

Incremental Singular Value Decomposition Algorithms for Highly Scalable Recommender Systems

JajimoggaRaghavendar
Assoc Professor
NallaMalla Reddy Engineering College
Telangana India

Abstract

We investigate the use of dimensionality reduction to improve the performance for a new class of data analysis software called “recommender systems”. Recommender systems apply knowledge discovery techniques to the problem of making personalized product recommendations during a live customer interaction. The tremendous growth of customers and products in recent years poses some key challenges for recommender systems. These are: producing high quality recommendations and performing many recommendations per second for millions of customers and products. Singular Value Decomposition(SVD)-based recommendation algorithms can quickly produce high quality recommendations, but has to undergo very expensive matrix factorization steps. In this paper, we propose and experimentally validate a technique that has the potential to incrementally build SVD-based models and promises to make the recommender systems highly scalable.

Introduction

Recommender systems have evolved in the extremely interactive environment of the Web. They apply data analysis techniques to the problem of helping customers find which products they would like to purchase at Ecommerce sites. These systems, especially the collaborative filtering based ones [3, 5, 7, 8, 11], are rapidly becoming a crucial tool in E-commerce on the Web. Nowadays, they are being stressed by the huge volume of customer data in existing corporate databases, and will be stressed even more in future by the increasing volume of customer data available on the Web. The tremendous growth of customers and products poses two key challenges for recommender systems. The first challenge is to improve the quality of the recommendations for the consumers. Consumers need recommendations they can trust to help them find products they will like. If a

consumer trusts a recommender system purchases a product, and finds out he does not like the product, the consumer will be unlikely to use the recommender system again. Another challenge is to improve the scalability of the collaborative filtering algorithms. These algorithms are able to search tens of thousands of potential neighbors in real-time, but the demands of modern E-commerce systems are to search tens of millions of potential neighbors

In some ways these two challenges are in conflict, since the less time an algorithm spends searching for neighbors, the more scalable it will be, and the worse its quality. For this reason, it is important to treat the two challenges simultaneously so the solutions discovered are both useful and practical. New technologies are needed that can dramatically improve the scalability of recommender systems. Researchers [1, 4, 9, 10] suggest that Singular Value Decomposition (SVD) may be such a technology in

some cases. SVD-based approach produced results that were better than a traditional collaborative filtering algorithm most of the time when applied to a Movie data set [9]. However, SVD-based recommender systems suffer one serious limitation that makes them less suitable for large-scale deployment in E-commerce. The matrix factorization step associated with these systems is computationally very expensive and is a maj

In this paper, we experiment with an incremental model-building technique for generating SVD-based recommendations that has the promise of being highly scalable while producing good predictive accuracy. In particular, we present an algorithm that builds upon a small pre-computed SVD model and provides larger SVD models using inexpensive techniques. Our experimental results suggest that the overall algorithm works twice

The rest of the paper is organized as follows. The next section gives a brief overview of the SVDbased prediction generation process and discusses its promises and challenges. Section 3 outlines our incremental SVD algorithm. Section 4 presents our experimental procedure, results and discussion. The final section provides some concluding remarks and future research directions

Dimensionality Reduction for Collaborative Filtering

In this section we briefly discuss how one dimensionality reduction technique can potentially be used for prediction generation. We then present some of the challenges of such algorithms and propose an incremental technique to make them highly scalable

Promises of Dimensionality Reduction

The goal of CF-based recommendation algorithms [7, 8, 11] is to suggest new products or to predict the utility of a certain product for a particular customer, based on the customer's previous liking and the opinions of other like-minded customers. These systems have been successful in

several domains. However, in our earlier papers [9, 10], we mentioned some limitations of these systems, namely sparsity, scalability, and synonymy. The weakness of CF algorithms for large, sparse databases led us to explore alternative recommender system algorithms. After reviewing several techniques, we decided to try applying Latent Semantic Indexing (LSI) to reduce the dimensionality of our customer-product ratings matrix. LSI is a dimensionality reduction technique that has been widely used in information retrieval (IR) to solve the problems of synonymy and polysemy [2]. LSI, which uses singular value decomposition (SVD) as its underlying dimensionality reduction algorithm, maps nicely into the collaborative filtering recommender algorithm challenge.

Singular Value Decomposition (SVD).

