A Micro-video Recommendation System Based on Big Data

[1]Songtao Shang, [2]Minyong Shi, [3]Wenqian Shang, [4]Zhiguo Hong
[1, 2, 3] School of computer science, Communication University of China
[4]School of Computer, Faculty of Science and Engineering, Communication University of China
Beijing, China
songtao.shang@foxmail.com

***Abstract*—With the development of the Internet and social networking service, the micro-video is becoming more popular, especially for youngers. However, for many users, they spend a lot of time to get their favorite micro-videos from amounts videos on the Internet; for the micro-video producers, they do not know what kinds of viewers like their products. Therefore, this paper proposes a micro-video recommendation system. The recommendation algorithms are the core of this system. Traditional recommendation algorithms include content-based recommendation, collaboration recommendation algorithms, and so on. At the Bid Data times, the challenges what we meet are data scale, performance of computing, and other aspects. Thus, this paper improves the traditional recommendation algorithms, using the popular parallel computing framework to process the Big Data. Slope one recommendation algorithm is a parallel computing algorithm based on MapReduce and Hadoop framework which is a high performance parallel computing platform. The other aspect of this system is data visualization. Only an intuitive, accurate visualization interface, the viewers and producers can find what they need through the micro-video recommendation system.**

Keywords—micro-video; recommendation system; Slope one; data visualization

I. INTRODUCTION

Micro-video is a new form of information media. With the development of the Internet, 3G (the 3rd Generation mobile communication technology), and 4G (the 4th Generation mobile communication technology) network, the bandwidth and speed of network become faster and faster. These technologies provide conditions for dissemination of information media. Micro-video is a short time video [1] [2], which lasts for 30 seconds to 300 seconds. The short time micro-videos are popular with young people, because the teenagers are prefer to watch the micro-vide on their confetti time through mobile devices. For micro-video producers, the problem is they do not know how many people like their products, and do not know how many times their video have been watched. Therefore, this paper proposes a micro-video recommendation system (MRS). One of the purposes of MRS is an overview of micro-videos for the producer. In this way, the producer knows how many users love their video, and how many times their videos are on-demand. Another purpose of MRS is for users. The system can analyze the users' favorites and watching history, automatically push appropriate video to the users.

Big Data is becoming more popular with the internet technology development, which means the data sets whose size is beyond the ability of current technology, method and theory to capture, manage, and process the data within a tolerable elapsed time [3]. In order to enhance the MRS accuracy, we need to collect large volume data sets about who and when watched the micro-video, how many times the micro-video on-demanded, and how many people love the micro-video. Therefore, the MRS, proposing in this paper, use the Big Data technology to process the collected data sets.

Data sets are the foundation of the recommendation system. The first step of MRS is to collect data as far as possible from the Internet. We download data from video websites, video forum, video online chat websites, and so on. Web crawlers [4] [5], one of basic data sets collection, can download resources from Internet. The web crawlers originally used for search engine. In this paper, the results of crawler directly affect the accuracy of recommendation system.

After the data collection, we get enough information of micro-video, which contains the micro-videos' information and the users' watching information. Therefore, we can construct two models, i.e. micro-video model and user model. According to the user's exploring trace, the MRS can automatically compare the user's watching history with the micro-video model and the user's model, and push appropriate video that the user is interested in [6] [7]. There are many recommendation algorithms, such as collaborative filter algorithm [8], social recommendation algorithm [9], content-based filtering algorithm [10], and so on. In this paper we use Hadoop framework as the offline Big Data analyzing platform. Apache Hadoop [11] is an open-source software framework for storing and processing large data. Hadoop Mahout [12] is an open-source subproject based on Hadoop platform. At the same, Mahout is a scalable machine learning library for solving clustering, categorization, classification problems. It has been proved to be an inexpensive solution for machine learning problems. In this paper, we use Hadoop platform storing the web crawlers' downloaded data, and use Mahout to process the data.

978-1-5090-0806-3/16/$31.00 copyright 2016 IEEE
ICIS 2016, June 26-29, 2016, Okayama, Japan

The other important aspect of MRS is the processed data presentation. D3.js is a visualization JavaScript library based on data [13]. It provide customized mapping rule for users. Therefore, users can determine the mapping values to the graphic, such as display color, size, and so on. D3.js is suitable for processing with SVG (scalable vector graphics), which is the World Wide Web (WWW) specified network vector graphics standard. In this paper we user d3.js to present the recommendation results. Fig. 1 is an example of data presentation.

Fig. 1. Example of data presentation by using d3.js

II. MICR-VIDEO RECOMMENDATION SYSTEM ARCHITECTURE

The MRS architecture contains 5 service layers, as shown in Fig. 2.The function of each layer described as follows.

