Applying SVD on Item-based Filtering

Manolis G. Vozalis and Konstantinos G. Margaritis
Parallel Distributed Processing Laboratory
Department of Applied Informatics, University of Macedonia
Egnatia 156, P.O. 1591, 54006, Thessaloniki, Greece
E-mail: {mans,kmarg}@uom.gr
URL: http://macedonia.uom.gr/~{mans,kmarg}

Abstract

RAAWS: In this paper we will examine the use of a matrix factorization technique called Singular Value Decomposition (SVD) in Item-Based Collaborative Filtering. After a brief introduction to SVD and some of its previous applications in Recommender Systems, we will proceed with a full description of our algorithm, which uses SVD in order to reduce the dimension of the active item's neighborhood. The experimental part of this work will first locate the ideal parameter settings for the algorithm, and will conclude by contrasting it with plain Item-based Filtering which utilizes the original, high dimensional neighborhood. The results show that a reduction in the dimension of the item neighborhood is promising, since it does not only tackle some of the recorded problems of Recommender Systems, but also assists in increasing the accuracy of systems employing it.

keywords*: Recommender Systems, Item-based Collaborative Filtering, Personalization, Singular Value Decomposition (SVD)*

1 Introduction

Recommender Systems were introduced as a computer-based intelligent technique that assists people with the problem of information and product overload, their aim being to provide efficient personalized solutions in e-business domains, that would benefit both the customer and the merchant. They feature two basic entities: the *user* and the *item*. A user who utilizes the Recommender System, provides his opinion about past items. The purpose of the Recommender System is to generate suggestions about new items for that particular user. The process is based on the input provided, which is usually expressed in the form of ratings from that user, and the filtering algorithm, which is applied on that input.

Recommender Systems suffer from a number of fundamental problems that cause a reduction in the quality of the generated predictions, such as *sparsity*, *scalability*, and *synonymy*. A number of solutions have been introduced, intending to solve those problems [7, 3, 10]. We will focus on the case of Singular Value Decomposition which only recently was proposed by various Recommender Systems researchers as a possible means of alleviating the aforementioned problems. The algorithm we present in this paper utilizes SVD as a technique that is able to effectively reduce the dimension of the user-item data matrix, and then it executes Item-based Filtering with this low rank representation to generate its predictions.

The following sections are structured as follows: We first introduce the concept of Singular Value Decomposition. Then, we present some Recommender Systems which have employed SVD in order to improve their performance. We include a brief discussion of the collaborative filtering algorithm which was utilized to support our technique, and provide an analysis of our experimental methodology, regarding the data set and evaluation metrics. At this point we give a step-by-step description of the proposed filtering method. The experimental work which follows, closes by comparing our technique with the base filtering algorithm. We conclude with a summary and possible directions for future work.

2 Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) is a matrix factorization technique which takes an $m \times n$ matrix A, with rank r, and decomposes it as follows:

$$SVD(A) = U \times S \times V^T$$

U and V are two orthogonal matrices with dimensions $m \times m$ and $n \times n$ respectively. S, called the *singular matrix*, is an $m \times n$ diagonal matrix whose diagonal entries are non-negative real numbers.

The initial r diagonal entries of S $(s_1, s_2, \ldots, s_r)$ have the property that $s_i > 0$ and $s_1 \geq s_2 \geq \ldots \geq s_r$. Accordingly, the first r columns of U are eigenvectors of AA^T and represent the left singular vectors of A, spanning the column space. The first r columns of V are eigenvectors of A^TA and represent the right singular vectors of A, spanning the row space. If we focus only on these r nonzero singular values, the effective dimensions of the SVD matrices U, S and V will become $m \times r$, $r \times r$ and $r \times n$ respectively.

An important attribute of SVD, particularly useful in the case of Recommender Systems, is that it can provide the best low-rank approximation of the original matrix A. By retaining the first $k << r$ singular values of S and discarding the rest, which can be translated as keeping the k *largest* singular values, based on the fact that the entries in S are sorted, we reduce the dimensionality of the data representation and hope to capture the important "latent" relations existing but not evident in the original representation of matrix A. The resulting diagonal matrix is termed S_k. Matrices U and V should be also reduced accordingly. U_k is produced by removing $r - k$ columns from matrix U. V_k is produced by removing $r - k$ rows from matrix V. Matrix A_k which is defined as:

$$A_k = U_k \times S_k \times V_k^T$$

stands for the closest linear approximation of the original matrix A with reduced rank k. Once this transformation is completed, users and items can be represented as points in the k-dimensional space.

