A Collaborative Filtering Approach for Big Data Application Based on Clustering

Ravi L. Batheja[1], S. A. Vyawhare[2]

[1]*Student, Department of CSIT, Sanmati Engineering College, Washim, India*

[2]*Professor, Department of CSIT, Sanmati Engineering College, Washim, India*

***Abstract*: In the recent days the web domain is augmented with new types of services, with the increase in service and cloud computing. As a result, new forms of web content collecting /designing is done based on the numerous openly available web services online. These services are utilized in many ways by different domains and with the exponential growth of these web services users are experiencing difficulties in finding and utilizing a best matching service for their mash up. A collaborative filtering approach is going to filter and recognize the similar services under same cluster and followed by those evaluations recommendations are made Recommender systems are now popular both commercially and in the research community, where many approaches have been suggested for providing recommendations. In many cases a system designer that wishes to employ a recommendation system must choose between a set of candidate approaches. A first step towards selecting an appropriate algorithm is to decide which properties of the application to focus upon when making this choice. Indeed, recommendation systems have a variety of properties that may affect user experience, such as accuracy, robustness, scalability, and so forth. In this paper the system discusses how to compare recommenders based on a set of properties that are relevant for the application. Recommender systems can now be found in many modern applications that expose the user to huge collections of items. Such systems typically provide the user with a list of recommended items they might prefer, or predict how much they might prefer each item. These systems help users to decide on appropriate items, and ease the task of finding preferred items in the collection.**

***Keywords*: Big data application, cluster, collaborative filtering, mash up.**

1. Introduction

Initially, most recommenders have been evaluated and ranked on their prediction power their ability to accurately predict the user’s choices. However, it is now widely agreed that accurate predictions are crucial, but insufficient to deploy a good recommendation engine. In many applications people use a recommendation system for more than an exact anticipation of their tastes. Users may also be interested in discovering new items, in rapidly exploring diverse items, in preserving their privacy, in the fast responses of the system, and many more properties of the interaction with the recommendation engine. The system must hence identify the set of properties that may influence the success of a recommender system in the context of a specific application. Then, the system can evaluate how the system performs on these relevant properties.

Due to large amounts of data in the dataset, too much time is required for this calculation, and in these systems, scalability problem is observed. Therefore, in order to calculate the similarities between data easier and quicker and also to improve the scalability of the dataset, it is better to group data, and each data should be compared with data in the same group. Clustering technique, as a model based method, is a promising way to improve the scalability of collaborative filtering by reducing the quest for the neighborhoods between clusters instead of using whole data set. It recommends better and accurate recommendations to users. In this paper, by reviewing some recent approaches in which clustering has been used and applied to improve scalability, the effects of various kinds of clustering algorithms (partition, clustering such as hard and fuzzy, evolutionary based clustering such as genetic, mimetic, ant colony and also hybrid methods) on increasing the quality and accuracy of recommendations have been examined. Collaborative filtering, as one of the most successful techniques, is based on the assumption that people who has similar interests in terms of some items; they will have the same preferences in other items. So the goal of collaborative filtering is to find the users who have similar ideas and preferences or to find the nearest neighbor of them. This method is carried out in three steps: preprocessing, similarity computation and prediction / recommendation generation.

Collaborative filtering is grouped into two general classes, namely, neighborhood-based (memory based) and model-based methods. In Memory based CF systems, the whole user-item rating dataset is used to make predictions. This system can be performed in two ways known user-based and item-based recommendations. User-based collaborative filtering predicts an active user rating in an item, based on rating information from similar user profiles, while item-based method looks at rating given to similar items. A cluster contains some similar services just like a club contains some like-minded users. This is another reason besides abbreviation that the system calls this approach Club CF. Since the number of services in a cluster is much less than the total number of services, the computation time of the CF algorithm can be reduced significantly. Besides, since the ratings of similar services within a cluster are more relevant than that of dissimilar services, the recommendation accuracy based on users‟ ratings may be enhanced.

Automated collaborative filtering systems soon followed, automatically locating relevant opinions and aggregating them to provide recommendations. Collaborative filtering (CF) is a popular recommendation algorithm that bases its predictions and recommendations on the ratings or behavior of other users in the system. The fundamental assumption behind this method is that other users' opinions can be selected and aggregated in such a way as to provide a reasonable prediction of the active user's preference. The majority of collaborative filtering algorithms in service today, including all algorithms detailed in this section, operates by first generating predictions of the user's preference and then produces their recommendations by ranking candidate items by predicting preferences. Often this prediction is in the same scale as the ratings provided by users, but occasionally the prediction is on a different scale and is meaningful only for candidate ranking. Finding similar users in advance is therefore complicated: a user's neighborhood is determined not only by their ratings, but also by the ratings of other users, so their neighborhood can change as a result of new ratings supplied by any user in the system. For this reason, most user–user CF systems find neighborhoods at the time when predictions or recommendations are needed. In systems with a sufficiently high user to item ratio, however, one user adding or changing ratings is unlikely to significantly change the similarity between two items, particularly when the items have many ratings. Therefore, it is reasonable to pre-compute similarities between items in an item–item similarity matrix.

