Advanced Science and Technology Letters
Vol.77 (UNESST 2014), pp.1-4
http://dx.doi.org/10.14257/astl.2014.77.01

Similar User Index-based MapReduce for Distributed Recommender Systems

Haesung Lee*, Joonhee Kwon**

*,**Department of Computer Science, Kyonggi University,
San 94-6, Yiui-dong, Yeongtong-ku, Suwon-si, Gyeonggi-do, Korea
{seastar0202, kwonjh}@kgu.ac.kr

Abstract. Due to the time complexity in composing recommendations, matrix factorization-based approaches are inefficient in dealing with large scale datasets. In this paper, we propose a similar user index-based parallel matrix factorization approach. Since the group of similar users is indexed in advance, there is no need to compute similarities between all users in datasets. Furthermore, the size of a matrix is reduced because the matrix is only composed of indexed user's ratings and items. The current advanced cloud computing including Hadoop, MapReduce and Amazon EC2 are employed to implement the proposed approaches.

Keywords: Scalable Recommender systems, Similar user index, ALS, Cloud computing

1 Introduction

To handle web-scale datasets with millions of users and billions of ratings, scalability becomes an important issues in the domain of recommender systems. However, most recommender systems based on CF suffer from two limitations, data sparsity and scalability. MF(Matrix Factorization) has become one of the leading techniques for the data sparsity which is main limitation of CF. Among many MF-based approaches, the alternating least squares (ASL) appears to be the most widely used method. ASL is inherently suitable for parallelization. But, due to the cubic time complexity in composing recommendations, ALS is not scalable to large-scale datasets.

In this paper, we propose a similar user index-based parallel matrix factorization algorithms. With the use of similar user index, it is possible to reduce computation time needed in composing recommendations. Also, the use of the similar user index reduces the size of a matrix based on which similarities of users are computed. That is, with the advantage of the similar user index, ALS-based recommender system is efficiently improved in the scalability. In addition, we implemented our approaches with the recent advanced cloud computing techniques including Hadoop, MapReduce

ISSN: 2287-1233 ASTL

and Mahout in order to solve the problem of high computation complexity of large scaled datasets.

2 Related Works

Collaborative filtering (CF) is the most successful recommendation technology and is used in many of the most successful recommender systems [1]. Basically, a CF problem is modeled as regularized low-rank matrix approximation. However, CF has two fatal limitations which are respectively related with the data sparsity and scalability [2]. Matrix factorization approaches have been applied successfully for both rating-based and implicit feedback-based CF problems [3]-[5]. There are two commonly used approaches to perform the approximate matrix factorization task, gradient descent and alternating least squares (ALS). ALS is effectively used for implicit feedback dataset. Besides, it is widely known that ALS is inherently suitable for parallelization [6].

The prediction formula of ALS for user u and item i is . The parameters of the model and are called user and item feature vectors when F denotes the number of features. Let P and Q denote the matrices that contain user and item feature vectors as rows. The objective function associated with the model is , thus the goal of ALS is to approximate the rating matrix as so that the weighted squared error of the approximation is low. A nice property of ALS is that it does not replace objective function by an approximation, as other approaches often do. It achieves speedup by applying mathematical simplification. However, the more the matrix size increases, the more the computation time of ALS is needed. That is, ALS is not efficient for dealing with the large-scale dataset.

3 ALS-based Recommender System with the Use of Similar User Index

With the use of the similar user index, unnecessary data is efficiently filtered out and the matrix size of recommender algorithms is reduced. Therefore the recommender system with similar user index can quickly compose recommendation.

To solve the problem of high computation complexity of predicting user preferences and composing recommendations, we propose the use of similar user index. Since cloud computing including Hadoop and MapReduce can largely simplify the complexities with large-scale system development, job creation and job scheduling, we implemented the proposed approach with the cloud computing technique.

In order to implement similar user index with Hadoop framework, we define the MapReduce algorithm. Table 1 shows the algorithm for the MapReduce of similar user indexing. Algorithm 1 takes the MapReduce processing for the generation of the similar user index.