SVD is a matrix factorization technique commonly used for producing low-rank approximations. Given an $m \times n$ matrix A , with rank r , the singular value decomposition, $SV D(A)$, is defined as

Where U , S and V are of dimensions $m \times m$, $m \times n$, and $n \times n$, respectively. Matrix S is a diagonal matrix having only r nonzero entries, which makes the effective dimensions of these three matrices $m \times r$, $r \times r$, and $r \times n$, respectively. U and V are two orthogonal matrices and S is a diagonal matrix, called the singular matrix. The diagonal entries (s_1, s_2, \dots, s_r) of S have the property that $s_i > 0$ and $s_1 \geq s_2 \geq \dots \geq s_r$. The first r columns of U and V represent the orthogonal eigenvectors associated with the r nonzero eigenvalues of $A A^T$ and $A^T A$, respectively. In other words, the r columns of U corresponding to the nonzero singular values span the column space, and the r columns of V

span the row space of the matrix A . U and V are called the left and the right singular vectors, respectively. SVD has an important property that makes it particularly interesting for our application. SVD provides the best low-rank linear approximation of the original matrix A . It is possible to

retain only k singular values by discarding other entries. We term this reduced matrix S_k . Since the entries in S are sorted i.e., $s_1 \geq s_2 \geq \dots \geq s_r$, the reduction process is performed by retaining the first k singular values. The matrices U and V are also reduced to produce matrices U_k and V_k , respectively. The matrix U_k is produced by removing $(r - k)$ columns from the matrix U and matrix V_k is produced by removing $(r - k)$ rows from the matrix V . When we multiply these three reduced matrices, we obtain a matrix A_k . The reconstructed matrix $A_k = U_k S_k V_k^T$ is a rank- k matrix that is the closest approximation to the original matrix A . More specifically, A_k minimizes the Frobenius norm $\|A - A_k\|_F$ over all rank- k matrices. Researchers [1, 2] pointed out that the low-rank approximation of the original space is better than the original space itself due to the filtering out of the small singular values that introduce “noise” in the customer-product relationship. The dimensionality reduction approach in SVD can be very useful for the collaborative filtering process. SVD produces a set of uncorrelated eigenvectors. Each customer and product is represented by its corresponding eigenvector. The process of dimensionality reduction may help customers who rated similar products (but not exactly the same products) to be mapped into the space spanned by the same eigenvectors. We now present an outline of the prediction generation algorithm using SVD (see [9] for details)

In this paper, we experiment with an incremental model-building technique for generating SVD-based recommendations that has the promise of being highly scalable while producing good predictive accuracy. In particular, we present an algorithm that builds upon a small pre-computed SVD model and provides larger SVD models using inexpensive techniques. Our experimental results suggest that the overall algorithm works twice as fast while producing similar prediction accuracy

In this section we briefly discuss how one dimensionality reduction technique can potentially be used for prediction

generation. We then present some of the challenges of such algorithms and propose an incremental technique to make them highly scalable

The dimensionality reduction approach in SVD can be very useful for the collaborative filtering process. SVD produces a set of uncorrelated eigenvectors. Each customer and product is represented by its corresponding eigenvector. The process of dimensionality reduction may help customers who rated similar products (but not exactly the same products) to be mapped into the space spanned by the same eigenvectors. We now present an outline of the prediction

The singular value decomposition of a matrix has many applications. Here I'll focus on an introduction to singular value decomposition and an application in clustering articles by topic. In another notebook ([link](#)) I show how singular value decomposition can be used in image compression

users that had rated 20 or more movies). The data set was converted into a user-movie matrix R that had 943 rows(users) and 1682 columns (movies).

For our experiments, we divided the data set into a training and a test portion. We varied the training and test data ratio by using a parameter x , where $x = 0.8$ means that 80% data was used for training the algorithm and 20% was used as test. Evaluation metric. For our experiments, we use a widely popular statistical accuracy metric named Mean Absolute Error (MAE), which is a measure of the deviation of recommendations from their true userspecified values [5, 11]. For each ratings-prediction pair $\langle p_i, q_i \rangle$, this metric treats the absolute error between them i.e., $|p_i - q_i|$ equally. The MAE is computed by first summing these absolute errors of the N corresponding ratings-prediction pairs and then computing the average. Formally, $MAE = \frac{1}{N} \sum_{i=1}^N |p_i - q_i|$.