Clustering is a critical step in our approach. Clustering methods partition a set of objects into clusters such that objects in the same cluster are more similar to each other than objects in different clusters according to some defined criteria. Generally, cluster analysis algorithms have been utilized where the huge data are stored. Clustering algorithms can be either hierarchical or partitions. Clustering and classification are both fundamental tasks in Data Mining. Classification is used mostly as a supervised learning method, clustering for unsupervised learning (some clustering model is for both). The goal of clustering is descriptive, that of classification is predictive. Since clustering is the grouping of similar instances/objects, some sort of measure that can determine whether two objects are similar or dissimilar is required. There are two main types of measures used to estimate this relation: distance measures and similarity measures. Information is pair wise constraints, which include must link and cannot-link constraints specifying that two points must or must not belong to the same cluster. A number of previous studies have demonstrated that, in general, such constraints can lead to improved clustering performance. However, if the constraints are selected improperly, they may also degrade the clustering performance. Moreover, obtaining pair wise constraints typically requires a user to manually inspect the data points in question, which can be time consuming and costly. For example, for document clustering, obtaining a must-link or cannot-link constraint requires a user to potentially scan through the documents in question and determine their relationship, which is feasible but costly in time. For those reasons, we would like to optimize the selection of the constraints for clustering, which is the topic of active learning.

Clustering is performed by measuring exact distances only between points that occur in a common canopy. Using canopies, large clustering problems that were formerly impossible become practical. Under reasonable assumptions about the cheap distance metric, this reduction in computational cost comes without any loss in clustering accuracy. Canopies can be applied to many domains and used with a variety of clustering approaches, including Greedy Agglomerative Clustering, K-means and Expectation-Maximization.

We present experimental results on grouping bibliographic citations from the reference sections of research papers. Here the canopy approach reduces computation time over a traditional clustering approach by more than an order of magnitude and decreases errors in comparison to a previously used algorithm by 25%. Traditional clustering algorithms become computationally expensive when the data set to be clustered is large. There are three different ways in which the data set can be large: (1) there can be a large number of elements in the data set, (2) each element can have many features, and (3) there can be many clusters to discover. Recent advances in clustering algorithms have addressed these efficiency issues, but only partially.

Collaborative Filtering (CF) systems work by collecting user feedback in the form of ratings for items in a given domain and exploiting similarities in rating behavior amongst several users in determining how to recommend an item. CF methods can be further sub-divided into neighborhood-based and model-based approaches. Neighborhood-based methods are also commonly referred to as memory- based approaches. Model-based techniques provide recommendations by estimating parameters of statistical models for user ratings. For example, describe an earlier approach to map CF to a classification problem, and build a classifier for each active user representing items as feature vectors over users and available ratings as labels, possibly in conjunction with dimensionality reduction techniques to overcome data sparsest issues. Other predictive modeling techniques have also been applied in closely related ways.

Recommender Systems (RSs) are software tools and techniques, providing suggestions for items to be of use to a user. In this introductory chapter, we briefly discuss basic RS ideas and concepts. Our main goal is to delineate, in a coherent and structured way, the chapters included in this handbook and to help the reader navigate the extremely rich and detailed content that the handbook offers. RSs development initiated from a rather simple observation: individuals often rely on recommendations provided by others in making routine, daily decisions. Recommender systems play an important role in such highly rated Internet sites as Amazon.com, YouTube, Netflix, Yahoo, Trip advisor, Last FM, etc. Moreover, many media companies are now developing and deploying RSs as part of the service they provide to their subscribers. For example, Netflix, the online movie rental service, awarded a million-dollar prize to the team that first succeeded in improving substantially the performance of its recommender system. Now we want to refine this definition illustrating a range of possible roles that an RS can play. First of all, we must distinguish between the role played by the RS on behalf of the service provider from that of the user of the RS. For instance, a travel recommender system is typically introduced by a travel intermediary (e.g., Expedia.com) or a destination management organization (e.g., Visitfinland.com) to increase its turnover (Expedia), i.e., sells more hotel rooms, or to increase the number of tourists to the destination.

