International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 – 5161
©2015 INPRESSCO®, All Rights Reserved
Available at http://inpressco.com/category/ijcet

Research Article

Designing a Parallel Hybrid and Commercial Movie Recommendation System

Saurav Maiti†*, Aishwarya Chandrasekhar†, Saif Pathan†, Madhavi A Pradhan†, Shiva Karthik‡ and Swati Mehta‡

†Department of Computer Engineering, Savitribai Phule Pune University, Pune, India
‡Department of Applied Artificial Intelligence Group, Center for Development of Advanced Computing, Pune, India

Accepted 15 May 2015, Available online 18 May 2015, **Vol.5, No.3 (June 2015)**

Abstract

Recommendation systems have gained tremendous popularity over the past few years. As we all know recommendation systems have provided a boon in the field of online shopping and other browsing portals. Recommendation systems use machine learning techniques to predict what the user may like based on his history of interaction with a system full of items. At present there are many approaches known to implement a recommendation system. These approaches have several algorithms with various efficiencies. The efficiency and the accuracy of the recommendation system solely depend on the algorithm used. Hence, we've designed, implemented, and successfully deployed a system that uses an ensemble of item-, and content-based recommendation systems. In this paper we are going to explain how the results from two algorithms which are run on Hadoop are combined to get more accurate movie recommendations.

Keywords: *Recommendation system algorithms, collaborative filtering, content based filtering, Hadoop, Apache Solr*

1. Introduction

Recommendation systems basically filter unwanted data from a large pool of data and recommend the user only that data which is desired by the user. Recommendation systems are very good at suggesting people what they may like. Because of this, recommendation systems have gained lot of interest among people. Today, most large-scale social and e-commerce portals have installed a recommendation engine to gather interest of people in order to make greater profits. To implement this, there are many techniques/algorithms by the use of which one can design a recommendation system. But, we have implemented only two such algorithms. These algorithms are divided into two categories

1. Item-Based Collaborative Filtering
2. Content Based Filtering

Depending upon the use for which the recommendation system is being designed different algorithms is used. There are some important criteria that determines how useful the algorithm is

- Quality of Prediction
- Speed/scalability
- Easily Updated

A. Quality of Prediction

The quality of prediction determines the effectiveness of the recommendation system. We want our system to be a learned system. Quality output should be the main goal of the system. In our movie recommendation system, this criterion is fulfilled by using the combination of results from two algorithms which makes it hybrid system.

B. Speed/Scalability

Now days, most recommendation systems are mainly designed for commercial and online data processing which are bound to operate over large datasets. Instant and appropriate results are much appreciated and thus speed of algorithm over scalable large data is an important factor. This means that the algorithm cannot take too long to make any predictions. Scalability problem is minimized in our system by using Hadoop's mapreduce.

C. Easily Updated

The amount of data is constantly increasing and so the algorithm should be able to cope up with this ever increasing problem in an efficient way which should not affect the speed and quality of recommendation.

*Corresponding author: **Saurav Maiti**

Apart from this, there are few hurdles that are faced while designing an algorithm which are Cold Start Problem --which is faced by recommendation engine in suggesting items or services to a new user which has no record of ratings or items viewed, the other problem is Sparse Data Problem –sometimes the datasets available are very sparse and the system still needs to make a good prediction.

2. Overview of Algorithms

2.1 Collaborative filtering

This is the most widely used approach in recommendation systems. It is used to provide recommendations using known attributes of users with similar interests. It is mainly based on analysing user history, user behaviour, user likes and dislikes. This analysis is used to find the degree of similarity between users. Generally, two algorithms are used to identify similar users, namely K-Nearest Neighbour (k-NN) algorithm and the Pearson Correlation. We are reviewing K-nearest neighbour approach.

2.1.1 Item-Based K-Nearest Neighbour algorithm

This is similar to User Based K-Nearest Neighbour approach, except, the nearest neighbour similarity is based upon item to item similarity instead of user to user similarity. This approach is explained in the following steps. This algorithm is written in hadoop's mapreduce framework in our system for efficient processing.

