I J C T A, 9(2) 2016, pp. 369-381

UPRS: User Preferences-based Recommendation System for Big Data Application

Archana S.[1] and Indira K.[2]

ABSTRACT

In the last period, the quantity of customers, services and online information has developed randomly, yielding the big data analysis tricks for recommender systems. Consequently, services for recommender systems often undergo scalability and problems in efficiency when processing or analyzing such huge-scale data. Additionally, most of existing recommender systems satisfies the same ratings and rankings of items to different users without considering allotted user's preferences, and as a result fails to meet users personalized requirements. This paper proposes a User Preferences-based Recommendation method, to address the above issues. It aims at providing a personalized recommendation list and recommending the most appropriate items to the users effectively. Explicitly, keywords are used to designate user's preferences, and a User-based Collaborative Filtering algorithm is used to produce appropriate recommendations. It is executed on Hadoop, a widely-adopted distributed computing platform using the MapReduce parallel processing paradigm to improve its scalability and efficiency using measures in big data environment.

Keywords: Recommender system, Big Data, user preference, keyword, MapReduce, Hadoop

1. INTRODUCTION

The random development of cloud computing and cloud data stores has been a reason to the emergence of big data. Cloud computing is the commodification of computing time and data storage by means of harmonized technologies. Using cloud infrastructure, we are able to analyse big data makes sense because:

i. Investments in big data analysis can be significant and provides a need for efficient, cost-effective infrastructure.

ii. Big data may combine with internal and external sources.

iii. Data services are needed to expect value from big data.

Big Data

Big Data refers to datasets of size which is beyond the capability of current technology, method and practises to capture, man-age, and process the data within a tolerable transfer time. Now a day, Big Data management stands out as a challenge for IT companies. The appropriate solution to such a challenge is increasing efficiently from providing hardware to provisioning more manageable software solutions [1].

Big data refers to enormous data sets that are orders of degree larger (volume); more different, involving structured, semi-structured, and unstructured data (variety); and arriving more rapidly (velocity). This overflow of data is generated by coupled devices—from PCs and smart phones to sensors such as RFID readers and traffic cams. Additionally, it's diverse and comes in many formats, including text, document,

[1] Department of Information Technology, Madurai, India, *E-mail: archusubburaman@gmail.com*

[2] Department of Information Technology, Faculty of Engineering, Madurai, India.

image, video, and more. Even though, big data does not refer to any particular measure, the term is frequently used when exclamation about petabytes and exabytes of data, much of which cannot be incorporated easily. Figure 1, shows the model of Big Data in the cloud.

The authentic value of big data is in the insights it produces when analysed—discovered patterns, derived meaning, indicators for decisions, and literally the ability to respond to the world with greater intelligence. A set of advanced technologies designed to work with large volumes of heterogeneous data refers Big data technologies. It uses contaminated quantitative methods such as machine learning, neural networks, robotics, computational mathematics, and artificial intelligence to explore the data and to discover interrelationships and patterns.

Figure 1: Model of Big Data in Cloud

Hadoop

Hadoop is one of the Apache open source framework. It allows distributed processing of large datasets across clusters of computers using uncomplicated programming models. The Hadoop framework application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is planned to scale up from single server to thousands of machines, each providing detained computation and storage.

Hadoop Architecture

At its core, Figure-2, Hadoop has two major layers namely:

a) Processing/Computation layer (MapReduce), and

b) Storage layer (Hadoop Distributed File System).

MapReduce- MapReduce is a programming model and for implementation for processing and generating large data sets with a parallel,distributed algorithm on cluster.It devised at Google for efficient processing of large amounts of data (multi-terabyte data-sets), on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner. The MapReduce program runs on Hadoop which is an Apache open-source framework.

Hadoop Distributed File System- The Hadoop Distributed File System (HDFS) is based on the Google File System (GFS) It provides a distributed file system that is planned to run on commodity hardware. It contains various similarities with existing distributed file systems. The differences from other distributed file systems are significant. It is highly fault-tolerant and is planned to be expanded on low-cost hardware. It provides high throughput access to application data and it is appropriate for applications having massive datasets. Apart from the above-mentioned two core components, Hadoop framework also includes the following two modules:

- *Hadoop Common*: It refers to the collection of common utilites and libraries that suppot other Hadoop modules.
- *Hadoop YARN*: This is a framework for job scheduling and cluster resource management.

