Personalized Recommendation Strategy and Algorithm Optimization on Cloud Computing Platform

Xiang Li* and Li Wei

School of Software, East China University of Technology, Nanchang, 330013, China

Abstract

Information overload is a key issue of the current network information retrieval, and a personalized recommendation with special information filtering methods is an important way and means to solve this problem. Based on the analysis of the common methods used of personalized recommendation, the architectural design of the personalized recommendation is proposed on the cloud computing platform. Then, combined with the specific issues of employment recommendation, this article proposes an optimized algorithm of Mahout distributed personalized recommendation based on content and items. Compared with the current single target recommendation algorithm, this algorithm is more efficient with a good practical significance and reference value.

Keywords: personalized recommendation; cloud computing platform; distributed recommendation; information overload

© 2017 Totem Publisher, Inc. All rights reserved.

1. Introduction

With the rapid development of Internet technology, various types of information show explosive growth, and people gradually enter the information overloaded era from the age of information scarcity. In the face of vast amounts of information, it is impossible to easily access useful information. At present, search engines like Baidu and Google facilitate people’s need to find desired information. Although this can meet our need of some information to a certain extent, it is not powerful enough to enable a great number of Internet users to obtain the highly targeted information that they want. In order to solve the defects of search engine, the personalized recommendation system with special information filtering method emerged. In recent years, with the rise of e-commerce, the personalized recommendation system has gradually become an important research content for IT technology in e-commerce [1]. In 2001, IBM added some personalized features to its e-commerce platform WebSphere, providing convenience for customers to develop personalized e-commerce sites. Since ACM hosted the first International Conference on recommendation systems (ACM Conference on Recommender systems, referred to as RecSys) in the United States in 2007, it held five consecutive sessions [2-3]. In practical applications, some famous Recommended system at abroad include the Grouplens online news filtering system, the Jester joke recommendation system, Ringo music recommendation system, Amazon’s book recommender system, Netmx network video recommendation system and YouTube video recommendation systems, etc. The most famous one is the Amazon’s recommendation system, which adopts item-based collaborative filtering recommended techniques, uses the users’ item score to acquire an item neighbor set similar to target item, and chooses the top N items with the highest utility value to recommend to the user.

Compared with some foreign countries, the research of recommendation system started relatively later in China. However, with the widespread use of Internet and the rapid development of e-commerce, the recommendation system, especially personalized recommendation system, has attracted the attention of scholars and enterprise applications. In addition, more and more companies have applied recommendation system to their websites. For example, JingDong, DangDang, Alibaba and other shopping sites in e-commerce field have installed recommendation systems. Youku, Tudou and other video sites in multimedia field have applied recommended module; besides, Tencent and Sina also use corresponding recommendation system in their social applications.

* Corresponding author.
E-mail address: tom_lx@126.com
2. Personalized Recommendation Overview

A complete personalized recommendation system mainly consists of three parts: the module for collecting user information, user history and preferences; candidate resource object modules for recommending; and personalized recommendation strategy (or personalized recommendation algorithm). Among them, personalized recommendation strategy is the core [2, 4-7].

The model flow of the generic personalized recommendation system is shown in Figure 1. Feature information matching and information filtering are implemented upon both the user model constructed by user preferences, user history behaviors and other user information as well as the candidate recommended object model, according to a personalized recommendation strategy. At last, the resource objects that users are most likely to be interested in will be recommended to them.

Personalized recommendation system is a special information filtering system different from a search engine. It can find out the resources that can meet the current or potential needs of users to a degree from the massive information database, realizing the efficiently filtering of information and improving the utilization rate of resources [5-8].

![Figure 1. Generic model of personalized recommendation system](image)

2.1. Collaborative Filtering Recommendation

The basic idea of collaborative filtering recommendation is that if some users had the same preferences in the past, they will have similar preferences in the future. Therefore, how a neighbor with high similarity to current user evaluates the product can be used to predict to what degree the current user prefers to a specific product [9].