2. Proposed system

The system discusses the core algorithms for collaborative filtering and traditional means of measuring their performance against user rating data sets. The system will then move on to discuss building reliable, accurate data sets; understanding recommender systems in the broader context of user information needs and task support; and the interaction between users and recommender systems.

Collaborative filtering (CF) is a popular recommendation algorithm that bases its predictions and recommendations on the ratings or behavior of other users in the system. The fundamental assumption behind this method is that other users' opinions can be selected and aggregated in such a way as to provide a reasonable prediction of the active user's preference. The focus of this survey is on collaborative filtering methods, although content-based filtering will enter our discussion at times when it is relevant to overcoming a particular recommender system difficulty. The majority of collaborative filtering algorithms in service today, including all algorithms detailed in this section, operates by first generating predictions of the user's preference and then produces their recommendations by ranking candidate items by predicted preferences.

CF is a straightforward algorithmic interpretation of the core premise of collaborative filtering: find other users whose past rating behavior is similar to that of the current user and use their ratings on other items to predict what the current user will like. To predict Mary's preference for an item she has not rated, user–user CF looks for other users who have high agreement with Mary on the items they have both rated. These users' ratings for the item in question are then weighted by their level of agreement with Mary's ratings to predict Mary's preference. Pre-computation and truncation is essential to deploying collaborative filtering in practice, as it places an upper bound on the number of items which must be considered to produce a recommendation and eliminates the query-time cost of similarity computation. It comes with the small expense of reducing the number of items for which predictions can be generated.

Advantages:

Making an optimal decision for the recommendation within an acceptable time. Making recommendations from a wide array of services. Updated dynamically and thereby the predictions and recommendations are updated one.

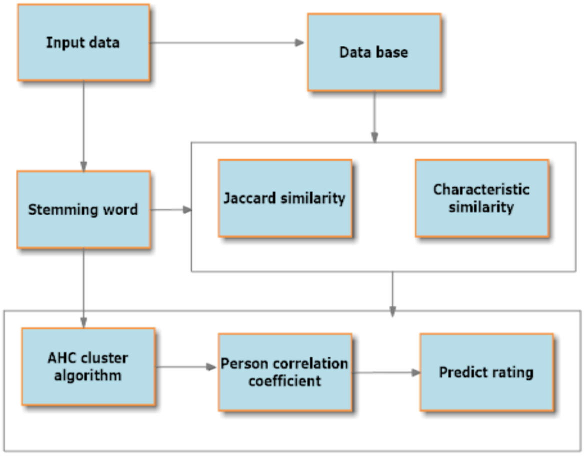

Fig. 1. System architecture

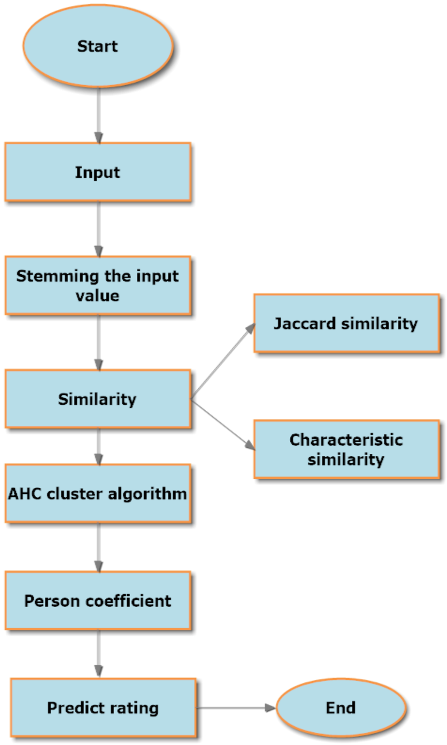

Fig. 2. Flow diagram

3. Modules

Clustering

- Stemmer Stemming
- Jaccard Similarity Coefficient

- Characteristic Similarity
- Agglomerative Hierarchical Clustering

Collaborative Filtering

- Pearson Correlation Coefficient
- Similarity Rating for neighbors evaluation
- Compute and Predict rating

A. Module description

Stemmer Stemming: Stemmer is used to remove the inflected part of the word to get their root form. It is used to reduce the word to its root form. Different variants of a term can be conflated to a single representative form. It saves storage space and time. A stemming is a technique used to reduce words to their root form, by removing derivational and inflectional affixes. The stemming is widely used in information retrieval tasks. Many researchers demonstrate that stemming improves the performance of information retrieval systems. Stemmer is the most common algorithm for English stemming.