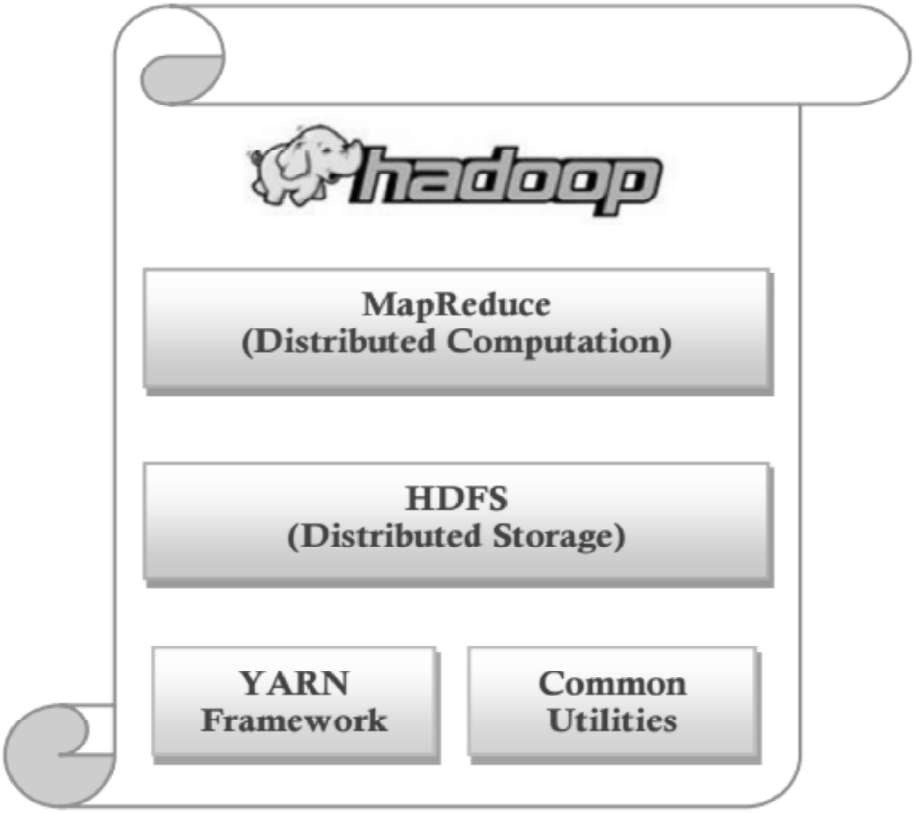
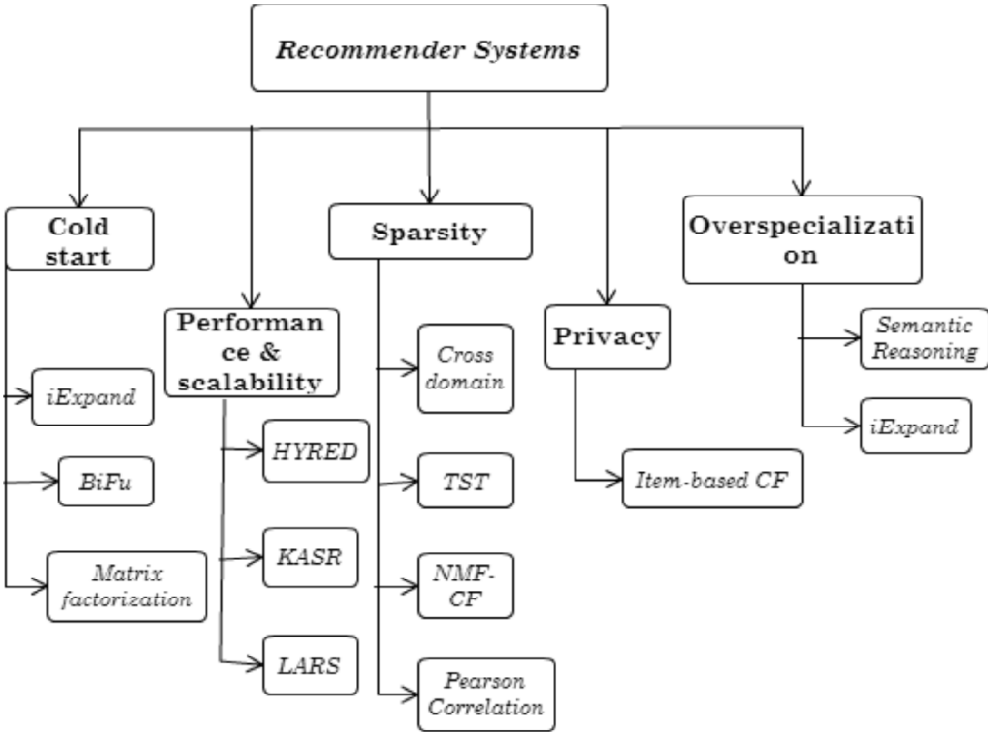


Figure 2: Hadoop Architecture

Service recommender systems have been affirmed as valuable tools to help users deal with services overload and provide suitable recommendations to them. Examples of such practical applications include CDs, books, web pages and various other products now use recommender systems [2], [3], [4]. Over the last decade, there has been much research done both in industry and academia on developing new approaches for service recommender systems [5], [6].

2. LITERATURE SURVEY

(Qi Liu 2012) proposed a novel collaborative-filtering-based recommender system by user interest expansion via personalized ranking, named iExpand. The goal is to build an item-oriented model-based collaborative-filtering framework. The iExpand method introduces a three layers, user–interests–item, representation scheme, which leads to more accurate ranking recommendation results with less computation cost and helps the understanding of the interactions among users, items, and user interests [7].



(Daqiangzhang 2014) propose bi-clustering and fusion (BiFu)-a newly-fashioned scheme for the cold-start problem based on the BiFu techniques under a cloud computing setting. To identify the rating sources for recommendation, it introduces the concepts of popular items and frequent raters. To reduce the dimensionality of the rating matrix, BiFu leverages the bi-clustering technique. To overcome the data sparsity and rating diversity, it employs the smoothing and fusion technique. Finally, BiFu recommends social media contents from both item and user clusters. Experimental results show that BiFu significantly alleviates the cold-start problem in terms of accuracy and scalability [8].

(Sheng gao 2013) proposed a novel cross-domain recommendation model, which not only learn the common rating pattern across domains with the flexibility in controlling the optimal level of sharing, but also learn the domain-specific rating patterns in each domain involving discriminative information propitious to performance improvement. Extensive experiments on real world data sets suggest that our proposed model outperforms the state-of-the-art methods for the cross-domain recommendation task in CPS [9].

