Hybrid Recommender System based on Collaborative Filtering*

†

Mohamed Farhan
University of Central Florida
Orlando, Florida
mfarhan@knights.ucf.edu

Pushkar
University of Central Florida
Orlando, Florida
pushkar1992m@knights.ucf.edu

Mithun Mohanraj
University of Central Florida
Orlando, Florida
mohanmithun005@knights.ucf.edu

ABSTRACT

The goal of the work in this paper is towards the integration of item-based collaborative filtering and SVD-like matrix factorization technique to build a hybrid recommender system that delivers better personalized recommendation results.

KEYWORDS

collaborative filtering, recommender systems , hybrid recommendation system, SVD (Singular Value Decomposition), Latent Semantic Indexing

1 INTRODUCTION

Recommender systems are based on a set of algorithms that estimate the user interest in given items or products to deliver personalized recommendations for items that will be appropriate with the user's taste. Generally, recommender systems attempt to match user preferences with the interaction between users and products.

There are efforts being directed into developing sophisticated recommender systems with the growing popularity of e-commerce. The huge economic prospective led some of the biggest e-commerce giants, for example web merchant Amazon.com and the online movie rental company Netflix, make the recommender system a significant part of their web sites. High quality personalized recommendations improves the user experience to great extent.

Generally, recommender systems are designed on two different approaches. The content based approach creates a profile for each user or product to characterize its nature. As an example, a movie profile could include features regarding its genre, the participating cast, its box office popularity, etc. User profiles might include demographic statistics or answers to a suitable questionnaire. The resulting profiles allow algorithms to associate users with matching products. However, content based approaches require gathering external information that might not be obtainable or easy to gather.

An alternative approach, our focus in this work, relies only on past user behavior without requiring the creation of categorical profiles. This approach is known as Collaborative Filtering (CF), a term coined by the developers of the first recommender system - Tapestry [4]. CF analyzes relationships between users and interdependencies among products, in order to discover new user-item associations. For example, some CF systems identify pairs of items that tend to be rated similarly or like-minded users with similar history of rating or purchasing to deduce unknown relationships between users and items. The only required information is the past behavior of users, which might be, e.g., their previous transactions or the way they rate products. A major appealing factor of CF is that it is domain free, yet it can address aspects of the data that are often elusive by bringing out the latent factors of the user-item rating matrix.

The CF problem can be thought of as a missing value estimation. We are given a user-item matrix of scores with many missing values, and our goal is to estimate the missing values based on the given ones. The known user-item scores measure the amount of interest between respective users and items. They can be explicitly given by users that rate their interest in certain items. We call these user-item scores ratings, and they constitute the input to our algorithm.

2 RELATED WORK

2.1 Memory-based Collaborative Filtering

The original neighborhood-based approach, which was shared by virtually all earlier CF systems, is user-oriented. In order to estimate the unknown rating, we resort to a set of users that tend to rate similarly to neighbors, and that actually rated item. The predicted value of is taken as a weighted average of the neighbor's rating.

The similarities computed play a central role here, as they are used both for choosing the neighbors and for weighting the above averages. Common choices are the Pearson correlation coefficient and the closely related cosine similarity. Sarwar et al[7]. found that item-oriented approaches deliver better quality estimates than user-oriented approaches while allowing more efficient computations. The greater efficiency follows because, typically, the number of items is significantly lower than the number of users, which allows pre-computing all item-item similarities for retrieval as needed. Neighborhood-based methods became widespread because they are intuitive and relatively simple to implement. In particular, they do not require tuning many parameters or an extensive training stage. They also provide a concise and intuitive justification for the computed predictions. This enables presenting the user a list of similar items that he or she has previously rated, as the basis for the estimated rating.

Standard neighborhood-based methods, however, raise some concerns:

1. The similarity function which defines the interpolation weights, is arbitrary. Various CF algorithms use different similarity measures,

*Produces the permission block, and copyright information
†The full version of the author's guide is available as `acmart.pdf` document

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

, ,
© 2016 Copyright held by the owner/author(s).
ACM ISBN 123-4567-24-567/08/06.
https://doi.org/10.475/123_4

trying to quantify the vague notion of user- or item-similarity. Suppose that a particular item is predicted accurately by a subset of the neighbors. In that case, we would want the predictive subset to receive all the weight, but that is impossible for constrained similarity scores like the Pearson correlation coefficient.

2. Earlier neighborhood-based methods did not account for interactions among neighbors. Each similarity between an item and a neighbor is calculated independently of the content of and the other similarities.

