality reduction techniques have already been applied (Goldberg et al., 2001; Honda et al., 2001; Kim and Yum, 2005; Russell and Yoon, 2008; Sarwar et al., 2000). Goldberg et al. (Goldberg et al., 2001) propose a model-based algorithm called Eigentaste in which items are grouped into gauge and non-gauge sets. Gauge set is absolutely dense part collected ratings for the same items from users via universal queries applied all the users before the prediction generation. It is then reduced in dimensionality by PCA and users are clustered based on the reduced data. Our scheme is different from Eigentaste because our method does not require any universal query or dense gauge set, which may not be always feasible in practice.

Honda et al. (Honda et al., 2001) uses PCA with fuzzy clustering for CF tasks. Their simultaneous approach uses incomplete data set including missing values, which are predicted using the approximation of the data matrix. Although they utilize clustering in their scheme, we apply PCA only to obtain projection matrix. Kim and Yum (Kim and Yum, 2005) propose a scheme that utilizes both singular value decomposition (SVD) and PCA. They fill the missing entries with SVD and perform iterative PCA for generating predictions. Although the accuracy of the recommendations generated using their scheme is better, iterative PCA, clustering a, and computing p_{aq} are conducted online. Thus, online process time of the algorithm is costly. Unlike their scheme, our scheme determines a projection matrix off-line, which is then used to produce referrals online.

Russell and Yoon (Russell and Yoon, 2008) apply discrete wavelet transformation (DWT) to CF systems to improve performance. Sarwar et al. (Sarwar et al., 2000) propose an SVD-based scheme to overcome the problems that memory-based schemes face. Their approach is an effective solution; however, a must be in the generated model as one of the users. Thus, predictions cannot be generated before the model is updated; and the model construction is too costly for online response immediately. If a recommendation should be generated for a new user, the model must be updated, which costly. In our scheme, a does not need to participate in the prediction model. Therefore, our scheme can response to any new comer.

3 PCA-BASED COLLABORATIVE FILTERING

Our method includes both off-line and online computations. During off-line phase, a model is constructed using PCA. In online recommendation process, p_{aq} is estimated from the projection matrix of user-item matrix and a’s ratings vector. PCA is a dimensionality reduction technique that can be widely used in signal processing applications (Zhang et al., 2001) and pattern recognition (Gunal et al., 2005). Obtaining eigenvalues and eigenvectors of covariance matrices for normal distributions is also known as Karhunen-Loeve transforms (KLT). PCA is a special case of KLT subspace analysis. KLT is very effective to find the best representation of a given data set using minimum number of basis vectors (Gunal et al., 2005). It is also known as the optimal linear dimensionality reduction (Bishop, 1996). PCA is a multivariate procedure rotating the data such that maximum variabilities are projected onto the axes. Essentially, a set of correlated variables are transformed into a set of uncorrelated variables, which are ordered by reducing variability. The uncorrelated variables are linear combinations of the original variables, and the last of these variables can be removed with minimum loss of real data. Our goal here is to transform the given data set $\mathbf{x}_i = (x_{1i}, x_{2i}, \ldots, x_{mi})$ for $i = 1,2,\ldots,n$ into $\mathbf{z}_i = (z_{1i}, z_{2i}, \ldots, z_{ki})$ for $i = 1,2,\ldots,n$, where $k < m$. To do so, covariance matrix ($\mathbf{C}$) of the data must be obtained, as follows:

$$\mathbf{C} = \sum_{i=1}^{n} (\mathbf{x}_i - \overline{\mathbf{x}})((\mathbf{x}_i - \overline{\mathbf{x}}))^T, \qquad (1)$$

where $\overline{\mathbf{x}_i} = \frac{1}{n}\sum_{j=1}^{n} \mathbf{x}_i$. After determining $\mathbf{C}$, its eigenvalues ($\lambda_1 > \lambda_2 > \ldots > \lambda_m$) and the corresponding eigenvectors ($\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_m$) are calculated. Since *span* of $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_m\}$ is $\mathbb{R}^m$, $\mathbf{x}_i = \sum_{j=1}^{m} z_j \mathbf{e}_j^T$ for $i = 1,2,\ldots,n$, where $z_j = \mathbf{x}_i \mathbf{e}_j^T$. In the dimension reduction phase, since there is information loss, as much information as possible must be saved. To achieve this, the eigenvectors corresponding to the largest k eigenvalues should be used, as follows:

$$\hat{\mathbf{x}}_i = \sum_{j=1}^{k} z_j \mathbf{e}_j = \sum_{j=1}^{k} (\mathbf{x}_i \mathbf{e}_j^T) \mathbf{e}_j = \sum_{j=1}^{k} (\mathbf{e}_j \mathbf{e}_j^T) \mathbf{x}_i = \mathbf{P}\mathbf{x}_i \quad (2)$$

for $i = 1, 2, \ldots, n$, where $\mathbf{P} = \sum_{j=1}^{k} \mathbf{e}_j \mathbf{e}_j^T$. Selection of the eigenvectors minimizes the error $\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|$. k must be chosen so that the sum of the first largest k eigenvalues is greater than some fixed percentage θ, as follows (Oja, 1983):

$$\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{m} \lambda_i} > \theta. \tag{3}$$

3.1 Our Method

Ratings for various items collected from many users are used in CF tasks. Suppose that **A** is an $n \times m$ matrix including n users' votes for m items, where there are n rows and m columns. To work with PCA properly, there should not be any cells with null values in **A**. However, **A** is not fully dense due to the nature of collection of taste information. Given an entire items set, customers usually rate a limited number of them; that leads to very sparse set. The convenient way to obtain a dense data set is to fill the empty cells with non-personalized or default votes (v_ds). We propose to use row (user) mean, column (item) mean, or distribution of users' ratings to determine v_ds.

Each user can be considered as a feature vector. After filling the sparse feature vectors with v_ds, the resulting dense data matrix is normalized by subtracting the corresponding row averages from each element of matching feature vectors. At the end of normalization, normalized data matrix ($\tilde{\mathbf{A}}$) is obtained. Then, the covariance matrix **C** is computed, as follows:

$$\mathbf{C} = \frac{1}{n-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{A}}. \tag{4}$$

After performing PCA of **C**, eigenpairs (λ_i and $\mathbf{e}_i$) for $i = 1, 2, \ldots, m$ are obtained. **P**, which is an $m \times m$ matrix, is constructed, as follows:

$$\mathbf{P} = \sum_{i=1}^{k} \mathbf{e}_i \mathbf{e}_i^T, \tag{5}$$

where $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k$ are the eigenvectors corresponding to the largest k eigenvalues. Determining the best value of k is a key to the success of our scheme. Thus, we focus on the percentage of information kept by each eigenvalue.

The fixed percentage value (θ) described previously is used to select eigenvectors. Note that θ represents the percentage of variance kept by the selected eigenvectors from the first to the kth one. Since its value is important and can affect our results, we also experimentally test how varying θ values affect the overall performance of our proposed scheme. Once **P** is constructed, the predictive model is obtained. Aforesaid processes discussed so far are performed off-line. The output of such computations is the model or **P**, which is used for providing CF tasks online. Unlike online costs, note that off-line costs are not critical for overall performances. Online phase can described, as follows:

1. An active user a sends her known ratings and a query (for which item (q) she is looking for prediction) to the CF system.
2. The scheme first preprocess her ratings vector by filling the empty cells with v_ds, as explained previously.
3. The preprocessed filled ratings vector called $\boldsymbol{a}$ and **P** then are used to estimate p_{aq}, as follows:

$$p_{aq} = \boldsymbol{a} \cdot \mathbf{P}(q), \tag{6}$$

where $\mathbf{P}(q)$ is the q^{th} column of **P** and $\cdot$ is the scalar dot product.

In addition to Eq.(6), p_{aq} can be estimated using a's normalized ratings vectors, as follows:

$$p_{aq} = \overline{v_a} + \tilde{\boldsymbol{a}} \cdot \mathbf{P}(q), \tag{7}$$

where $\tilde{\boldsymbol{a}}$ includes normalized ratings using deviation from mean approach and $\overline{v_a}$ is the mean rating of a's ratings.

3.2 Analysis of the Proposed Scheme

We analyze the proposed method in terms of accuracy, coverage, and communication, storage, and computation costs. Since off-line costs are not critical for overall performance, we scrutinize our method's online costs. In order to assess the scheme in view of preciseness, we perform various experiments using well-known real data sets explained in the following section.