(Coello 2013) focused on the construction of collaborative filtering (CF) recommender systems for Web services. The main contribution of the proposed approach is to reduce the problems caused by sparse rating data - one of the main shortcomings of memory-base CF algorithms - using semantic markup of Web services. In the presented algorithm, the similarity between users is computed using the Pearson correlation coefficient, extended to consider also the ratings of users for similarity services. Likewise, to predict the rating a user would give to a target service, the algorithm considers the ratings of neighbor users for the target service and also for similar services. Experiments conducted to evaluate the algorithm show that our approach has a significant impact on the accuracy of the algorithm, particularly when rating data are sparse [10].

(Shunmei Meng 2014) proposed a Keyword-Aware Service Recommendation method, named KASR, to address the above challenges. It aims at presenting a personalized service recommendation list and recommending the most appropriate services to the users effectively. Specifically, keywords are used to indicate users' preferences, and a user-based Collaborative Filtering algorithm is adopted to generate appropriate recommendations. To improve its scalability and efficiency in big data environment, KASR is

implemented on Hadoop, a widely-adopted distributed computing platform using the MapReduce parallel processing paradigm. Finally, extensive experiments are conducted on real-world data sets, and results demonstrate that KASR significantly improves the accuracy and scalability of service recommender systems over existing approaches [11].

(Mohamed Sarwat 2014) LARS*, on the other hand, supports a taxonomy of three novel classes of location-based ratings, namely, spatial ratings for non-spatial items, non-spatial ratings for spatial items, and spatial ratings for spatial items. LARS* exploits user rating locations through user partitioning, a technique that influences recommendations with ratings spatially close to querying users in a manner that maximizes system scalability while not sacrificing recommendation quality. LARS* exploits item locations using travel penalty, a technique that favors recommendation candidates closer in travel distance to querying users in a way that avoids exhaustive access to all spatial items. LARS* can apply these techniques separately, or together, depending on the type of location-based rating available. Experimental evidence using large-scale real-world data from both the Foursquare location-based social network and the MovieLens movie recommendation system reveals that LARS* is efficient, scalable, and capable of producing recommendations twice as accurate compared to existing recommendation approaches [12].

(Dongsheng 2014) proposed privacy-preserving collaborative filtering (PPCF) methods, using computation-intensive cryptography techniques or data perturbation techniques are not appropriate in real online services. In this paper, an efficient privacy-preserving item-based collaborative filtering algorithm is proposed, which can protect user privacy during online recommendation process without compromising recommendation accuracy and efficiency. The proposed method is evaluated using the Netflix Prize dataset. Experimental results demonstrate that the proposed method outperforms a randomized perturbation based PPCF solution and a homomorphic encryption based PPCF solution by over 14X and 386X, respectively, in recommendation efficiency while achieving similar or even better recommendation accuracy [13].

3. SYSTEM ARCHITECTURE

Here, in this paper we provide a User Preference-based Recommendation method (UPRS). In our proposed technique; keywords are used to signify both of user's prepossessions and the quality of candidate services. A User-based Collaborative Filtering algorithm is used to generate appropriate recommendations. This service aims at manipulating a personalized rating of each user service for a user, and then offering a personalized service recommendation list and recommending the most suitable services to the user. Figure-3, represents the architecture diagram of the proposed method.

The main steps of UPRS are:

- Collect user preferences by a keyword-aware approach.
- Keyword extraction process done by two phases as preprocess [14] and keyword extraction.
- Similarity computation is performed using two ways
    i. Approximate similarity computation using Jaccard Coefficient algorithm.
    ii. Exact Similarity computation using cosine based approach [15], [16].
- Using MapReduce process
- Calculating personalized ratings and generating recommendation list

4. MODULE DESCRIPTION

Our architecture undergoes different modules to build user-based recommendation system. It consists of five modules.