3. By definition, the interpolation weights sum to one, which may result in over-fitting. If an item has no useful neighbors rated by a particular user, in that case, it would be best to ignore the neighborhood information. Yet, the standard neighborhood formula uses a weighted average of ratings for the uninformative neighbors.

4. Neighborhood methods may not work well if variability differs considerably among neighbors.

2.2 Singular Value Decomposition (SVD)

Matrix factorization is factorizing a large matrix into two lower rank matrices called factors. Factors are multiplied to obtain the original matrix. Through matrix factorization, we can reduce the dimensionality of the original user-item rating matrix significantly[5].

This allows us to save memory of our computing system to a great extent.Also, the two new factored matrices bring out the latent features of both the users and items. The user-feature matrix represents the affinity of a user towards a feature, while the item-feature matrix represents the affinity of an item towards a feature. Feature here, could represent genre if the item we consider is a movie. Since we tend to deal with comprehensive data most of the time, SVD cannot be performed on sparse user-item rating matrices. To overcome this, our approach integrates the concept of SVD-like matrix factorization with model based machine learning algorithms to predict the missing values in a user-item rating matrix.

3 METHOD

Our approach is to integrate ideas of both memory-based CF and model-based CF to build a hybrid recommender that delivers better personalized results to the user. We perform item-based CF (memory-based CF) on the user's profile to identify similar items that would be recommended to the user. We then apply SVD-like matrix factorization on the original sparse user-movie rating matrix to get a dense matrix with all predicted user-movie ratings. Therefore, our hybrid model delivers better personalized results to the user by making use of the predicted dense matrix with all user-item ratings.

3.1 Item-Item based CF

Memory based CF approach tries to find similar users using a similarity score and take the weighted average of ratings[4]. The item-item based algorithm recommends items to a user that are similar to a given item of interest by determining users who liked those items in the past. To estimate the unknown rating r_{ui}, we identify a set of neighboring items N(i; u), which other users tend to rate similarly to their rating of i.

$$\frac{\sum_{j \epsilon N_{(i;u)}} s_{ij} r_{uj}}{s_{ij}} \tag{1}$$

The item-item similarities (denoted by s_{ij} are typically taken as either cosine similarities or correlation coefficients[1].

$$Similarity(S_{ij}) = \frac{i.j}{\mid i \mid\mid j \mid} \tag{2}$$

If the neighbor set contains three movies that are very similar in their nature, this approach while predicting the ratings might triple count the existing information and magnifies the similarity aspect significantly causing the rating to be biased[6]. However, this approach is effective to identify similar items for a given user's taste. To overcome the above stated limitation, we use a different approach to predict the ratings of the similar items that we determine using item-based CF.

3.2 Matrix Factorization

We now shift towards an approach where we calculate features that characterizes all users and movies. These features could represent genres of the movies. For example, one of such features could measure the action content in a movie, while another feature could measure the user's affinity towards comedy movies. We categorize each movie and each user within these newly defined genre-oriented scales. The latent features extract the information on how closely the user and the movies being recommended fit. Our goal is to identify latent features in a given user-movie matrix which help us predict the missing ratings. In our approach, we achieve this through SVD-like matrix factorization. In the context of information retrieval, this is widely known as Latent Semantic Indexing[1]. Given an m x n matrix R, SVD computes the best rank-f-approximation R^f, which is defined as the product of two rank matrices P_{mxf} and Q_{nxf}, where f< m,n[1]. That is, R^f =PQ^T minimizes the Frobenius norm ||R-R^f||$_F$ between all rank=f matrices. In this sense, the matrix Rf captures the f most prominent features of the data, leaving out less significant portion of the data that might be mere noise[1]. Unknown rating, r_{ui} is estimated as ${R^f}_{ui}$, which is a dot product of the u-th row of P with the i-th row of Q. Consequently, we refer to P as the user factors and to Q as the item factors[1].After decomposing the original user-movie rating matrix into user-factor matrix and item-factor matrix, we randomly initialize the elements of these matrices.

$$Err(P, Q) = \sum_{(u,i) \epsilon K} (r_{ui} - p_u^T q_i)^2 \tag{3}$$

We calculate the error function with respect to the known ratings in the original user-movie matrix. By minimizing the error function, we find the model that gives us the latent relationship between users and movies. The product of the resulting latent-factor matrices gives the estimated ratings for the missing values in the user-movie matrix.