2.1 Using SVD in Recommender Systems

SVD, as part of *Latent Semantic Indexing (LSI)*, was widely used in the area of **Information Retrieval** [4] in order to solve the problems of *synonymy*, which is a result of the many possible ways to express a known concept, and *polysemy*, which comes from the fact that most words usually have multiple meanings. Furthermore, techniques like *SVD-updating* or *folding-in* were proposed to alleviate the problem of *updating*, which refers to the process of new terms and/or documents being added to existing matrices [1].

Those ideas were adopted by researchers in the area of Recommender Systems. Initially, Billsus and Pazzani [2] took advantage of the properties of SVD in their attempts to formulate collaborative filtering as a classification problem. In their work they utilize SVD as a dimensionality reduction technique, before they feed their data matrix, which is now represented in the reduced feature space, into an Artificial Neural Network (ANN). This ANN, or, as the authors claim, any alternative Machine Learning algorithm, can then be trained in order to generate predictions.

GroupLens have also applied SVD in at least 3 distinct cases regarding Recommender Systems: (i) in an approach that reduces the dimensionality of the user-item space and forms predictions in that reduced space, by not building explicit neighborhoods at any point of the procedure, (ii) in an approach that generates a user-neighborhood in the SVD reduced space and then applies normal user-based collaborative filtering on it, and (iii) in an approach that aims at increasing the scalability by applying folding-in for the incremental computation of the user-item model [9, 8].

Finally, Goldberg et al. use Principal Component Analysis [5], a technique very similar to SVD, in order to optimally project highly correlated data along a smaller number of orthogonal dimensions. Then, their Eigentaste algorithm clusters the projected data, being concluded by an online computation of recommendations.

2.2 The Base Algorithm: Item-based Filtering

In this section we will briefly discuss the filtering algorithm which will be utilized by the proposed technique. Similarly to User-based Collaborative Filtering, Item-based Filtering is based on the creation of neighborhoods. Yet, unlike the User-based Collaborative Filtering approach, those neighbors consist of similar items rather than similar users [11].

The execution steps of the algorithm are (a) *Data Representation* of the ratings provided by m users on n items. This step is based on the construction of an $m \times n$ user-item matrix, R. (b) *Neighborhood Formation*, which concludes by the construction of the active item's neighborhood. Similarities for all possible pairs of items existing in the data set are computed by the application of the preferred similarity metrics. Items most similar to the active item, which refers to the item for which predictions should be generated, are selected for its neighborhood. (c) *Prediction Generation*, where final predictions are calculated as a weighted sum of ratings given by a user on all items included in the active item's neighborhood.

2.3 Experimental Methodology

For the execution of our subsequent experiments we utilized the data publicly available from the GroupLens movie recommender system. The MovieLens data set consists of 100.000 ratings which were assigned by 943 users on 1682 movies. Users should have stated their opinions for at least 20 movies in order to be included. Ratings follow the 1(bad)-5(excellent) numerical scale. Starting from the initial data set five distinct splits of training and test data were generated.

Several techniques have been used to evaluate Recommender Systems. Those techniques have been divided by Herlocker et al. [6] into three categories: *Predictive Accuracy Metrics*, *Classification Accuracy Metrics*, and *Rank Accuracy Metrics*. The choice among those metrics should be based on the selected user tasks and the nature of the data sets. We wanted our proposed algorithms to derive a predicted score for already rated items rather than generate a top-N recommendation list. Based on that specific task we proceeded in the selection of the initial evaluation metric for our experiments. That metric was *Mean Absolute Error (MAE)*. It is a statistical accuracy metric which measures the deviation of predictions, generated by the Recommender System, from the true rating values, as they were specified by the user. MAE is measured only for those items, for which a user has expressed his opinion.

3 The Algorithm: Item-based Filtering Enhanced by SVD

We will now describe the steps of how SVD can be combined with Item-based Filtering in order to make the base algorithm more scalable.

1. Define the original user-item matrix, R, of size $m \times n$, which includes the ratings of m users on n items. r_{ij} refers to the rating of user u_i on item i_j.

2. Preprocess user-item matrix R in order to eliminate all missing data values. The preprocessing is described in detail here:

   (a) Compute the average of all rows, r_i, where $i = 1, 2, ..., m$, and the average of all columns, r_j, where $j = 1, 2, ..., n$, from the user-item matrix, R.

   (b) Replace all matrix entries that have no values, denoted by $\perp$, with the corresponding *column* average, r_j, which leads to a new filled-in matrix, $R_{filled-in}$.

   (c) Subtract the corresponding *row* average, r_i, from all the slots of the new filled-in matrix, $R_{filled-in}$, and obtain the normalized matrix R_{norm}.