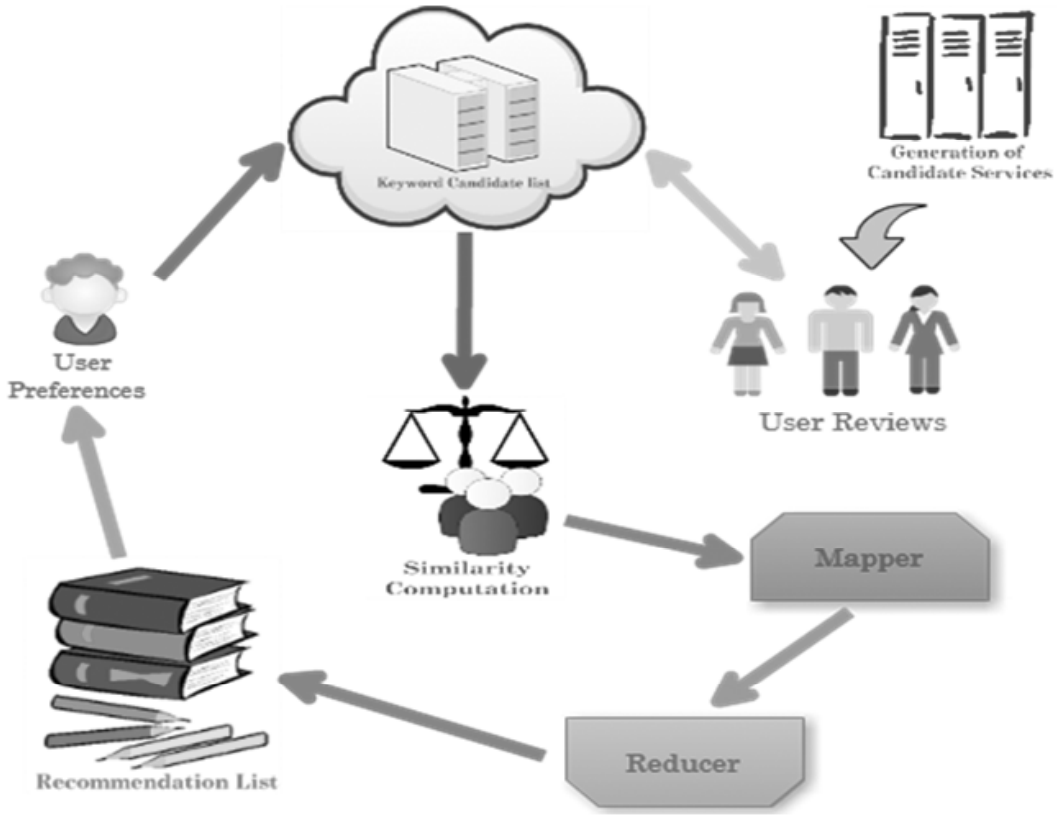


Figure 3: User Preferences-based Recommendation System (UPRS) Architecture

Loading and Preprocessing the data

In this module, we load the data as a dataset D and analyze the data. After the analyzing process, the information present in the dataset D is viewed. The next step is preprocessing step, in which we remove all null values, missing tuples, stop words, HTML tags, etc. To remove the commoner morphological and inflexional endings from words in English we use Porter Stemmer algorithm. Figure-4, represents preprocessing and analyzing the data.

Analyzing User Reviews

After preprocessing, the cleaned or processed data is visualized. At the same time the user reviews are analyzed. The user reviews contain the information about the place or hotels or transportation etc. Using these reviews we further continue our process by calculating similarities as

(i) Approximate Similarity Computation

Algorithm 1: SIM-ASC (Approximate Similarity Computation)

Input: The preference keyword set of the active user APK
The preference keyword set of a previous user PPK_j

Output: The similarity of APK and PPK_j, $sim_{ASC}(APK, PPK_j)$

1: $$sim_{ASC}(APK, PPK_j) = \frac{|APK \cap PPK_j|}{|APK \cup PPK_j|}$$

2: **return** the similarity of APK and PPK_j, $sim_{ASC}(APK, UPK_j)$

A commonly used method for comparing the similarity and variety of sample sets, Jaccard coefficient is smeared in the approximate similarity computation.

Jaccard coefficient is the measurement of asymmetric information on binary (and non-binary) variables, it is useful when no information is given by negative values. The similarity between the preferences of the active user and a previous user based on Jaccard coefficient is termed as follows:

sim(APK,PPK) = Jaccard (APK,PPK)

where, APK - preference keyword set of the active user,

PPK - preference keyword set of a previous user.

The weight of the keywords is not considered in this tactic.

(ii) Exact Similarity Computation

Algorithm 2: SIM-ESC (Exact Similarity Computation)

Input: The preference keyword set of the active user APK
The preference keyword set of a previous user PPK_j
Output: The similarity of APK and PPK_j, $sim_{ESC}(APK, PPK_j)$
1: for each keyword k_i in the keyword-candidate list
2: **if** $k_i \in APK$ **then**
3: get_$\vec{W}_{AP,i}$ by formula (2)
4: else $W_{AP,i} = 0$
5: end if
6: **if** $k_i \in PPK_j$ **then**
7: get_$W_{PP_j,i}$ by formula (5)
8: else $W_{PP_j,i} = 0$
9: **end if**
10: end for
11: get $sim_{ESC}(APK, UPK_j)$ by formula (6)
12: return the similarity of APK and PPK_j, $sim_{ESC}(APK, UPK_j)$

A Cosine-based approach is smeared in the exact similarity computation, which is similar to the Vector Space Model (VSM) in information retrieval [17], [18]. In this approach, the preference keyword sets of the active user and previous users will get transformed into *'n'* dimensional weight vectors respectively, namely preference weight vector, which are denoted as $W_P = [w_1, w_2, \ldots, w_n]$,

where,

'*n*' - number of keywords in the keyword-candidate list,

w_i - weight of the keyword k_i in the keyword-candidate list.

If the keyword k_i is not contained in the preference keyword set, then the weight of k_i in the preference weight vector is 0, i.e., $w_i=0$. The preference weight vectors of the active user and a previous user are noted as W_{AP} and W_{PP}, respectively.

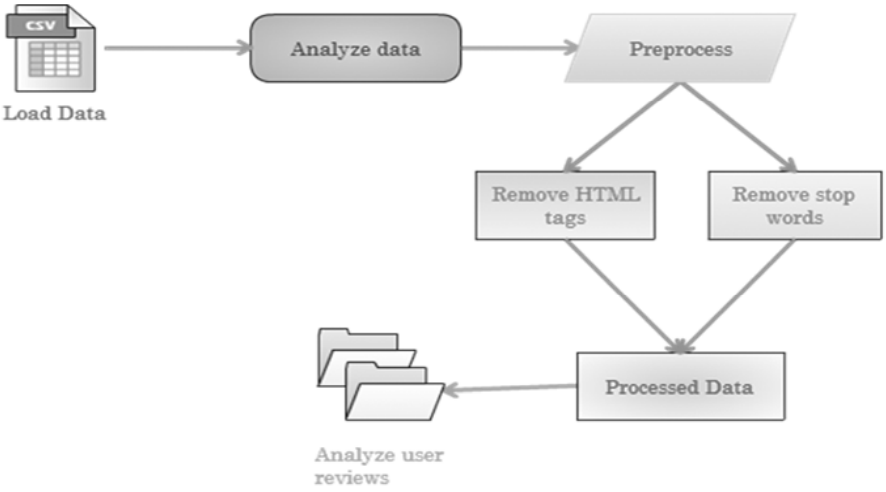


Figure 4: Preprocessing and Analyzing process

Mapper and Reducer process

In this module we first collect the user preferences in the form of query model. We implement the query model to get the user request, which is the user preference. Using the map reduce mechanism, the user preference is processed. The process is performed by splitting the preferences using mapper process Fig. 5. After the processing, the results are aggregated.