For example, the purchase records of user A and user B are very similar. Recently, A has bought a book that user B does not know, A can recommend this book to B. Since the selection of a book that users may be interested in means to filter out one of the most promising book from a large amount of information, the users would implicitly collaborate with others in this course. This technique is called collaborative filtering (CF, Collaborative Filtering).

Collaborative filtering recommendation strategy is the most widely used recommended strategy at present because there are no special requirements for the recommended target; for example, whether the structured text messages or the video infeasible to be text structured can be used as a recommended target.

The input data is a given User-Item score matrix, and the output data is generally a list of Top-N recommended items or a predictive value of the degree of the current user’s preference to specific item.

There are two types of collaborative filtering recommendation strategy: user-based collaborative filtering recommendation, and items-based collaborative filtering recommendation [10].

The main idea of the user-based collaborative filtering recommendation algorithm is as follows: the input data is a given score matrix and the current user ID; find other users with similar preference to the current user, which are referred to as the nearest neighbor. Then, for an item p that the current user has never seen, we can utilize the item score of his/her nearest neighbor to predict the score of item p that current user may make.
This strategy has two major defects: a large number of users and items makes it hard to compute the nearest neighbor users in real time, and as fewer items have been scored by users, the sparse data cannot support to find the nearest neighbor users. In order to overcome these defects, the items-based collaborative filtering recommendation method is developed, which realizes the offline pretreatment before recommendation. Even if the score matrix is huge, calculation and recommendation can be carried out in a real time.

The main idea of the items-based collaborative filtering recommendation is to use the similarity between items to calculate a predictive value. If the scores of the vast majority of users are similar for the same item, the current user may have a similar score for this item.

2.2. Content-based Recommendation

Content-based recommendation is based on the development of content filtering technology, which does not require the user’s evaluation information of items. However, it does extract features of items from the user’s selection in the past to establish the user’s preference model. Then, according to the candidate items matching the user preference model, the highest matching degree of N objects will be recommended to the user [11-12].

The basic process of content-based recommended strategy is shown in Figure 2. The system requires the information in two aspects, i.e., the record of user’s historical interest preference and the feature description of candidate items. In reality, the feature description of items may be varied, such as a book, a movie director, or the style of music. Moreover, the items feature that users concerned may vary from person to person. Thus, how to obtain a subjective qualitative feature that matches with the items features is a great challenge.

![Figure 2. The schematic diagram of content-based recommendations](image)

For such unstructured files as music and video, the existing technology cannot accurately extract the unique constraints of identification. However, for text files, the constraints can be easily recognized.

2.3. Knowledge-based Recommendation

Collaborative filtering and content-based recommendation strategy are widely used. In these cases, they appeared to be inadequate. The knowledge-based recommendation can effectively solve the problem [13].

In comparison, knowledge-based recommendation is more interactive, providing a personalized method to guide users to find their interested items in a large number of candidate items. It does not need the score data, so the recommended results do not depend on user ratings. There are two basic types of knowledge-based recommendations: constraint-based and case-based recommendations. The two recommended processes are very similar: the user specifies the specific requirements, and the system will give the recommended results. If the system fails to give a result, the users need to modify their requirements.

The difference between these two processes is how to use the knowledge; the constraint-based recommendation depends on clear recommendation rules so that we can search for candidate items set from all items under the recommendation rules. For example, the Hungarian Fundamental credit unions have developed a VITA financial services platform, which is a typical constraint-based recommendation system. The Case-based recommendation is to find the
similar candidate items set according to a variety of similarity measure method. For example, a well-known case is the Entrée-type system, which successfully recommended Chicago restaurants to those who participated in the Democratic National Convention in Chicago in 1996.

3. Design of Personalized Recommendation System based on Cloud Computing Platform

3.1. System Architecture

1) The data layer: The system general frame diagram is shown in Figure 3. Data is the foundation of a personalized employment recommendation system. If there is no data to support, all the functions of the entire system are useless. The final recommended result depends on the quantity and quality of the data to a certain extent. The recommended strategy belongs to the data intensive in its nature. A large high-quality data is a good thing for recommendation system.