3. Compute the SVD of R_{norm} and obtain matrices U, S and V, of size $m \times m$, $m \times n$, and $n \times n$, respectively. Their relationship is expressed by: $R_{norm} = U \cdot S \cdot V^T$.

4. Perform the dimensionality reduction step by keeping only k diagonal entries from matrix S to obtain a $k \times k$ matrix, S_k. Similarly, matrices U_k and V_k of size $m \times k$ and $k \times n$ are generated. The "reduced" user-item matrix, R_{red}, is obtained by $R_{red} = U_k \cdot S_k \cdot V_k^T$, while rr_{ij} denotes the rating by user u_i on item i_j as included in this reduced matrix.

5. Compute $\sqrt{S_k}$ and then calculate two matrix products: $U_k \cdot \sqrt{S_k}^T$, which represents m users and $\sqrt{S_k} \cdot V_k^T$, which represents n items in the k dimensional feature space. We are particularly interested in the latter matrix, of size $k \times n$, whose entries represent the "meta" ratings provided by the k pseudo-users on the n items. A "meta" rating assigned by *pseudo*-user u_i on item i_j is denoted by mr_{ij}.

6. Proceed with Neighborhood Formation which can be broken into two substeps:

   (a) Calculate the similarity between items i_j and i_f by computing their Adjusted Cosine Similarity as follows:

   $$sim_{jf} = adjcorr_{jf} = \frac{\sum_{i=1}^{k} mr_{ij} \cdot mr_{if}}{\sqrt{\sum_{i=1}^{k} mr_{ij}^2 \sum_{i=1}^{k} mr_{if}^2}}$$

   where k is the number of *pseudo*-users, selected when performing the dimensionality reduction step. We have to note a change between the Adjusted Cosine Similarity equation utilized in plain Item-based Filtering and here. In plain Item-based Filtering the difference in rating scale between distinct users was offset by subtracting the corresponding user average from each co-rated pair of items. In SVD-enhanced Item-based Filtering, that difference in rating scale was offset during the normalization of the original user-item matrix which yielded matrix R_{norm}.

   (b) Based on the results from the Adjusted Cosine Similarity calculations for pairs of items including the active item and a random item, isolate the set of items which appear to be the most similar to the active item.

7. Conclude with Prediction Generation, achieved by the following weighted sum:

$$pr_{aj} = \frac{\sum_{k=1}^{l} sim_{jk} * (rr_{ak} + \bar{r_a})}{\sum_{k=1}^{l} |sim_{jk}|}$$

which calculates the prediction for user u_a on item i_j. It is similar to the equation utilized by plain Item-based Filtering in that it bases its predictions on the ratings given by the active user, u_a, on the l items selected as the most similar to active item i_j. Yet, it is different in that the user ratings are taken from the reduced user-item matrix, R_{red}. Also, we have to add the original user average, $\bar{r_a}$, back since it was subtracted during the normalization step of the preprocessing.

4 Benefits of Application

A Recommender System running Item-based Filtering with a lower dimensional representation, as described in the previous section, will benefit in the following ways:

- The complexity of Item-based Filtering, utilizing the original data representation, is $O(mn^2)$. By reducing the dimension to k, where $k << m$, the complexity becomes $O(kn^2)$. We can assume that this reduction in complexity will improve the scalability of the system, while both the processing time and storage requirement should also move down.
- Based on the properties of SVD, any latent relations between users and items should be located when employing the low rank data representation.
- Before the main part of the algorithm is executed, during its preprocessing phase, all the empty entries of the user-item matrix are filled. As a result, once the execution is completed, the n items, taken from the original data array, have now been rated by *all*, k, *pseudo*-users. This means that the sparsity problem is solved and the achieved coverage for the Recommender System is always equal to 100%.

Still, we are interested to find out if the benefits from applying Item-based Filtering on a low-dimensional neighborhood are also extended to the accuracy of the generated predictions. To achieve that we have set up a number of experiments which will be discussed in detail in the following paragraphs.

5 Experiments

The aim of the experiments which will be described in the following sections is first to locate the optimal parameter settings for the Item-based enhanced by SVD filtering algorithm. Once those settings are found and applied, we will compare the results with the predictions generated by a plain Item-based Recommender System running also with optimal settings.

5.1 Locating the optimal value for reduced dimension, k

As mentioned earlier, k refers to the number of singular values retained from the singular matrix S. As a result, k also corresponds to the rank of the reduced user-item matrix R_{red}, and also to the number of *pseudo*-users which are used, instead of the actual m users, in order to represent the n items.