Evaluation

After the executions of map reduce process, we merge the results to generate the recommendation list. Using user collaborative filtering algorithm the recommendation list is generated. This algorithm generates the output, recommendation list.

Input: The preference keyword set of the active user APK
The candidate services $WS=\{ws_1, ws_2, \ldots, ws_N\}$
The threshold δ in the filtering phase
The number K
Output: The services with the Top-K highest ratings $\{tws_1, tws_2, \ldots, tws_K\}$
1: for each service $ws_i \in WS$
2: $\hat{R} = \Phi, sum = 0, r = 0$
3: **for** each review R_j of service ws_i
4: process the review into a preference keyword set PPK_j
5: **if** $PPK_j \cap APK \neq \Phi$ **then**
6: insert PPK_j into $\hat{R}$
7: **end if**
8: **end for**
9: **for** each keyword set $PPK_j \in \hat{R}$
10: $sim(APK, PPK_j) = SIM(APK, PPK_j)$
//$SIM(APK, PPK_j)$ can be $SIM\text{-}ASC(APK, PPK_j)$ or $SIM\text{-}ESC(APK, PPK_j)$
11: **if** $sim(APK, PPK_j) < \delta$ **then**
12: remove PPK_j from $\hat{R}$
13: **else** $sum=sum+1$, $r=r+r_j$
14: **end if**
15: **end for**
16: $\overline{r} = r / sum$
17: get pr_i by formula (7)
18: end for
19: sort the services according to the personalized ratings pr_i
20: return the services with the Top-K highest ratings $\{tws_1, tws_2, \ldots, tws_K\}$

Generating Recommendation List

This process takes place to predict the accuracy of the recommendation list. The result is in the form of graph-based models.

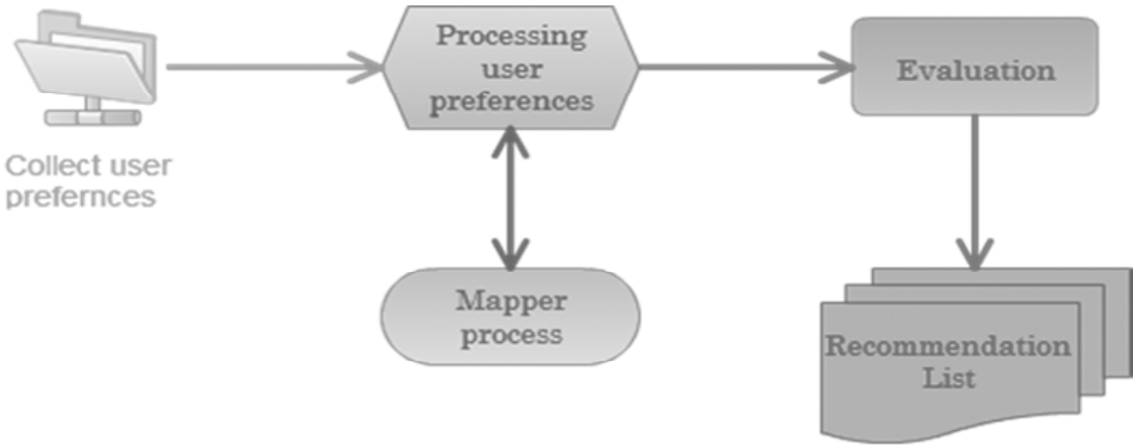


Figure 5: Mapping and predicting process

5. EXPERIMENTAL EVALUATION

In this section, experiments are designed and analysed to evaluate the accuracy and scalability of our method UPRS. To figure the performance of UPRS in accuracy, we compare UPRS with other two well-known recommendation methods: User-based algorithm using Pearson Correlation Coefficient (PCC) and Item-based algorithm using PCC, which are called as UPCC [19] and IPCC [20] respectively. There are three metrics are used to evaluate the accuracy: Mean Absolute Error (MAE) [21], Mean Average Precision (MAP) [22] and Discounted Cumulative Gain (DCG) [23]. As to the scalability, a well-known scalability metric, Speedup [24], is adopted to measure the performance in the scalability of User Preference-based recommendation.

Experiment Setup and Datasets

Absolutely, our experiments are leads in a Hadoop platform. To evaluate the accuracy and scalability of this method, two kinds of dataset are adopted in the experiments: a real dataset and a manmade dataset.

Experiment Evaluation

Two sets of experiments are conducted to evaluate the accuracy and scalability of UPRS. In the major one, we compare UPRS with UPCC and IPCC in MAE, MAP and DCG to evaluate the accuracy of UPRS. The additional one is to explore the scalability of UPRS.

Accuracy evaluation

1) Comparison of UPCC, IPCC, UPRS-ASC and UPRS-ESC in MAE

MAE is a statistical accuracy metric often used in Collaborative Filtering methods to measure the prediction quality. And the Normalized Mean Absolute Error (NMAE) metric is also used to measure the prediction accuracy. The lower the MAE or NMAE presents the more accurate predictions. Figure-6, shows the MAE and NMAE values of UPCC, IPCC, UPRS-ASC and UPRS-ESC. It could be found that the MAE and NMAE values of UPRS-ASC and UPRS-ESC are much lower than UPCC and IPCC. Thus our methods UPRS-ASC and UPRS-ESC can offer more accurate predictions than traditional methods UPCC and IPCC.