Data stored on data layer can be divided into four categories: all employment information crawled from the Internet, user information (mainly personal job information), recommended results, and user feedback information. Distributed database HBase is used to store data.

All employment information is obtained through the distributed crawler Nutch, which can crawl employment information on third party employment services website, parse the web pages and store them in the HBase. The user’s resume includes user ID, name, gender, contact information, professional name, graduate school, job target, professional skills, expertise, personal introduction, internship or practical experience, and so on. For each different user, the recommended results calculated by personalized recommendation strategy are stored in HBase database, which can be quickly displayed after the user logs in. User’s feedback information is a reflection to recommendation system or recommended results, such as the degree of user’s interest in each recommended information, and user’s opinion or suggestion on the recommended information.

The abovementioned feedback-information can objectively evaluate the results of the personalized recommendation, whether good or bad, in order to better improve the existing recommendation system.

2) Logic layer: This is the engine of personalized recommendation engine and the core of the whole personalized recommendation system. If the system has a lot of data but does not use them to generate value, these data are only “waste”. Through logic layer, a large amount of data can be analyzed, calculated or inferred according to the features of each user so that hidden “treasure” can be discovered, i.e. personalized recommended results are produced. The distributed computation relying on Mahout and MapReduce can enable us to achieve double effect.

3) Presentation layer: This is the user interface of personalized recommendation system, which is the part closest to the user and also an important component of a complete system. The functions of presentation layer include user registration and login, user’s personal information interface, recommended information list, part of the information feedback, and a list of information that has been scored. Recommended information list, the most important part of client end in personalized recommendation system, is to present the results of personalized recommendation system to the user. Users can not only score on interested degree to each recommendation information, but also write down their own opinion or advice. All the information is the most objective evaluation to personalized recommendation system as well as the power for system improvement. User information and feedback information are two of the underlying data sources for personalized
recommendation system, and also a very important part. The recommended information with expressed interest scores can be browsed and the score can be modified at your wish.

3.2. Working Flow of the System

The working flow of the whole system clearly indicates the data flow and the collaborative relationship of each component of systems. The system is shown in Figure 4; each step of the process is described as follows:

1) The distributed crawler system Nutch is used to crawl job information from a third-party employment information service website and save the information in the HBase database. Then, the system’s registered users submit personal job-hunting information to the HBase database. At the same time, user’s interest scores to recommended information and feedback information are collected.

2) With the basic data, personalize recommended work can begin. Extract relevant data from the HBase database for off-line analysis and calculation, store the analysis of user feature vectors, and store the filtered offline recommended results into the HBase database for future calls.

3) Offline processing is completed.

4) If the user online modifies the key job-hunting information, the system will start online recommendation after the information is saved. The combination of online analysis result and offline recommendation results are presented to the user as a final recommendation result. If the user does not modify the job-hunting information, the system will present the off-line recommended results to the user.

5) After viewing recommended information lists, the user may express interest scores and comments to each recommended information; the feedback information will be saved into the HBase database.

6) The whole system flow ends.

3.3. Detailed Design of Personalized Recommendation Engine

In order to solve the cold start and data sparseness problem of recommendation system, the personalized recommendation strategy of the system uses a parallel hybrid mode, which combines items–based collaborative filtering recommendation
Personalized Recommendation Strategy and Algorithm Optimization on Cloud Computing Platform

4. Design of Recommendation in Mahout

4.1. Content-based Distributed Recommendation

In Mahout, it did not provide the implementation of content-based recommendation. However, the related API and some extension modules [14-15], such as the provided item similarity classes that calculate the similarity between items based on Pearson correlation coefficient, compare the sequence of preference values given by multiple users to a specific item, which
still relies on historical rating data. Therefore, it cannot be applied in this system. The system shall consider the employment information content, rather than relying on historical rating data, to make a content-based recommendation.

4.1.1. Content Representation

The employment information stored in employment information database exists in the form of text. The textual content is usually represented by a vector space model converted by TF-IDF method. Through the weight of the word, important words in the document can be found; but, the relationship between words is not recognized. The theme document model method LDA constructs the feature vector of the document through the theme content, which can express the subtle relationship between certain words.