The number of dimensions selected for the reduced space representation, as expressed by the value of k, is significant for the efficiency of the Recommender System which incorporates this representation for its data. The number of dimensions should be rather small in order to actually lead to an improvement in the filtering algorithm's scalability and at the same time to exclude any over-fitting errors. On the other hand, it should be big enough in order to capture any latent relations among the users or the items included in the original data matrix, R.

By this experiment we wanted to determine the ideal value of this dimension. We kept the size of the active item's neighborhood fixed to 60, and ran our algorithm repeatedly for different values of k, $k = \{2, 4, 6, 8, 10, 15, 20, 25\}$. Figure 1 collects the Mean Absolute Errors observed by those runs, averaged over the 5 data splits from the data set.

Our results indicate that applying Item-based Filtering on a neighborhood of reduced dimension displays a similar behavior to utilizing a low dimensional representation in User-based Filtering. The quality of our Recommender System's predictions follows the pattern reported for the ML data set in Sarwar [9]. Specifically, at first, the predictions' accuracy improves as we increase the rank of the lower dimension space and quickly reaches its peak for $k = 6$. For any further increase in the rank, the results keep getting worse. It appears that the size of the data set and its sparsity allow for existing latent relations among items to be discovered for rather low rank values, while the over-fitting of the data is evident when the value of k increases. Consequently, in our subsequent experiments we will utilize a fixed value of $k = 6$.

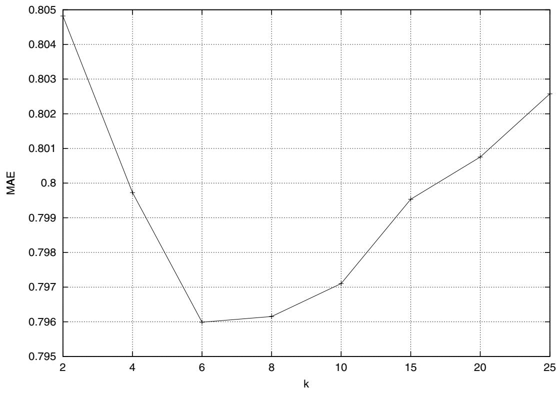

Figure 1. Average MAE for various k values

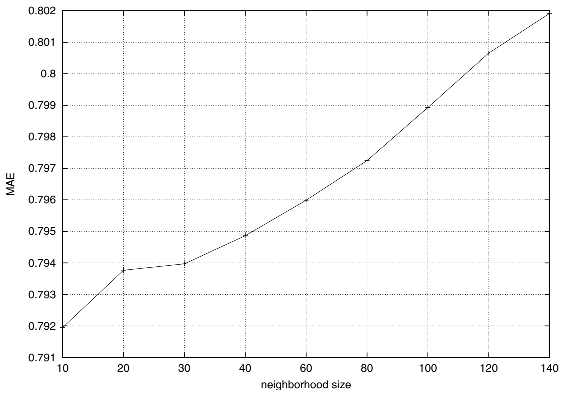

Figure 2. Average MAE for various neighborhood sizes

5.2 Locating the optimal value for the neighborhood size

As reported in previous work [12], the size of the neighborhood greatly affects the behavior of Collaborative Filtering algorithms. It is a common trend for such algorithms to show an initial improvement in accuracy, as the neighborhood size increases, reach their maximum performance for a specific threshold, and remain stable or show a slight decline after that threshold.

For our experiments, we kept the dimension of the lower rank representation fixed to 6, retaining the ideal value for k as recorded in the preceding section, and varied the size of the active item's neighborhood, *neigh-size={10-140}*. The errors observed from the runs of our algorithm under those settings, averaged over the 5 data splits of the data set, are collected in Figure 2.

Other researchers have reported that the ideal number of neighbors in a collaborative filtering algorithm is data set dependent [9]. Furthermore, experimental results, involving neighborhoods of varying sizes [12], have indicated that Item-based Filtering reaches its peak performance for quite smaller neighborhoods than those required for similar experiments in User-based Filtering. In that sense, the behavior recorded in Figure 2 is quite interesting. The accuracy of Item-based Filtering, with its original user-item matrix reduced by SVD, starts with a rather low error value for a neighborhood including 10 items, and gets steadily worse for neighborhoods whose size continuously increases, concluding with a maximum size of 140 items. As a result, according to the findings of this experiment we can assume that a neighborhood including only 10 items, which are the most similar to the active item, is able to generate the most accurate predictions.