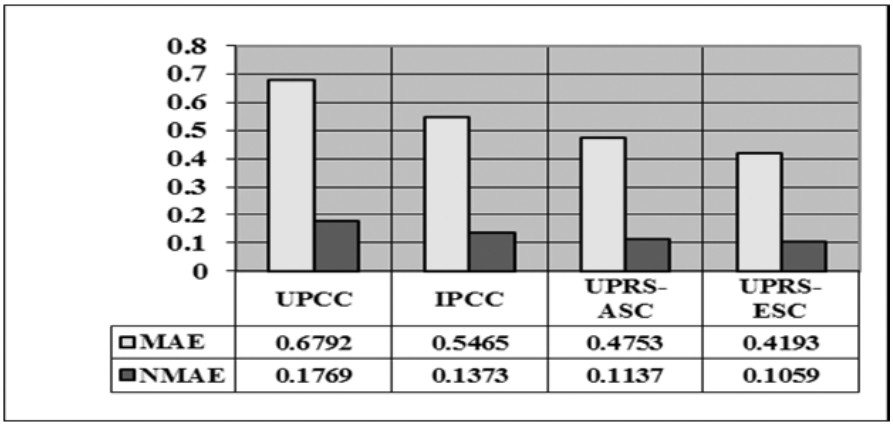


Figure 6: Comparison of UPCC, IPCC, UPRS-ASC, UPRS-ESC in MAE

(2) Comparison of UPCC, IPCC, UPRS-ASC and UPRS-ESC in MAP and DCG

In most of the service recommender systems, users tend to be recommended the top services of the recurred result list. The services in higher position, especially the first position, should be more rewarding than the services in lower position of the returned result list. To evaluate the quality of Top-K service recommendation list, MAP and DCG are used as performance evaluation metrics. And the higher MAP or DCG presents the higher quality of the predicted service recommendation list.

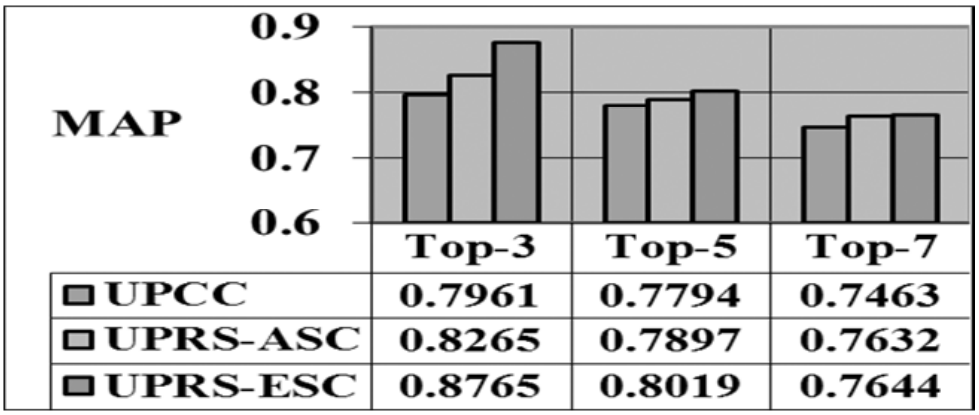


Figure 7. (a)

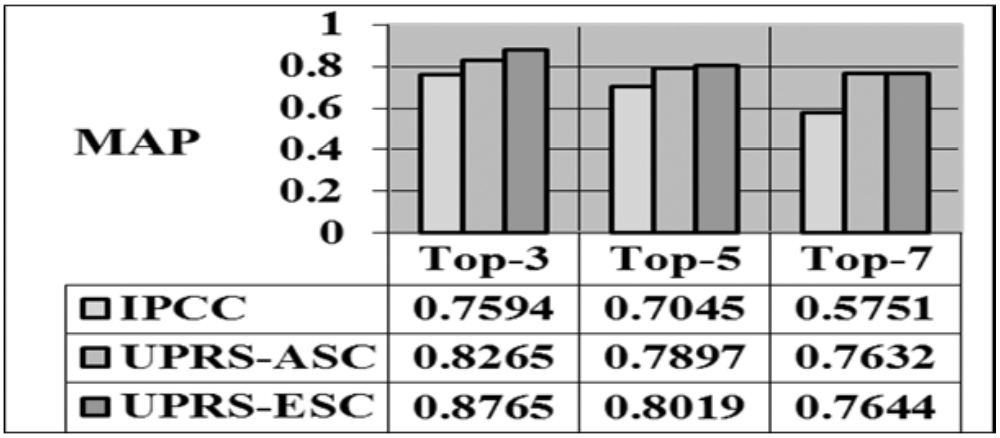


Figure 7. (b)

Figure 7: Comparison of UPCC, IPCC, UPRS-ASC and UPRS-ESC in the MAP values of Top-K (K=3, 5, 7) recommendation list. (a) shows the comparison of UPCC, UPRS-ASC and UPRS-ESC in MAP. (b) shows the comparison of IPCC, UPRS-ASC and UPRS-ESC in MAP

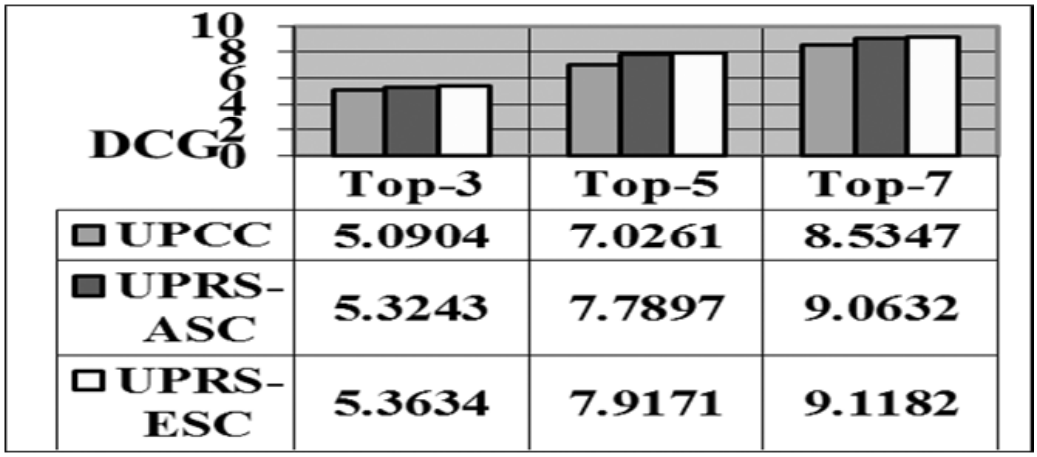


Figure 8. (a)