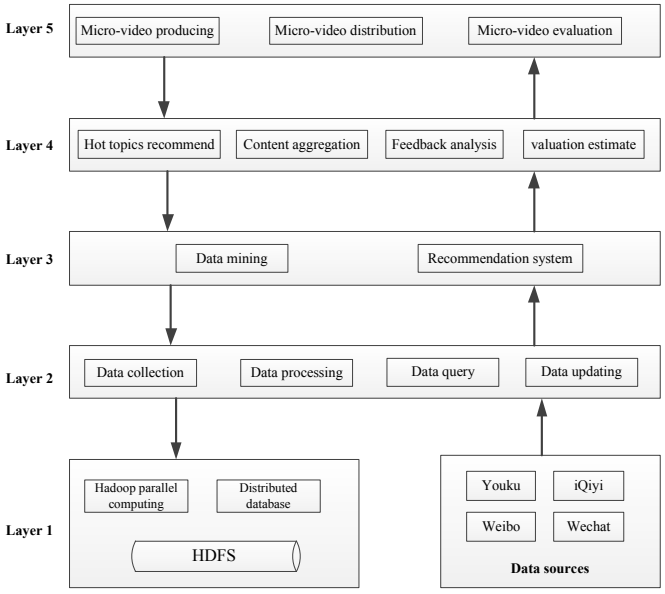

Fig. 2. Micro-video recommendation system

- Layer 1

Data source is the basic of MRS, since all the data what we need are download from the web sites. The video websites are the main data source from which we can download, such as youku, iQiyi, and so on. The contents that we need download include the video ID, brief description, click rate, ranking list and so on. We also download video comments from social networking service web sites, for example, Weibo, Wechat, and micro-video forum.

Another function of Layer 1 is Big Data processing. We use Hadoop framework as Big Data storage and processing platform. All the data what we download from the web sites are stored in the HDFS (Hadoop distributed file system) [14].

- Layer 2

Layer 2 is data interface layer, which contains data collection, data processing, data query, and data updating.

The main idea of data collection is to download micro-video information from the data source automatically, i.e. web crawler. Data processing transfer the download data format that the data mining algorithms or recommendation system can process. Data query is an interface for users to query the database. Data updating is a partner of data collection, which can update the download data simultaneously.

- Layer 3

Layer 3 is the core parts of this system, including data mining algorithms, recommendation algorithms, and so on. This part will be detailed introduction in section III.

- Layer 4

The layer 4 has two main functions. One of them is push recommend videos to user according to MRS results. The other function of this layer is to reflect the processing results to the micro-video producers who will take into account what kind of videos are on their next schedule.

- Layer 5

The layer 5 is the interface layer for users. This layer contains many interfaces for many applications. The micro-video producers can use the micro-video producing interface to push their new video to target users through this platform. We can also use the micro-video evaluation interface to evaluate a video whether the audiences like it or not.

III. THE CORE ALGORITHMS OF MICRO-VIDEO RECOMMENDATION SYSTEM

A. Web crawler

The traditional web crawler starts with a predetermined one or some initial URL seeds. We can get the URL lists from the initial web page. Then, we take out the URL from the lists one by one, and download data from the URL. The last step is to analyze the downloaded data, extract the data what we need and store into database.

The traditional web crawler only downloads and analyzes the static web pages. However, there are many dynamic web pages on the Internet. Dynamic page is a special web page which is running on the server. The contents of the dynamic page are different according to different request. Hence, the traditional web crawler could not get the right contents of the dynamic pages.

AJAX (Asynchronous JavaScript and XML) [15] is a set of web development techniques using many web technologies on the client-side to create asynchronous web applications. Web applications can send data to and retrieve from a server asynchronously without interfering with the display and behavior of the existing. The dynamic web sites using AJAX technology do not provide a meaningful URL that can be used

by the web crawlers. The key problem of web crawler in this paper is to download the dynamic web sites contents automatically.

The Google's AJAX crawling scheme proposes to mark the address of all the pages that load AJAX content with specific chars [16]. The whole idea behind it is to use special hash fragments, such as #! , in the URLs of those pages to indicate that they load AJAX content, for example, "http://example.com/page?query#!state", it automatically interprets it as "http://example.com/page?query&_escaped_fragment_=state". Therefore, the Google's AJAX crawling scheme interprets the real AJAX URLs for web crawler, which can download the dynamic web sites' contents.