Table 1. Average MAEs for both neighborhood dimensions

	high-dimensional Item-based	low-dimensional Item-based
MAE	0.83987391	0.79658809

5.3 Comparing Item-based Filtering enhanced by SVD with plain Item-based Filtering

Once the optimal settings for our algorithm were located, we were able to proceed with our final experiment. The purpose of this experiment was to evaluate the accuracy of Item-based Filtering, when utilizing a low-dimensional neighborhood, by contrasting it with Item-based Filtering employing the original, high-dimensional neighborhood. The parameters in both implementations were set for optimal predictions.

Figure 3 includes the Mean Absolute Errors for high (*ib*) and low (*svd-ib*) dimensions, as observed for each of the 5 data splits of the data set. These error values are then averaged and Table 1 records the final results for both implementations.

From both the preceding figure and table, we can conclude that applying Item-based Filtering on the low-rank neighborhood, provides a clear improvement over the higher dimension neighborhood. The interesting part is that this improvement, owed to the low-rank representation, is not limited only to the solution of the sparsity problem, or to the improvement of the scalability of the Recommender System. This improvement can also be measured by a considerable increase in the values of the achieved accuracy.

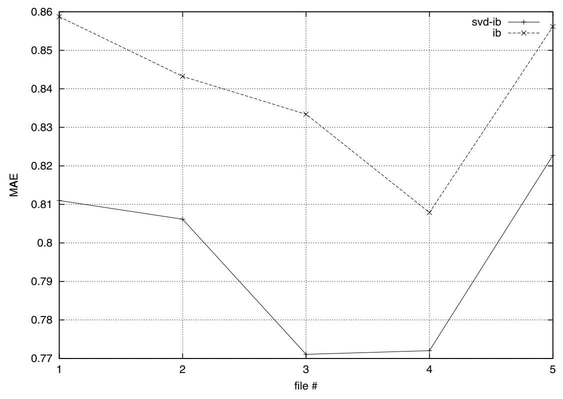

Figure 3. Comparing Item-based Filtering enhanced by SVD with plain Item-based Filtering

6 Conclusions and Future Work

In the past, Singular Value Decomposition was mainly combined with User-based Collaborative Filtering, proving to be an effective solution for recorded problems of Recommender Systems such as scalability and sparsity. By this work we extended the application of SVD to Item-based Collaborative Filtering. We gave a detailed description of the way SVD can be utilized in order to reduce the dimension of the user-item representation, and, afterwards, how this low-rank representation can be employed in order to generate item-based predictions.

A number of experiments were set up to check how this algorithm fares against Item-based filtering, running in collaboration with the original data representation. The results showed that low-dimension Item-based Filtering not only alleviates problems like scalability or sparsity of the data, but also proves to be a more accurate recommender than the original Item-based Filtering.

After those promising results, it is in our intentions to attempt a possible collaboration of SVD with a couple of hybrid filtering techniques which were proposed recently [13]. These techniques incorporate demographic data about users or items in the recommendation procedure in order to generate their predictions, and it would be interesting to see whether SVD can actually locate latent relations among the featured demographic vectors. Furthermore, Artificial Neural Networks employing pseudo-SVD as a possibly simpler, faster and less costly solution, would be an interesting alternative that we can compare our algorithm against.

References

[1] M. W. Berry, S. T. Dumais, and G. W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37:573–595, 1995.

[2] D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *15th International Conference on Machine Learning*, Madison, WI, 1998.

[3] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper. In *ACM SIGIR Workshop on Recommender Systems-Implementation and Evaluation*, Berkeley, CA, 1999.

[4] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[5] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval Journal*, 4:133–151, 2001.

[6] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22:5–53, 2004.

[7] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering. In *ACM SIGIR Workshop on Recommender Systems*, New Orleans, LA, 2001.

[8] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Incremental singular value decomposition algorithms for highly scalable recommender systems. In *Fifth International Conference on Computer and Information Technology (ICCIT 2002)*, 2002.

[9] B. M. Sarwar. *Sparsity, Scalability, and Distribution in Recommender Systems.* PhD thesis, University of Minnesota, 2001.

[10] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Analysis of recommendation algorithms for e-commerce. In *Electronic Commerce*, 2000.

[11] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Item-based collaborative filtering recommendation algorithms. In *10th International World Wide Web Conference (WWW10)*, Hong Kong, 2001.

[12] E. G. Vozalis and K. G. Margaritis. Recommender systems: An experimental comparison of two filtering algorithms. In *Proceedings of the 9th Panhellenic Conference in Informatics - PCI 2003*, 2003.

[13] M. Vozalis and K. G. Margaritis. Enhancing collaborative filtering with demographid data: The case of item-based filtering. In *Proceedings of the fourth IEEE International Conference on Intelligent Systems Design and Applications*, Budapest, Hungary, 2004.