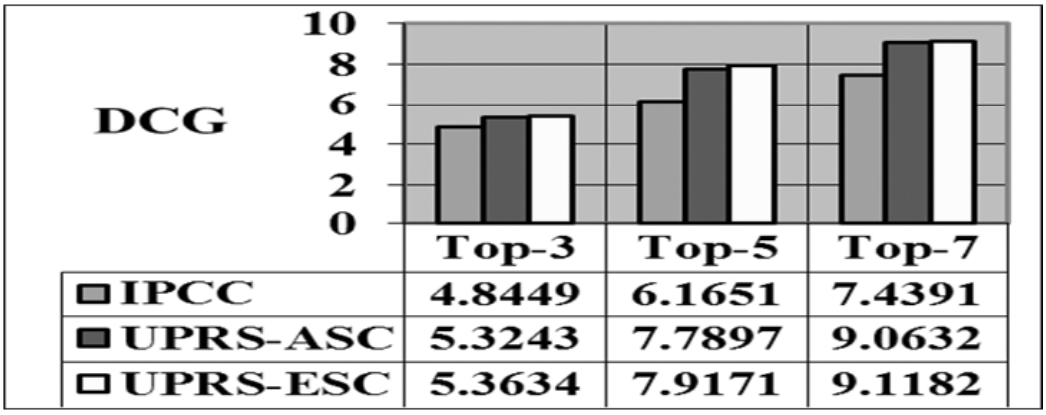


Figure 8. (b)

Figure 8: Comparison of UPCC, IPCC, UPRS-ASC and UPRS-ESC in the DCG values of Top-K (K=3, 5, 7) recommendation list. (a) shows the comparison of UPCC, UPRS-ASC and UPRS-ESC in DCG. (b) shows the comparison of IPCC, UPRS-ASC and UPRS-ESC in DCG

Figure-7 and Figure-8, respectively show the MAP values and DCG values of Top-K (K=3, 5, 7) recommendation list of UPCC, IPCC, UPRS-ASC and UPRS-ESC. From Figure-7 and Figure-8, we can see that the MAP values and DCG values of UPRS-ASC and UPRS-ESC are comparatively higher than UPCC and IPCC. It also could be found that the MAP values decrease when K increases, while the DCG values increase when K increases.

Scalability evaluation

A well accepted scalability metric, Speedup [25], is adopted to measure the performance in the scalability of UPRS. Speedup refers to how much a parallel algorithm is faster than a corresponding sequential algorithm, which can be defined as follows:

$$S_p = \frac{T_1}{T_p} \tag{1}$$

Where, p - number of processors,

T1 - sequential execution time,

Tp - parallel execution time with p processors.

If the speedup has a linear relation with the numbers of nodes with the datasize fixed, the algorithm will have good scalability. To verify the scalability of UPRS (1), experiment is conducted respectively in a cluster of nodes ranging from 1 to 8. There are 4 man-made datasets used in the experiments (128M, 256M, 512M and 1G datasize). The speedup of UPRS increases relative linearly with the growth of the number of nodes. Meanwhile, larger dataset obtained a better speedup. The experimental result shows that UPRS on Map-Reduce in Hadoop platform has good scalability over "Big Data" and performs better with larger dataset.

Overall, these experimental results show that UPRS performs well in accuracy and UPRS on Mapreduce framework has good scalability in "Big Data" environment.

6. CONCLUSION

We have proposed a user preferences-based recommendation method. In the proposed method, keywords are used to indicate user's preferences, and a User-based Collaborative Filtering algorithm is used to produce appropriate recommendations. More expressly, keyword user list and domain thesauruses are provided to help acquire user's preferences. By culling the keywords from the keyword candidate list the active user gives his/her preferences and the preferences of the previous users can be mined from their reviews for services according to the keyword candidate list and domain thesaurus. Our method aims at bestowing a personalized service recommendation list and recommending the most appropriate service(s) to the users. Additionally, to improve the scalability and efficiency of our method in "Big Data" environment, we have employed it on a Map Reduce framework in Hadoop platform. As a final point, the experimental evaluation demonstrates that this method significantly improves the accuracy and scalability of service recommender systems over existing approaches. The recommendation processes of the candidates are condensed in the Map function. The proposed results also show that our design algorithm provides facility for Collaborative Filtering algorithm in Hadoop platform to obtain the efficient performance. Throughout the experiments we find that the Map Reduce framework is that in the calculation process, whenever a new input file(or file blocks), it desires to initialize a mapper, and this process for some algorithms are very resource consuming. The Collaborative Filtering algorithms on Hadoop platform cannot reduce the recommendation response time for a single user.

REFERENCES

[1] C. Lynch, "Big Data: How do your data grow?" Nature, Vol. 455, No. 7209, pp. 28-29, 2008.

[2] G. Linden, B. Smith, and J. York, "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," IEEE Internet Computing, Vol. 7, No.1, pp. 76-80, 2003.