Selenium [17] is an automatic web testing tool. It is for automating web applications for testing purpose. We can use the tools for web crawlers to download the contents of dynamic web sites. Selenium can open and close an URL automatically. When selenium opened an URL from a dynamic web site, the web page has already cached in local, i.e. the dynamic web page has been transformed into static web page. Therefore, we can use traditional web crawler to download the web page. However, the performance of Selenium is not good, because we should to wait for selenium open the URL completely, the web crawler can get the contents correctly.

B. Recommendation algorithms

The recommendation problem can be formulated as follows. Supposed C is the set of all users and S is the set of all possible items that can be recommended. Supposed u is a utility function that measure usefulness of item s to user c, i.e., $u : C \times S \to R$, where R is a totally ordered set. Then for each user $c \in C$, we want to choose such item $s \in S$ that maximize the user's utility.

$$\forall c \in C, s_c = \arg\max_{s \in S} u(c, s) \qquad (1)$$

We usually represent the utility of an item by a rating in recommendation system, which indicates how a particular user liked a particular item.

1. Content-based recommendation

Many current content-based recommendation systems focus on recommending items containing textual information. We always use *features* to represent a text document. Feature weighting measure the importance of the feature word. The best-known feature weighting algorithm is TF-IDF (term frequency/inverse document frequency) algorithm [18]. Supposed N is the total number of documents that can be recommended to users, and feature w_i appears n_i of them. Supposed that $f_{i,j}$ is the frequency feature w_i appears in document d_j. Hence, $TF_{i,j}$, the term frequency of feature w_i in document d_j can be defined as

$$TF_{i,j} = \frac{f_{i,j}}{\max_z f_{z,j}} \qquad (2)$$

where the maximum is computed over the frequencies $f_{z,j}$ of all features w_z that appear in the document d_j.

However, the features are not important or not useful, if it appears in many documents. Then the inverse document frequency (IDF_i) of feature w_i is usually defined as

$$IDF_i = \log \frac{N}{n_i} \qquad (3)$$

Therefore, the TF-IDF weight for feature w_i in document d_j is defined as

$$\omega_{i,j} = TF_{i,j} \times IDF_i \qquad (4)$$

So, the recommended document d_j can be represented as a vector, $d_j = \{< t_{1,j}, \omega_{1,j} >, < t_{2,j}, \omega_{2,j} >, ..., < t_{i,j}, \omega_{i,j} >\}$, where $t_{i,j}$ is the i[th] feature in document d_j, and $\omega_{i,j}$ is the $t_{i,j}$'s weight.

In recommendation system, many candidate documents are compared with the input document, and the best-matching documents are recommended. The utility function *u(c,s)*, in formula (1), is usually represented the similarity between the candidate document and the input document, such as cosine similarity measure.

$$u(c,s) = \cos(\omega_c, \omega_s) = \frac{\omega_c \cdot \omega_s}{\| \omega_c \| \times \| \omega_s \|} = \frac{\sum_{i=1}^{K} \omega_{i,c} \omega_{i,s}}{\sqrt{\sum_{i=1}^{K} \omega_{i,c}^2} \sqrt{\sum_{i=1}^{K} \omega_{i,s}^2}} \qquad (5)$$

where, ω_c is the candidate document vector, ω_s is the input document vector, and K is the total number features in the recommendation system.

2. Collaboration recommendation

The collaboration recommendation system is to predict the particular items for a particular user based on the items previously rated by other users. For example, in the micro-video application, in order to recommend videos to user c, the collaboration system tries to find other users who have the similar tastes in the videos that user c liked. The collaboration recommendation usually can be grouped into two general classes: memory-based and model-based algorithms.

The memory-based algorithm is based on the entire collection of previously rated items by the users. That is, in recommend system usually computes the similarity between the particular user and other users for the same item.

First, we need to calculate the similarity among users by particular items. The adjust cosine [19], as following formula, is used in some collaborative filtering methods for similarity among users where the difference in each user's use of the rating scale is taken into account.

$$sim(i,j) = \frac{\sum_{s \in I_{ij}} (R_{is} - A_s)(R_{js} - A_s)}{\sqrt{\sum_{s \in I_{ij}} (R_{is} - A_s)^2 \sum_{c \in I_{ij}} (R_{js} - A_s)^2}} \qquad (6)$$

where, $R_{i,s}$ is the rating of item s by user i, A_s is the average rating of user i for all the co-rated items, and $I_{i,j}$ is the items set both rating by user i and user j.