The lower the MAE, the more accurately the recommendation engine predicts user ratings. Environment. All our experiments were done using a combination of MATLAB and C, running on a Linux platform. The machine had 650 MHz Intel Pentium III CPU with 256 MB of RAM, and 512 KB of cache memory

Experimental Procedure

For this experiment, we use the prediction generation algorithm using SVD described in [9], but instead of computing the SVD model (decomposition of matrix A into matrices U , S , and V) for all users, we use a threshold size to build an initial model and then use the folding-in technique to incrementally compute the SVD model for additional users. Before performing the prediction experiments, we first determine the optimal values of our two experimental parameters—i) the number of dimensions k , and ii) the threshold model size (basis size). We then perform the folding-in step and generate predictions using the incremental model. Finally, we investigate the performance implications of the folding-in technique. We used 10-fold cross validation by selecting random training and test data for all our experiments

Prediction quality experiments. Figure 4(a) plots the prediction quality results with the incremental model building using folding-in technique. We report our results at three different training-test ratios— at $x = 0.2, 0.5,$ and 0.8 . For each experiment, we start with a model size equal to the threshold basis size for folding-in. Then we use the projection method to fold in the rest of $(943 - \text{basis})$ users onto the SVD space. This process gives us an approximate SVD model for 943 users, where unlike the original SVD model the component matrices U , S , and V are not orthogonal. We then use the component matrices to generate prediction for a test set of ratings. We start with a model size of 600 and go up to 900 with an increment of 50. The plots of Figure 3 show that even with a small basis size it is possible to obtain a good quality. The MAE value at

$x = 0.8$, for example, is 0.733 for the full model size and 0.742 for a model size of 600 (only 1.22% quality drop!). Similar numbers can also be found at other x values. This suggests that the inexpensive projection technique provides good quality even with a small basis size. Performance implications The observation from Figure 4 that the quality does not change dramatically with varying model size suggests that the SVD prediction generation system can be made more scalable by using the folding-in method. To investigate these scalability impacts, we record the run-time in seconds for each run and from there, we compute the throughput performance metric. The throughput plot, presented in Figure 4(b), shows the number of predictions generated per second at different basis sizes. From the plot corresponding to $x = 0.8$ and the basis size of 600, we have a test case size of $(100,000 * (1 - 0.8))$. This means that our algorithm generates 20,000 ratings in 408.27 seconds, from there we obtain a throughput rate of 88.82 recommendations per second. Accordingly, at a full model size of 943 the throughput becomes 48.9 (81.63% performance gain!). This difference is even more prominent at lower values of x , where the workload size is larger. Overall, the folding-in technique shows the potential to be very useful in addressing the scalability challenge of SVD-based prediction generation systems. We have demonstrated that although folding-in results in slight quality loss due to the non-orthogonality of the resultant space, it shows substantial performance gain

Conclusion and Future Work

SVD-based recommendation generation technique leads to very fast online performance, requiring just a few simple arithmetic operations for each recommendation but computing the SVD is very expensive. Use of incremental SVD algorithms such as folding-in [1] can significantly speed up the SVD computation cost while providing comparable prediction quality. In this paper, we have demonstrated that incremental SVD algorithms, based on folding-in, can

help recommender systems achieve high scalability while providing good predictive accuracy

References

[1] Berry, M. W., Dumais, S. T., and O'Brian, G. W. (1995). Using Linear Algebra for Intelligent Information Retrieval. *SIAM Review*, 37(4).

[2] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*. 41(6).

[3] Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM*. December.

[4] Gupta, D., and Goldberg, K. (1999). Jester 2.0: A Linear Time Collaborative Filtering Algorithm Applied to Jokes. In *Proc. of the ACM SIGIR '99*.

[5] Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. (1999). An Algorithmic Framework for Performing Collaborative Filtering. In *Proc. of ACM SIGIR'99*.

[6] Hill, W., Stead, L., Rosenstein, M., and Furnas, G. (1995). Recommending and Evaluating Choices in a Virtual Community of Use. In *Proc. of CHI '95*.

[7] Resnick, P. and Varian, H. R. (1997). Recommender Systems. *Communications of the ACM*. 40(3), pp. 56-58.

[8] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proc. of CSCW '94*, Chapel Hill, NC.

[9] Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. (2000). Application of Dimensionality Reduction in Recommender System—A Case Study. In *ACM WebKDD'00*

(Web-mining for ECommerce Workshop). [10] Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. (2000). Analysis of Recommendation Algorithms for E-Commerce. In *Proc. of the ACM EC'00 Conference*. Minneapolis, MN, pp. 158-167.

[11] Shardanand, U., and Maes, P. (1995). Social Information Filtering: Algorithms for Automating 'Word of Mouth'. In *Proc. of CHI '95*. Denver, CO. [12] Zha, H., and Zhang, Z. (1999). On matrices with Low-Rank-Plus-Shift Structure: Partial SVD and Latent Semantic Indexing. *SIAM Jou*