In traditional memory-based schemes, there should be enough commonly rated items between users and sufficient ratings for q in order to produce predictions. When this is not the case, online vendors, especially the newly established ones, face with coverage and/or cold start problem. We claim that our proposed method responds each request without such problems. Since **P** can be constructed without common item requirements in similarity calculations and generating prediction in conventional CF methods, any request can be addressed. Therefore, it is expected to reply for any request of prediction by any active user a. Moreover, **P** can be updated in particular periods with respect to the number of joined new users and recently released items.

Like in other CF schemes, since a asks a referral and the system returns a result, number of communications is two only. Similarly, amount of data to be transferred is also same as in the similar CF algorithms. A vector including ratings is sent to the CF system; and a value is returned. Thus, in terms of online communication costs (number of communications and amount of data to be sent), our scheme achieves the same performance with other algorithms. Storage cost is in the order of $O(nm)$ because the system stores n users' ratings for m items. This cost is also same with the similar CF algorithms. The only additional storage cost is due to saving v_ds, which are determined off-line to improve online performance. They are used to fill a's empty cells. They are saved in a $1 \times m$ vector. Hence, supplementary storage cost is in the order of $O(m)$. Compared to $O(nm)$, where n is large, this extra storage cost is negligible. It can be considered as adding a new user to the database.

In CF applications, online computation time is critical. In our method, only one scalar dot product is conducted between two vectors with length m during online phase. In other words, m multiplications and $m-1$ summations are performed online for producing each prediction. Since online computation time is in the order of $O(m)$, we can say that our scheme responds each query in linear time. The online cost depends on m. In e-commerce applications, the increase in n is much more larger than the increase in m. Therefore, scalability problems are generally caused by the increase in the number of users. Due to these facts, our method scales practically in CF applications and it is an efficient solution to resolve scalability problems.

4 EXPERIMENTS

To show whether the proposed scheme provides accurate predictions or not, we performed experiments using two well-known real data sets. We conducted trials using Jester and MovieLens Public (MLP) data sets. Jester (Goldberg et al., 2001) is a web-based joke recommendation system. It has 100 jokes and records of 73,421 users. Almost 56% of all possible ratings are present. Thus, it is a very dense set. The ratings range from -10 to +10, and the scale is continuous. MLP was collected by the GroupLens Research Project at the University of Minnesota (web,). Each user has rated at least 20 movies, and ratings are made on a 5-star scale. It consists of 100,000 ratings for 1,682 movies made by 943 users.

As evaluation criteria, we chose the *Mean Absolute Error* (MAE) and the *Normalized Mean Absolute Error* (NMAE) (Goldberg et al., 2001). They are the most common criteria for CF. The MAE is a measure of the deviation of recommendations from their true user-specified values (Sarwar et al., 1998). The MAE is normalized by dividing the difference between the maximum and the minimum ratings so that the NMAE can be obtained. The lower the MAE and the NMAE, the more accurate our results are. Therefore, they should be minimized.

We divided data sets into two disjoint sets: training and testing. Training data are used to create the **P** and we determined the amount of training data according to different experiment settings. In each experiment, we uniformly randomly selected 100 different users from Jester and MLP for testing. Similarly, five ratings were uniformly randomly chosen from available ratings of each test user. For each test item, we withheld their given ratings and tried to predict their values given all other votes using our proposed scheme. We compared the predictions that we found based on our method with the withheld votes. Since train and test sets were determined randomly, we ran our experiments 100 times by using different train test sets. After calculating overall averages of the MAE and NMAE values, we displayed them.

We first performed trials to show how the quality of the referrals changes with varying θ values. Remember that the value of k is determined based on θ. Thus, its value affects accuracy. We used Jester and MLP in these trials. We randomly selected 800 and 1,000 users' data for training from MLP and Jester, respectively. We varied θ values from 1 to 100. To fill empty cells, we used row mean votes as v_ds; and utilized Eq.(6) to estimate predictions. We displayed the MAEs for MLP and Jester in Fig. 1 and Fig. 2, respectively.

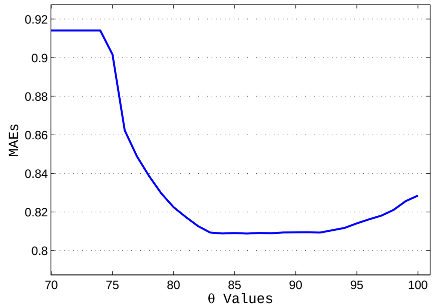

Figure 1: Accuracy vs. θ (MLP data set).