3. Slope one recommendation

Apache Mahout is an open source machine learning library that consists of a framework of tools allows researchers to create powerful and scalable recommendation and classification applications. Slope one algorithm [20] [21] is a collaborative filtering algorithm in Mahout. The algorithm is easy and more effective than traditional collaborative filter algorithms. Slope one is also based on the similarity of rating different items by the same user. With the development of the Internet, the items of users have increased sharply. The similarity computing complex space of traditional recommendation algorithms increase much and make it hard to realize. Slope one, Mahout, and Hadoop platform are based on Big Data, which can manage the computing of large volume data space, such as the large amount data similarity computing.

The main idea of Slop one algorithm is to find the similarities between two items among the users. If we compute the average rating of the two items, and compare the difference of them, the results can help us to predict another user's rating of those items.

The first step of Slop one algorithm is to calculate the average difference of two items.

$$D_{ij} = \sum_{u_c \in U(I_i \cap I_j)} \frac{R_{ci} - R_{cj}}{\sum_{i=1}^{n}\sum_{j=1}^{m}(I_i \cap I_j)} \qquad (7)$$

where D_{ij} is the average difference, R_{ci} is the rating of the user c to item i, R_{cj} is the rating of user c to item j, I_i and I_j is the i^{th} and j^{th} item, n and m are the total number of item i and item j. In commonly we use a matrix to represent the value of D_{ij}.

The next step of Slope one algorithm is to compute the final prediction. We compute with the matrix of D_{ij}, and predict the target user t to the target item i.

$$P_{ti} = \frac{1}{\sum_{c=1}^{N}(R_{tc} \neq 0)} \sum_{R_{tc} \neq 0} (R_{tc} + D_{ic}) \qquad (8)$$

where, P_{it} is the predicted rating of user t to item i, R_{tc} is the rating of the user t to item c, and D_{ic} is the average difference between item i and c.

4. Slope one algorithm based on MapReduce

MapReduce [22] is a parallel computing model based on Hadoop framework, which contains two stages, the Mapper stage and Reducer stage. It splits the input data into n parts and start n Mappers, and each Mapper deals with one data part. The inputs and outputs of Mapper are key/value pairs. If there are n key/value pairs of Mapper output, then n Reducers need to be started to get the final result.

We need two MapReduc stages and one Mapper stage to implement the Slope one algorithm. The first MapReduce stage of the Slope one algorithm is to compute the difference matrix D_{ij} for each user.

Mapper stage:

Mapper Input: <UserID, List<ItemID, Rating>>

Mapper Output: <UserID, List<ItemID, Rating>>

Reducer stage:

Reducer Input: <UserID, List<ItemID, Rating>>

Reducer Output: <Pair<item i, item j>, D_{ij}>

At the first MapReduce stage, the Mapper does not change anything, and only the Reducer need to compute the difference matrix D_{ij}.

The second MapReduce stage of Slope one algorithm is to compute the average difference matrix, *counts*.

Mapper stage:

Mapper Input: <Pair<item i, item j>, D_{ij}>

Mapper Output: <Pair<item i, item j>, D_{ij}>

Reducer stage:

Reducer Input: <Pair<item i, item j>, D_{ij}>

Reducer Output: <Pair<item i, item j>, Pair<AvgD_{ij}, count>>

The Mapper stage also does not change anything, and the Reducer stage calculates the average difference matrix D_{ij} and the total number of the same Pair<items i, item j>.

The final stage is to calculate the prediction rating of the user. We only use one Map stage to complete the task.

Mapper Input: <UserID, List<ItemID, Rating>>

Mapper Output: <UserID, List'<itemID, Rating>>

The Mapper's input is the user's rating list and output is the prediction rating list.

IV. CONCLUSIONS

With the development, the micro-video are increasingly common, especially young teenagers are likely to watch videos on mobile devices. The challenges what we need at present is how to find the favorite video. Otherwise, for micro-video producers, what they care for is how many viewers like what kind of videos. Based on this spot, this paper proposes a micro-video recommendation system. According to the viewers' browsing or watching history, this system can recommend the favorite videos to the viewers. On the other hand, this system can collect the reactions and give some suggestions for micro-video producers with how many viewers like the micro-video.

The core function of the system is the recommendation algorithms. The commonly recommendation algorithms are suite for tradition date sets, such as content-based recommendation, collaboration recommendation, and so on. However, with the development of Big Data, the recommendation algorithms should have the ability to deal with the Big Data. The Slope one algorithm is a Big Data

recommendation algorithm based on Hadoop framework. The MapReduce programming model is based on Hadoop framework, which can process Big Data. Thus, this paper use MapReduce programming model to implement the Slope one algorithm.