Stemming is a technique to detect different inflections and derivations of morphological variants of words in order to reduce them to one particular root called stem. A word's stem is its most elementary form which may or may not have a semantic interpretation. In documents written in natural language, it is hard to retrieve relevant information. Since the Languages are characterized by various morphological variants of words, this leads to mismatch vocabulary. In applications using stemming, documents are represented by stems rather than by the original words. Thus, the index of a document containing the words "computing", "compute" and "computer" will map all these words to one common root which is "compute". This means that stemming algorithms can considerably reduce the document index size, especially for highly inflected languages, which leads to important efficiency in time processing and memory requirements.

Similarity Measures: Jaccard and characteristic similarity has been processed between the set of services. In-order to enhance the frequency rate mechanisms the system find the weights of attributes and ranking it there by improve the Search scenario. Web Services data has to be categorized according to the set of open service descriptions and their properties. String matching mechanisms usually consist of keyword based search mechanisms and their degree of matching. Clustering of web documents enables semi-automated categorization, and facilitates certain types of search. Any clustering method has to embed the documents in a suitable similarity space.

Rating Similarity and Predicted Rating: PCC is applied to compute rating similarity between each pair of services in ClubCF. Ranking algorithm compute similarity between document and query vectors to yield a retrieval score to each document. According to the relevance with the user query retrieved document are ranked. Based on the enhanced rating similarities between services, neighbors are predicted.

Performance Evaluation: Collaborative based Service clustering achieves less number of clusters compare to whole system of clusters. Proposed system achieves less executional time. Performance is measured in terms of (Parameters) computation time, no of clusters and memory usage.

4. Conclusion

This paper, we present a ClubCF approach for big data applications relevant to service recommendation. Before applying CF technique, services are merged into some clusters via an AHC algorithm. Then the rating similarities between services within the same cluster are computed. As the number of services in a cluster is much less than that of in the whole system, ClubCF costs less online computation time. Moreover, as the ratings of services in the same cluster are more relevant with each other than with the ones in other clusters, prediction based on the ratings of the services in the same cluster will be more accurate than based on the ratings of all similar or dissimilar services in all clusters. These two advantageous of ClubCF have been verified by experiments on real-world data set.

Many recommendation systems employ the collaborative filtering technology, which has been proved to be one of the most successful techniques in recommender systems in recent years. With the gradual increase of customers and products in electronic commerce systems, the time consuming nearest neighbor collaborative filtering search of the target customer in the total customer space resulted in the failure of ensuring the real time requirement of recommender system. At the same time, it suffers from its poor quality when the number of the records in the user database increases. Sparsity of source data set is the major reason causing the poor quality. To solve the problems of scalability and sparsity in the collaborative filtering, this paper proposed a personalized recommendation approach joins the user clustering technology and item clustering technology. The algorithm is tested on several well-known real-life data sets. The experimental results indicate that the proposed optimization algorithm is at least comparable to the other algorithms in terms of function evaluations and standard deviations.

References

[1] M. A. Beyer and D. Laney, "The importance of "big data": A definition," Gartner, Tech. Rep., 2012.
[2] X. Wu, X. Zhu, G. Q. Wu, et al., "Data mining with big data," IEEE Trans. on Knowledge and Data Engineering, vol. 26, no. 1, pp. 97-107, January 2014.
[3] A. Rajaraman and J. D. Ullman, "Mining of massive datasets," Cambridge University Press, 2012.
[4] Z. Zheng, J. Zhu, M. R. Lyu. "Service-generated Big Data and Big Data-as-a-Service: An Overview," in Proc. IEEE BigData, pp. 403-410, October 2013.
[5] A. Bellogín, I. Cantador, F. Díez, et al., "An empirical comparison of social, collaborative filtering, and hybrid recommenders," ACM Trans. on Intelligent Systems and Technology, vol. 4, no. 1, pp. 1-37, January 2013.
[6] W. Zeng, M. S. Shang, Q. M. Zhang, et al., "Can Dissimilar Users Contribute to Accuracy and Diversity of Personalized Recommendation?," International Journal of Modern Physics C, vol. 21, no. 10, pp. 1217-1227, June 2010.

[7] T. C. Havens, J. C. Bezdek, C. Leckie, L. O. Hall, and M. Palaniswami, “Fuzzy c-Means Algorithms for Very Large Data,” IEEE Trans. on Fuzzy Systems, vol. 20, no. 6, pp. 1130-1146, December 2012.

[8] Z. Liu, P. Li, Y. Zheng, et al., “Clustering to find exemplar terms for keyphrase extraction,” in Proc. 2009 Conf. on Empirical Methods in Natural Language Processing, pp. 257-266, May 2009.

[9] X. Liu, G. Huang, and H. Mei, “Discovering homogeneous web service community in the user-centric web environment,” IEEE Trans. on Services Computing, vol. 2, no. 2, pp. 167-181, April-June 2009.