A. The aim is to predict rating of an item *'i'* by a user *'u'*. This can be achieved by finding similarity weights between items using cosine similarity formula.

$$\text{Similarity (Item1, Item2)} = \cos\theta = \frac{\text{Item1. Item 2}}{||\text{Item1}|| \, ||\text{Item2}||}$$

B. The similarity vector is then formulated by arranging the weights in descending order and thus K Nearest Neighbour of item *'i'* can be easily found.
C. Now, another vector R_U that consists of all items already rated by user is formed.
D. Finally, the similarity vector of every item is combined with vector R_U to predict rating of items by user. These ratings are then used to recommend an item to user.

In real time the number of users is vast compared to number of item, thus Item-Based K-Nearest Neighbour algorithm proves to be efficient in practice. Also, the ratings data is vast; hence Hadoop is used for big data management.

The top-5n results from this algorithm are taken for final ensemble.

2.2 Content Based Filtering

This is another common approach which is based on features of movie Meta data and history of user which in turn builds up user and movie profile. These profiles play the key role in recommending movies to the user.
The movie which shares the maximum features in a user's history is more likely to be recommended. We have implemented this algorithm in the following three simple steps in hadoop's mapreduce framework to achieve parallelism.

a. Firstly, a set of all features or attributes of movies present in the dataset is created.
b. The system maintains for each attribute value the number of movies in the user's history with that attribute value plus the rating the user has given for such movies.
c. The final score of a movie for a particular user is then a non- negative combination of the different scores each of the movie's attribute values have for the particular user.

Content-based filtering is efficient at calculating the predicted ratings for movies. It considers the contents of movies instead of item similarity.
The top-5n results are taken from this algorithm for final ensemble.

3. Final Parallel Hybrid Recommendation System Ensemble

The final top-*n* recommendations for a particular user *u* are computed by first asking each of the two recommenders (in parallel) to compute the top-*5n* recommendations for *u* and then computing for each recommended movie (by any of the individual recommenders), a linear weighted combination of the recommendation values of the two recommenders. We optimized the weights *w(i), w(c)* of the item-based, and content-based recommenders. The resulting values are sorted in descending order, and the top-*n* movies are returned.

4. Implementation Details

The implementation of our project takes place in two stages:

1. Offline Part.
2. Online Part.

4.1 Offline Part

In the offline part, the training is done after a regular period of time. For example after a week or after a month. All the user interactions are recorded within this period. This recorded data is then processed in order to train the system. These processes are carried out in the server. In our project we have used already recorded data from MovieLens.org.

4.1.1 Hadoop Multi-node cluster

In real-time, the Movie data generated is in petabytes and hence it is not feasible to process data using a single machine. Hadoop helps in distributing the processing.

In our system, we have set up a Hadoop Multi-Node cluster which automatically divides the input data into multiple chunks and processes it in a distributed environment. Hence, it increases the efficiency drastically.

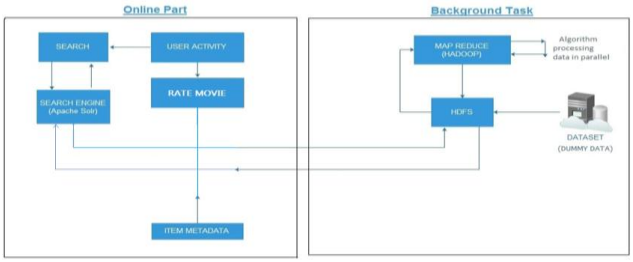

Figure 1 Block Diagram

4.2 Online Part

The online part deals with indexing the output file using Apache Solr Search engine tool. The indexed file can then be easily accessed from the client side. The client activities also come in the online part.

4.2.1 Search Engine

Solr enables powerful matching capabilities including phrases, wildcards, joins, grouping across any data type. The dataset is a Pipe Separated File and is posted to the Solr Server for providing quick search result. It is able to do so because it indexes each document in XML like tags and when a query is fired to the Solr Server, it parses through the indexed file instead of the actual document. This speeds up the search and the entire row is returned. Each field can be separately accessed as required. The working steps are as follows:

i) When a user searches for a Movie name, a query will be fired to the server that will return a set of relevant documents.
ii) User will select the required movie
iii) Another query will be fired to retrieve set of all movies that are recommended for the selected movie as "*People who liked this also liked:*"

5. Evaluating the Movie recommendation system.

In our system, we wish to predict the rating a user would give to a movie (e.g. 1-star through 5-stars). In such cases, we wish to measure the accuracy of the system's predicted ratings.