In Mahout, LDA is implemented in the form of MapReduce operation so can run on the Hadoop cluster.

First, cull high-frequency vocabulary in textual contents of job-hunting information from employment information database, transfer them into TF vector, and take them as the input data of LDA. The output is the theme sets.

Then, map each employment information to a topic vector \( \{w_1, w_2, \ldots, w_j, \ldots, w_n\} \), and store these theme vectors in the employment information database. Similarly, user’s job-hunting information can also be mapped to a theme vector \( \{w_1, w_2, \ldots, w_i, \ldots, w_n\} \), which will be stored in the database as user preference vectors.

When calling the LDA method in Mahout, the following parameters need to be specified:

- TF vector input directory
- Output directory after each round of iteration
- Number of themes
- Number of features in the corpus
- Theme smoothing parameter
- The maximum number of iterations

Theme smoothing parameter is set to decrease excessive noise in text contents and reduce the probability of errors. Usually, smoothed value can be appropriately increased to add the weight of those low-frequency keywords. Of course, this requires more iteration to generate meaningful theme model.

The calculated dimensions of theme feature vectors are determined by the set number of theme. If the dimension of employment information feature vector obtained by inert construction idea and the TF-IDF based VSM method is very high, the next step is not easy to calculate.

4.1.2. Content similarity calculation

After obtaining the theme vector of employment information and user’s job-hunting information, the next step is to realize a measure of similarity between employment information content and user’s job-hunting information. The cosine similarity calculation method is used. The similarity between user \( i \) and employment information \( j \) is calculated as Equation (1).

\[
\text{sim}(i, j) = \frac{\sum w_i w_j}{\sqrt{\sum w_i^2} \sqrt{\sum w_j^2}}
\]

4.2. Items-based Collaborative Filtering Distributed Recommendation

Items-based collaborative filtering distributed recommendation has been implemented in Mahout, and can be divided into three steps: (1) construct user preference vector, (2) construct the Co-occurrence matrix of items, and (3) generate the recommendation.

The first step: construct user preference vector. While carrying on off-line analysis, extract the user expressed interest
scores to recommended information. Let the data in the database in the form (UserID, URL, Rate) be a record, in which the UserID is the user’s ID, the URL is the employment information web site, and Rate is the user’s interest score to this job information. If the user does not express interest scores, the value of Rate is 0; the preference vector of user \( i \) can be represented as vector \( [r_{i1}, r_{i2}, \ldots, r_{in}]^T \).

The second step: construct a Co-occurrence matrix of items. Introduce the co-occurrence matrix (co-occurrence matrix), and use the times that each item pair commonly appears in the score list of certain users to fill the matrix. If there are six users giving scores to item X and item Y, the item X and Y will appear six times at the same time. For any items that are not found in any two users’ score list, the co-occurrence number will be 0. The greater the co-occurrence number is, the higher the degree of similarity between the items will be.

In the distributed calculation, the co-occurrence matrix is used to represent the similarity between two items. Equation (2) expresses the co-occurrence number between item \( i \) and item \( j \).

\[
I_{ij} = \sum N(i) \cap N(j)
\]

where \( N(i) \cap N(j) \) represents the number of users who have given scores at the same time on item \( i \) and item \( j \).

Equation (3) is an expression of co-occurrence matrix that is diagonal symmetry. The values of \( t_{in} \) and \( t_{ni} \) are the same, and therefore the meaning of the expression. But, the value of the diagonal has no practical significance.

\[
\begin{pmatrix}
  t_{11} & \cdots & t_{1n} \\
  \vdots & \ddots & \vdots \\
  t_{n1} & \cdots & t_{nn}
\end{pmatrix}
\]

The third step: Calculate the recommendation results. Multiple the co-occurrence matrix with the user preference vector; a new vector will generate that can represent the recommended results \( [R_{i1}, R_{i2}, \ldots, R_{in}]^T \), as shown in Equation (4).