[3] M. Bjelica, "Towards TV Recommender System Experiments with User Modeling," IEEE Transactions on Consumer Electronics, Vol. 56, No.3, pp. 1763-1769, 2010.

[4] M. Alduan, F. Alvarez, J. Menendez, and O. Baez, "Recommender System for Sport Videos Based on User Audiovisual Consumption," IEEE Transactions on Multimedia, Vol. 14, No.6, pp. 1546-1557, 2013.

[5] Y. Chen, A. Cheng and W. Hsu, "Travel Recommendation by Min-ing People Attributes and Travel Group Types From Community-Contributed Photos". IEEE Transactions on Multimedia, Vol. 25, No.6, pp. 1283-1295, 2012.

[6] Z. Zheng, X Wu, Y Zhang, M Lyu, and J Wang, "QoS Ranking Pre-diction for Cloud Services," IEEE Transactions on Parallel and Dis-tributed Systems, Vol. 24, No. 6, pp. 1213-1222, 2013.

[7] Qi Liu, Enhong Chen, HuiXiong, Chris H. Q. Ding and Jian Chen, "Enhancing Collaborative Filtering by User Interest Expansion via Personalized Ranking", IEEE Transactions on Systems, Man and Cybernetics Vol.42, No.1, February 2012.

[8] Daqiangzhang, Ching-hsienhsu, Min chen, Quanchen, Naixuexiongand Jaime lloret, "Cold-Start Recommendation Using Bi-Clustering and Fusion for Large-Scale Social Recommender Systems", IEEE transactions on Emerging Topics in Computing, 2014.

[9] Sheng gao, Haoluo, Da chen, Shantao li, Patrick gallinari, Zhanyu ma, and Jun guo, "A Cross-Domain Recommendation Model for Cyber-Physical Systems", IEEE Transactions on Emerging Topics in Computing, 2013.

[10] J. M. A. Coello, Y. Yuming and C. M. Tobar "A Memory-based Collaborative Filtering Algorithm for Recommending Semantic Web Services", IEEE Latin America Transactions, Vol.11, No.2, March 2013.

[11] Shunmei Meng, Wanchun Dou, Xuyun Zhang, and Jinjun Chen, "KASR:A Keyword-Aware Service Recommendation Method on MapReduce for Big Data Applications", IEEE Transactions on Parallel and Distributed Systems, Vol.25, No.12, December 2014.

[12] Mohamed Sarwat, Justin J. Levandoski, Ahmed Eldawy, and Mohamed F. Mokbel, "LARS*: An Efficient and Scalable Location-Aware Recommender System", IEEE Transactions on Knowledge and Data Engineering, Vol.26, No.6, June 2014.

[13] Dongsheng Li, Chao Chen, Qin Lv, LiShanga, Yingying Zhao, TunLu, NingGu, "An algorithm for efficient privacy-preserving item-based collaborative filtering", ELSEVIER 2014.

[14] Issac and W. J. Jap, "Implementing spam detection using baye-sian and porter stemmer keyword stripping approaches," TEN-CON 2009-2009 IEEE Region 10 Conference, pp. 1-5, 2009.

[15] P. Castells, M. Fernandez, and D. Vallet, "An Adaptation of the Vector-Space Model for Ontology-Based Information Retrieval," IEEE Transactions on Knowledge and Data Engineering, Vol. 19, No.2, pp. 261-272, February 2007.

[16] Y. Zhu, Y. Hu,"Enhancing search performance on Gnutella-like P2P systems," IEEE Transactions on Parallel and Distributed Sys-tems, Vol. 17, No. 12, pp. 1482-1495, 2006.

[17] P. Castells, M. Fernandez, and D. Vallet, "An Adaptation of the Vector-Space Model for Ontology-Based Information Retrieval," IEEE Transactions on Knowledge and Data Engineering, Vol. 19, No.2, pp. 261-272, February 2007.

[18] Y. Zhu, Y. Hu,"Enhancing search performance on Gnutella-like P2P systems," IEEE Transactions on Parallel and Distributed Sys-tems, Vol. 17, No. 12, pp. 1482-1495, 2006.

[19] G. Adomavicius, and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the Stateof- the-Art and Possi-ble Extensions," IEEE Transactions on Knowledge and Data Engi-neering, Vol.17, No.6 pp. 734-749, 2005.

[20] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, "Item-based collabora-tive filtering recommendation algorithms," Proceedings of the 10th international conference on World Wide Web, pp. 285-295, 2001.

[21] K. Lakiotaki, N. F. Matsatsinis, and A. Tsoukis,"Multi-Criteria User Modeling in Recommender Systems", IEEE Intelligent Sys-tems, Vol. 26, No. 2, pp. 64-76, 2011.

[22] Y. Pan, L. Lee, "Performance analysis for lattice-based speech in-dexing approaches using words and subword units," IEEE Transac-tions on Audio, Speech, and Language Processing, Vol. 18, No. 6, pp. 1562-1574, 2010.

[23] G. Kang, J. Liu, M. Tang, X. Liu and B. cao, "AWSR: Active Web Service Recommendation Based on Usage History," 2012 IEEE 19th International Conference on Web Services (ICWS), pp. 186-193, 2012.

[24] G. M. Amdahl, "Validity of the single-processor approach to achieving large scale computing capabilities," Proceedings of spring joint computer conference, pp. 483-485, 1967.

[25] X. Yang, Y. Guo, Y. Liu, "Bayesian-inference based recommenda-tion in online social networks," IEEE Transactions on Parallel and Distributed Systems, Vol. 24, No. 4, pp. 642-651, 2013.