As seen from Fig. 1, the best results are obtained when θ is 86 for MLP, where the MAE is 0.808. We obtained the similar results for θ values ranging from 83 to 92. For MLP, since the energy in the first principal component is 75% of the total energy kept by all the train data, the MAE is the same for θ values from 0 to 74, where the results are the worst due to insufficient information. Accuracy significantly improves with varying θ values from 74 to 83. While it is proper to select any θ value from 83 to 92 for better accuracy, it is preferable to set it at 83 for off-line computational efficiency. Since k is 32 for $\theta = 83$ and it is 132 for $\theta = 92$ in Eq. (5), quarter of train data produces the sam e accuracy. For values of θ larger than 92, accuracy is getting worse due to uncorrelated information (noise and/or redundancy).

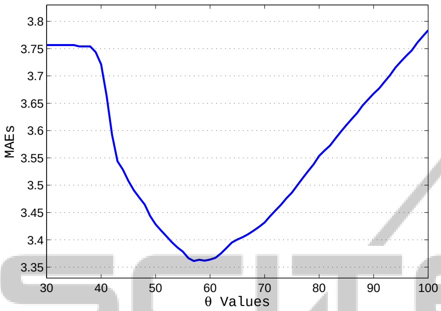

Figure 2: Accuracy vs. θ (Jester data set).

For Jester, as seen from Fig. 2, since 38% of the total energy is concentrated in the first principal component, the MAEs do not change significantly up to $\theta = 38$. The best MAE value, which is 3.361, is obtained when θ is 57. In other words, this value corresponds to k being 9 in Eq. (5). The scheme achieves better results for Jester than the scheme proposed in (Goldberg et al., 2001). Compared to the results in (Goldberg et al., 2001), the improvement is about 11.3% for jester. Accuracy becomes better with increasing θ values from 38 to 57. However, as seen from Fig. 2, MAEs become worse for values of θ larger than 57. For MLP and Jester, the best NMAE values are 0.202 and 0.168, respectively.

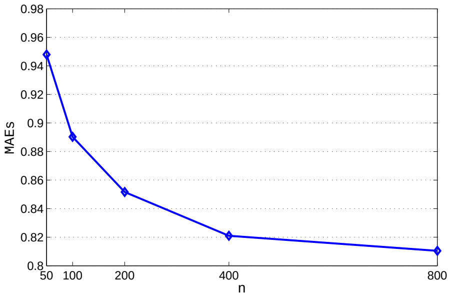

Figure 3: Accuracy vs. n (MLP data set).

We also conducted trials to observe accuracy changes with varying n values for both data sets. We set θ at its optimum values determined for both data sets previously. We used row mean votes to fill empty cells and utilized Eq. (6) to compute referrals. We varied n from 50 to 800 for MLP while changed it from 64 to 4,000 for Jester. After computing overall averages, we displayed them in Fig. 3 and in Fig. 4 for MLP and Jester, respectively.

As seen from Fig. 3, accuracy improves with increasing n values and then becomes stable. For n values larger than 400, the MAE gradually converges to 0.808 for MLP. However, accuracy significantly enhances with increasing n values from 50 to 400. Our results are still promising for n is greater than 400.

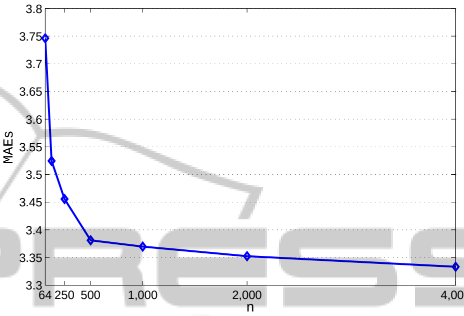

Figure 4: Accuracy vs. n (Jester data set).

Similar trends occur for Jester as seen from Fig. 4. With increasing n values from 64 to 500, MAEs significantly improve. For n values larger than 500, accuracy gradually improves and becomes stable. Although the best results are obtained when n is 4,000, the outcomes are still promising when n is 1,000.

Table 1: Accuracy vs. Different v_ds (MLP Data Set).

v_d	Row	Column	Distribution
MAE	0.8089	1.1935	0.9377
NMAE	0.2022	0.2983	0.2344

We then performed trials to show how different non-personalized rating computation methods affect accuracy. We proposed to use row or column mean, or user's ratings' distribution to determine v_ds. We ran experiments using both data sets while setting θ at its optimum values. For MLP and Jester, we used 800 and 1,000 randomly chosen users' data for training, where we used Eq. (6) to compute referrals. After running the trials 100 times, we computed overall averages and displayed them in Table 1 and Table 2 for MLP and Jester, respectively.