\[
\begin{pmatrix}
  t_{11} & \cdots & t_{1n} \\
  \vdots & \ddots & \vdots \\
  t_{n1} & \cdots & t_{nn}
\end{pmatrix}
\begin{pmatrix}
  r_1 \\
  \vdots \\
  r_n
\end{pmatrix}
= 
\begin{pmatrix}
  R_{11} \\
  \vdots \\
  R_{in}
\end{pmatrix}
\]

Exclude the items that users have given scores and sort the results. The greater the value of \( R_{ij} \), the more the user prefers item \( j \). Then, the first \( N \) items can be taken as the recommendation result.

However, the value of \( R \) does not represent the preference value. While large-scale data is calculated, the obtained result value will be large, and there will be thousands of recommendation results. This is because the recommendation system is concerned about the recommended order, that is, the relationship of the resulting value’s size rather than the specific numerical size. Thus, these will not affect the recommended results.

In the above calculation process, each user corresponds a vector; the MapReduce distributed computing is suitable to be adopted for matrix multiplication. For all users, the obtained recommended result is a matrix, in which the column value indicates user ID, the row value indicates the URL of job information, and the element values corresponded to the ranks indicates the user’s interest in URL.

Mahout uses the MapReduce approach to realize the item-based collaborative filtering recommendation. However, it uses co-occurrence matrix to express the items’ similarity matrix. The more the co-occurrence times are, the more similar the items are. If only considering the number of times that items have been selected at the same time, and ignoring the user’s rating data, the value of items similarity will be inaccurate.

Therefore, this article replaces the element value in the co-occurrence matrix with the item similarity value. The score of interest degree expressed by user on recommended information list can be regarded as the vectors in n-dimensional
space; the similarity between the employment information can be calculated by cosine method.

If user’s interest scores of the employment information \(i\) and employment information \(j\) are separately represented by vectors \(\vec{i}\) and \(\vec{j}\) in \(n\)-dimensional space, employment information similarity can be calculated as Equation (5).

\[
t_{ij} = \cos (\vec{i} \cdot \vec{j}) = \frac{\vec{i} \cdot \vec{j}}{||\vec{i}|| \cdot ||\vec{j}||}
\]  

(5)

In above formula, vector \(\vec{i}\) can be expressed as \([r_{i1}, r_{i2}, \ldots, r_{in}]^T\); \(r_{ni}\), indicating the interest scores expressed by user \(n\) on employment information; by the same way, vector \(\vec{j}\) can be expressed as \([r_{j1}, r_{j2}, \ldots, r_{jn}]^T\) so Equation (6) can be rewritten as follows:

\[
t_{ij} = \frac{\sum_{n} r_{ni} \cdot r_{nj}}{\sqrt{\sum_{n} r_{ni}^2} \cdot \sqrt{\sum_{n} r_{nj}^2}}
\]  

(6)

Use this formula to calculate the similarity value of two employment information, which can effectively use the interest score value expressed by user on employment information, make full use of the data, avoid the waste of data, and improve the quality of recommendation.

Normalize the value of each element in the user’s feature vector \([r_{i1}, r_{i2}, \ldots, r_{in}]^T\), and make its range of value between 0 and 1 in order to mix the recommended result.

4.3. Mixed Recommended Scheme

The personalized offline recommended part of the system adopts a parallel hybrid recommended scheme, as shown in Figure 6. The content-based recommendation in Mahout and the item-based collaborative filtering recommendation have been described above. This section will describe a mixed scheme combining two recommended strategies.

Before a weighted mixture is carried out, it is necessary to ensure that two recommended results are assigned to a comparable range; otherwise, the weighted mixture cannot start. The scope of the assignment for the content-based recommended results is \([0.1, 1]\), and the scope of the assignment for the item-based collaborative filtering recommendation is \([-1, 1]\). The values are within comparable range.