Root Mean Squared Error (RMSE) is perhaps the most popular metric used in evaluating accuracy of predicted ratings. The system generates predicted ratings r_{ui} for a test set T of user-item pairs (u, i) for which the true ratings r_{ui} are known. Typically, r_{ui} are known because they are hidden in an offline experiment, or because they were obtained through a user study or online experiment. The RMSE between the predicted and actual ratings is given by:

$$\mathrm{RMSE} = \sqrt{\frac{1}{|\mathcal{T}|}\sum_{(u,i)\in\mathcal{T}}(\hat{r}_{ui} - r_{ui})^2}$$

Mean Absolute Error (MAE) is a popular alternative, given by

$$\mathrm{MAE} = \sqrt{\frac{1}{|\mathcal{T}|}\sum_{(u,i)\in\mathcal{T}}|\hat{r}_{ui} - r_{ui}|}$$

The MAE for Item Based collaborative filtering alone, Content Based Filtering alone and combination of both were calculated and compared as follows.

Table 1 Comparison of MAE of various approaches

Approach	MAE	% Accuracy
Item Based Collaborative Filtering alone	0.754	75.4%
Content Based Filtering alone	0.7873	78.73%
Combination of ICF and CBF(Implemented in system)	0.842	84.2%

As observed from the above table that the accuracy of the recommendation system increases drastically with the combination of Item based collaborative filtering and Content based filtering. Hence, combination of the results predicts more accurate ratings.

Conclusions

We have developed a system for real time recommendation generation. The primary goal of the project is to create a system that will provide near relevant recommendations has been successfully achieved. Our system is aimed to ease the users woes of having to select from a huge variety of options available at hand. Also, accessing the system environment simpler so that the person who is unaware about the system can handle it with ease. Our system has also dealt with the issues related to design a real-time recommendation system like handling big data.

The ability for a recommendation system to bubble up activity in real time is a huge advantage because the system is always on.

Recommendation systems are popping up everywhere, from movies to news to travel and leisure. They provide valuable, personalized information that can greatly influence the way we use the Web. They might even be the beginnings of an artificial intelligence for each of us. Recommendation systems are out there: watching, and learning.

References

R. Shankar, Prof. Ateshkumar Singh, Ms. Ila Naresh Patil (2013), Overview Of Recommendation System & A Speedy Approach In Collaborative Filtering Recommendations Algorithms With Mapreduce, International Journal of Computational Engineering Research, Vol, 03, Issue, 9.

NilayNarlawar, IlaNaresh Patil (2014), A Speedy Approach:User-Based Collaborative Filtering With Mapreduce, International Journal of Computer Engineering &Technology(IJCET), Vol 5, Issue 5.

Zhi-Dan Zhao, Ming-Sheng Shang (2010), User-based Collaborative-Filtering Recommendation Algorithms on Hadoop, Third International Conference on Knowledge Discovery a Data Mining.

Mohamed Sarwat, Justin J. Levandoski, Ahmed Eldawy, and Mohamed F. Mokbel (2014), LARS*: An Efficient and ScalableLocation-Aware Recommendation System, IEEE Transactions On Knowledgeand Data Engineering, vol. 26, no. 6.

Emmanouil Amolochitis, Ioannis T. Christou, Zheng-Hua Tan (2014), Implementing a Commercial-Strength Parallel Hybrid Movie Recommendation Engine, IEEE.

Pedro G. Campos, Ignacio Fernández-Tobías, Iván Cantador, Fernando Díez (2011), Context-Aware Movie Recommendations: An Empirical Comparison of Pre-Filtering, Post-Filtering and Contextual Modeling Approaches, Escuela Politécnica Superior Universidad Autónoma de Madrid.

Shichao Zhang (2010), KNN-CF Approach: Incorporating Certainty Factor to kNN Classification, IEEE, Vol.11 No.1.

Mustansar Ali Ghazanfar and Adam Pr¨ugel-Bennett, An Improved Switching Hybrid Recommendation System Using Naive Bayes Classifier and Collaborative Filtering, School of Electronics and Computer Science, University of Southampton, Highfield Campus, SO17 1BJ, United Kingdom.

Incheon Paik and Hiroshi Mizugai (2010), Recommendation System Using Weighted TF-IDF and Naïve Bayes Classifiers on RSS Contents, Journal of Advanced Computational Intelligence and Intelligent Informatics, Vol.14 No.6.