Table 1. MapReduce algorithms for similar user indexing

Algorithm 3. MapReduce of similar user indexing

```
procedure: SimilarUserIndexer_Mapper_1
input: history records
output: (userTuple, <itemID,1>) pairs
begin
1. initiate temp_ketValue_vucket <userID, item_list>
2. for each userID i in history records
3.         item_list ← all items experienced by i
4.         Add (i, item_list) in temp_ketValue_vucket
5.    for each userID i in temp_ketValue_vucket
6.         for each userID j in temp_ketValue_vucket
7.              for each itemID k in item_list of userID i
8.                   for each itemID l in item_list of userID j
9.                        if k == j
10.                       then emit (userTuple(i,j), <k,1>)
end

procedure: SimilarUserIndexer_Reducer_1
input: (userTuple, <itemID,1>) pairs
output: (userTuple, <match count>) pairs
begin
1. initiate temp_ketValue_vucket1 (userTuple, <itemID,1>) pairs
2. initiate match_count
3. for each userTuple i in temp_ketValue_vucket1
4.     match_count = 0
5.     initiate temp_ketValue_vucket2
6.     for each userTuple j in temp_ketValue_vucket1
7.       if i == j
8.       then
9.           match_count ← match_count + 1
10.    emit (i, match_count)
end

procedure: SimilarUserIndexer_Mapper_2
input: (userTuple, match_count)
output: (userTuple, distance) pairs
begin
1. initiate temp_ketValue_vucket (userTuple, match_count)
2. for each userTuple i in temp_ketValue_vucket
3.      distance ←Distance(k in i, l in i)
4.      emit (i, distance)
end

procedure: SearchRecommendedItem_Reducer_2
input: (userID, recommended_itemID_list) pairs
output: (userID, recommended_itemIDList) pairs
begin
```

```
1. initiate itemIDList
2. for each itemID i in recommended_itemID_list
3.     Add i in itemIDList
4. emit (userID, itemIDList)
end
```

AS being described in Table 1, MapReduce of similar user indexing is consisted of two pairs of MapReduce functions. In other words, the output of the first MapReduce function is the input of the second MapReduce function.

4 Conclusion

In this paper, we propose a new approach that ALS-based systems use the similar user index in composing recommendations. With the use of the similar user index, unnecessary data is efficiently filtered out and the matrix size of recommender algorithms is reduced. Therefore the recommender system with similar user index can quickly compose recommendation. Through the experimental result, we verify that the recommender system with the use of similar user index is superior to the recommender system in which only ALS method is applied without the use of similar user index.

Acknowledgments. This work was supported by the Gyonggi Regional Research Center (GRRC) of Korea and Contents Convergence Software (CCS) research center of Korea.

References

1. H. Ma, I. King, and M. R. Lyu. Effective missing data prediction for collaborative filtering. In SIGIR ’07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, 39–46 (2007)
2. Sarwar, Badrul, et al. Item-based collaborative filtering recommendation algorithms, Proceedings of the 10th international conference on World Wide Web. ACM, 285-295 (2001)
3. Koren, Y., Bell, R., and Volinsky, Matrix factorization techniques for recommender systems. Computer, 42(8), 30-37 (2009)
4. Jamali, M., and Ester, M., A matrix factorization technique with trust propagation for recommendation in social networks. In Proceedings of the fourth ACM conference on Recommender systems, 135-142 (2010)
5. Takács, G., Pilászy, I., Németh, B., and Tikk, D, Investigation of various matrix factorization methods for large recommender systems. In Data Mining Workshops, 2008. ICDMW'08. IEEE International Conference, 553-562 (2008)
6. [6] Yu, H. F., Hsieh, C. J., Si, S., and Dhillon, I. S, Scalable Coordinate Descent Approaches to Parallel Matrix Factorization for Recommender Systems. In ICDM 765-774, (2012)