Table 2: Accuracy vs. Different v_ds (Jester Data Set).

v_d	Row	Column	Distribution
MAE	3.3610	3.5793	3.6321
NMAE	0.1681	0.1789	0.1816

As seen from Table 1 and Table 2, the row mean approach outputs the best results for both data sets.

The worst results are obtained for column mean method. Thus, the row mean votes should be used to fill empty cells.

We finally conducted experiments to scrutinize the two methods to estimate predictions. We again used both data sets while randomly chosen 800 and 1,000 users from MLP and Jester, respectively were used for training. We utilized the values and/or methods that give the best results. We displayed the outcomes in Table 3.

Table 3: Accuracy vs. different algorithms.

	MLP		Jester	
Algorithm	Eq. (6)	Eq. (7)	Eq. (6)	Eq. (7)
MAE	0.8089	0.7953	3.3610	3.3704
NMAE	0.2022	0.1988	0.1681	0.1685

As seen from Table 3, the algorithms almost achieve the same results for Jester. However, using Eq. (7) for prediction generation slightly makes accuracy better for MLP.

5 CONCLUSIONS AND FUTURE WORK

We have presented a model-based scheme to provide predictions in linear time with ensuring decent accuracy. We have shown that our scheme is scalable and guarantees coverage. Increasing *n* values do not affect our method's online performance. We have performed real data-based experiments. Our scheme produces better referrals than Eigentaste (Goldberg et al., 2001). It also achieves comparable results with the scheme explained in (Sarwar et al., 2000).

We are planning to investigate how to improve the accuracy of our scheme. Our scheme can be enhanced with some supporting algorithms like clustering, preprocessing, and so on. We will study whether it is possible to offer *top-N recommendation* from the proposed framework. One important issue that should be addressed is the applicability of our scheme to binary data. Finally, considering the increase in web users' privacy concerns, we will also study providing recommendations using the proposed algorithm while preserving users' privacy.

ACKNOWLEDGEMENTS

This work was partially supported by the Grant 108E221 from TUBITAK.

REFERENCES

www.cs.umn.edu/research/Grouplens.

Bishop, C. M. (1996). *Neural Networks for Pattern Recognition*, pages 310–314 and 318–319. Aston University, Birmingham, UK.

Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using collaborative filtering to weave an Information Tapestry. *Communications of ACM*, 35(12):61–70.

Goldberg, K., Roeder, T., Gupta, D., and Perkins, C. (2001). Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151.

Gunal, S., Ergin, S., and Gerek, O. N. (2005). Spam e-mail recognition by subspace analysis. In *Proceedings of INISTA-International Symposium on Innovations in Intelligent Systems and Applications*, pages 307–310, Istanbul, Turkey.

Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. T. (1999). An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 230–237, Berkeley, CA, USA.

Honda, K., Sugiura, N., Ichihashi, H., and Araki, S. (2001). Collaborative filtering using principal component analysis and fuzzy clustering. *Lecture Notes in Computer Science*, 2198:394–402.

Kim, D. and Yum, B. J. (2005). Collaborative filtering based on iterative principal component analysis. *Expert systems with Applications*, 28(4):823–830.

Oja, E. (1983). *Subspace Methods of Pattern Recognition*. John Wiley and Sons Inc., New York, USA.

Riedl, J. T. (2001). Guest editor's introduction: Personalization and privacy. *IEEE Internet Computing*, 5(6):29–31.

Russell, S. and Yoon, V. (2008). Applications of wavelet data reduction in a recommender system. *Expert Systems with Applications*, 34(4):2316–2325.

Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. T. (2000). Application of dimensionality reduction in recommender system–A case study. In *Proceedings of the ACM WebKDD 2000 Web Mining for E-commerce Workshop*, pages 682–693, Boston, MA, USA.

Sarwar, B. M., Konstan, J. A., Borchers, A., Herlocker, J. L., Miller, B. N., and Riedl, J. T. (1998). Using filtering agents to improve prediction quality in the GroupLens research collaborative filtering system. In *Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work*, pages 345–354, Seattle, WA, USA.

Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:1–20.

Zhang, B., Fu, M., and Yan, H. (2001). A nonlinear neural network model of mixture of local principal component analysis: Application to handwritten digits recognition. *Pattern Recognition*, 34:203–214.