Equation (7) calculates the recommended value of each result obtained through multiple recommended strategy calculation in a weighted mixture. In the formula, \(rec\) and \(\beta_k\) respectively represent the auxiliary and corresponding weights of recommended results obtained through each recommended strategy, and \(\sum_{k=1}^{n} \beta_k = 1\). In this way, the predictive power of a variety of recommended strategies can be combined in a weighted way.

\[
rec_{\text{weight}}(u, i) = \sum_{k=1}^{n} \beta_k \times rec_k(u, i)
\]  

(7)
In this system, the value of $n$ is 2, and the values of $\beta_1$ and $\beta_2$ are 0.5, which is a balanced weighted scheme. After a weighted calculation of the recommended results for the two recommended strategies, each recommended result is given a new recommended value. After sorted, store the result with recommended value greater than 0.1 to HBase database.

5. Experimental Result

The average absolute error (Mean Absolute Error, referred to as MAE) is used to evaluate the quality of the personalized recommendation results of the system. Assume a user $u$ and recommended employment information $i$, $r_u$ is the interest score expressed by user $u$ on employment information, and $R_u$ is the recommended values obtained by system according to the user $u$ and his/her employment information $i$. The calculated formula of MAE is shown as Equation (8), in which $N$ is the total number of interest scores expressed by user. The smaller the calculation results of MAE are, the more the recommended results are in line with the user’s mind.

$$MAE = \frac{\sum |r_u - R_u|}{N}$$ (8)

In the environment of the system configuration, an experiment was carried out for the personalized recommendation strategy of this system. The experiment results are as shown in Figure 7.

![Figure 7. MAE comparison with mixed recommendation](image)

In the above experiment, the employment information is about 2500 in the employment database. Due to less cluster nodes in the experimental environment, it is not suitable to use too many nodes in the experiment.

Although the number of users and interest scores data are less, MAE value is high, which will not affect the problems that experimental results manifest. More users and more mixed interest degree rating data will generate recommendation results with higher quality.

Due to the limitation of hardware and software, the system can only use a few nodes to test. Employment information database includes about 2500 employment information. Online recommendation response time and the number of nodes relationship are as shown in Figure 8. The number of nodes in cluster is four, and online recommendation response time and employment information quantity relationship are as shown in Figure 9.

Although the response time of the system is not very fast, it does not affect what the test results show. This is because the response time of online recommendation is irrelevant to the number of user, but closely related with the number of the cluster nodes and employment information. With the increase of cluster nodes number, the response time is shortened; the greater the number of employment information, the longer the response time will be.

The traditional stand-alone mode that depends on improving the hardware configuration or algorithm efficiency cannot meet the people’s need in pursuit of personalized information in large-scale data. The distributed cluster system can run in the low-cost PC, with efficient computation, vast storage space, safe data access, etc. However, the proposed method will inevitably replace the traditional stand-alone mode to provide perfect personalized information service to people in this big data era.
6. Conclusion

In the era of information overload, the cloud computing, Hadoop and other technologies are combined with personalized recommendation to form the best partner. Hadoop technology is utilized to supply efficient computing power and data-processing capacity for solving the large-scale data in personalized recommendation system, which is of great significance to the solution of practical problems.

Based on the study of personalized recommendation theory, this paper considers the features of MapReduce and Mahout, and integrates the practical application environment to improve the content-based recommendation. This enables the content-based recommendation to overcome the cold start of recommended system without depending on the user’s history records. Then, the improved content-based recommendation and item-based collaborative filtering recommendation are parallel mixed in order to overcome the sparse problems of recommendation system. Through the verification of experiments and system test, the system can be applied in the field of college students’ employment recommendation and is feasible to provide targeted personalized recommendation of employment information for college students, which has practical application value.

Acknowledgement

This work was supported by Key Technology R&D Program of Jiangxi Province (Nos. 20151BBE50072) and the Open Project Program of Jiangxi Engineering Research Center of Process and Equipment for New Energy, East China Institute of Technology (Nos. JXNE2016-10).
References


Xiang Li received his Ph.D. degree from Nanjing University of Science & Technology. He is an associate professor of East China University of Technology. His research interests include computer application.

Li Wei is a master student of East China University of Technology. Her research interests include big data and machine learning.