ACKNOWLEDGMENT

This paper is partly supported by "the Excellent Young Teachers Training Project (the second level, Project number: YXJS201508)", "Key Cultivation Engineering Project of Communication University of China (Project number: 3132016XNG1606 and 3132016XNG1608)", "Cultural technological innovation project of Ministry of Culture of P. R. China (Project number: 2014-12)", and partly supported by "The comprehensive reform project of computer science and technology, department of science and Engineering". The research work was also supported by "Chaoyang District Science and Technology Project (CYXC1504)".

REFERENCES

[1] Y. Z. Li, T. Gao, and X. Y. Li, "Design of video recommender system based on clooud computing", Journal on Communications, Vol. 34, No. Z2, pp. 138-140, 147, 2013.

[2] Y. Li, "Development mode of micro-video communication", Academic Exchange, Vol. 248, pp.177-181, 2014.

[3] S. M. Meng, W. C. Dou, X. Y. Zhang, et al., "KASR: a keyword-aware sevice recommendation method on MapReduce for Big Data application", IEEE Transactions on parallel and distibuted system, Vol. 25, No. 12, 2014.

[4] D. M. Zhou, Z. J. Li, "Survey of high-performancee web crawler", Computer Science, Vol. 36, No. 8, pp.26-29, 53, 2009.

[5] G. Y. Su, J. H. Li, and Y. H. Ma, et al., "New focused crawling algorithm", Journal of Systems Engineering and Electronics, Vol. 16, No. 1, pp.199-203, 2005.

[6] X. W. Meng, X. Hu, and L. C. Wang, et al., "Mobile recommender system and their applications", Journal of Software, Vol. 24, No.1, pp. 91-108, 2013.

[7] G. X. Wang, H. P. Liu, "Surevey of personal recommendation system", Computer Engineering and Applications", Vol. 48, No. 7, pp. 66-76, 2012.

[8] G. F. Sun, L. Wu, and Q. Liu, et al., " Recommendations based on collaborative filtering by exploiting sequential behaviors", Journal of Software, Vol. 24 No. 11, pp.2721-2733, 2013.

[9] L. Guo, J. Ma, and Z. M. Chen, et al., "Incorporating item relations for social recommendation", Chinese Journal of Computers, Vol. 37, No. 1, pp. 218-228, 2014.

[10] Y. R. Wang, M. Chen, and H. H. Wang, et al., "A content-based filtering algorithm for scientific literature recommendation", Computer Technology and Development, Vol .21, No. 2, pp. 66-69, 2011.

[11] W. F. Liu, J. Z. Gu, and X. Lin, et al., " A Big Data management and analysis system based on Hadoop and Mahout", Computer Application and Software, Vol. 32, No. 1, pp. 47-50, 2015.

[12] E. Jain, S. K. Jain, "Categoring Twitter users on the basis of their interests using Hadoop/Mahout platform", Industrial and Information Systems (ICIIS), 2014 9th International Conference, pp. 15-17, 2014.

[13] B. Fan, C. Jia, "Visual framework for big data in d3.js", Electronics, Computer and Applications, pp.47-50, 2014.

[14] Y. Z. Wang, S. Mao, "A blocks placement strategy in HDFS", Computer Technology and Development, Vol. 23, No. 5, pp. 90-92,96, 2013.

[15] X. S. Zhang, H. L. Wang, "AJAX Crawling Scheme Based on Document Object Model", Computational and Information Sciences (ICCIS), pp. 1198-1201, 2012.

[16] C. Qian, X. L. Yang, " Design and Realization of the Web Crawler Supporting Ajax Framework", Computer & Digital Engineering, Vol. 40, No. 4, pp. 69-71, 98, 2012.

[17] http://www.seleniumhq.org/

[18] M. Lan, C. L. Tan, J. Su, and Y. Lu, "Supervised and Traditional Term Weighting Methods for Automatic Text Categorization", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 31, No. 4, pp. 721-735, 2009.

[19] S. J. Gong, H. W. Ye, and H. S. Tan, "Combining Memory-Based and Model-Based Collaborative Filtering in Recommender System", 2009 Pacific-Asia Conference on Circuits,Communications and System, pp. 690-693, 2009.

[20] G. R. Bamnote, S. S. Agrawal, "Evaluating and Implementing Collaborative Filtering Systems Using Apache Mahout", 2015 International Conference on Computing Communication Control and Automation, pp. 858-862, 2015.

[21] S. Y. Song, K. J. Wu, "A creative personalized recommendation algorithm—User-based Slope One algorithm" 2012 International Conference on Systems and Informatics (ICSAI 2012), pp. 2023-2027, 2012.

[22] L. N. Li, H. Chen, and X. Y. Du, "MapReduce-Based SimRank Computation and Its Application in Social Recommender System" 2013 IEEE International Congress on Big Data, pp. 133-140, 2013.