4 EVALUATION

We evaluated our algorithm on the Movielens data. This dataset contains 100,000 ratings for 9,000 movies by 600 users. We measure the quality of our results using root mean squared error (RMSE), which puts more emphasis on large absolute error. Lower the RMSE,

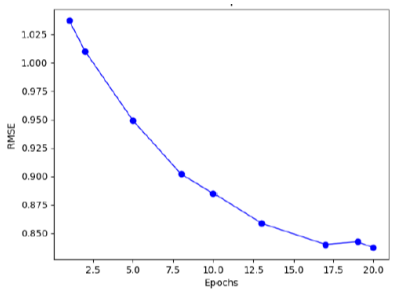

Figure 1: Graph obtained by plotting RMSE against Number of Epochs

USER_ID:1
MOVIE :ALIENS

	title	id	est
1011	The Terminator	218	3.045141
522	Terminator 2: Judgment Day	260	2.876597
6905	I Am Legend	6479	2.839699
7488	Avatar	19995	2.829389
987	Alien	348	2.822421
6640	Déjà Vu	7551	2.821203
922	The Abyss	2756	2.813491
7626	I Am Number Four	46529	2.619697
7946	Stake Land	52015	2.611557
6967	Doomsday	13460	2.610088

USER_ID:500
MOVIE :ALIENS

	title	id	est
6640	Déjà Vu	7551	3.103034
7939	Gantz	56832	3.017219
3619	Watchers	33172	3.006405
6967	Doomsday	13460	2.993972
7488	Avatar	19995	2.951482
1180	The Machine	197537	2.920187
6905	I Am Legend	6479	2.917017
987	Alien	348	2.912630
4347	Piranha Part Two: The Spawning	31646	2.897002
8042	The Darkest Hour	71469	2.884071

Figure 2: Unique recommendation results for two different users

better the recommendation accuracy.

$$RMSE = \sqrt{\frac{1}{n} \sum_{(u,i)} (p_{ui} - r_{ui})^2} \tag{4}$$

We trained the SVD model on 90% of the dataset where we performed five-fold cross-validation splitting. Learning rate of 0.005 yielded a low RMSE value of 0.8371 after 20 epochs.

The performance of our algorithm on test set is much better than the existing neighborhood-based techniques. The RMSE value on the test set is 0.9407.

5 DISCUSSION

In this paper, we designed a hybrid recommender system based on collaborative filtering techniques that personalizes the recommendations for any given user by taking into consideration, the user's preferences in the past. Item-Item similarity model filters out the top n items that a user would most likely be recommended. This model takes user's preferences into account to find similar items that he/she would like. We apply SVD based matrix factorization technique on those similar items to get the estimated ratings that the user might give. Based on the ratings obtained, we recommend the most highly rated movies to the user which are personalized to his/her taste.

From the above figure 2, we see that our hybrid model gives unique personalized recommendations to each user. As SVD model estimates the rating by identifying the latent factors of user and item, it helps the recommender to deliver user-specific recommendations.

6 CONCLUSION

In today's world we have an abundance of data , recommender systems help us in overcoming the problem by suggesting relevant items to the users. There is need for high quality recommender system . There are plenty of approaches to recommender systems. The are different approaches which are appropriate for different domains.We designed a hybrid recommender system which combines two methods.Our approach helps us in getting better performance when these methods are working in tandem. Our approach combines both item-item similarity and matrix factorization to recommend movies. Some of the common challenges that recommender systems today face are new users or items (cold start problem) , issues of privacy and security of user data, scaling the size of system with more users and items being added continuously, users not rating items, recommending to a Grey sheep(a user who is not similar to other users)[3].

7 REFERENCES

[1]Robert M. Bell, Yehuda Koren and Chris Volinsky "Modeling Relationships at Multiple Scales to Improve Accuracy of Large Recommender Systems"

[2] G. Adomavicious and A. Tuzhilin "Towards the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions" IEEE Transactions on Knowledge and Data Engineering 17 (2005)

[3]Kuinam J. Kim, Nikolai Joukov "Recommender Systems: Issues, Challenges, and Research Opportunities"

[4]D. Kim and B. Yum "Collaborative Filtering Based on Iterative Principal Component Analysis", Expert Systems with Applications 28 (2005).

[5]B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, "Application of Dimensionality Reduction in Recommender System - A Case Study", WEBKDD'2000

[6]Linden, G., Smith, B., York, J "Amazon. com recommendations: Item-to-item collabo- rative filtering." IEEE Internet Computing 7,(2003)

[7]B. Sarwar, G. Karypis, J. Konstan and J. Riedl, "Item-based Collaborative Filtering Recommendation Algorithms", Proc. 10th International Conference on the World Wide Web, pp. 285-295, 2001.