Multimed Tools Appl
DOI 10.1007/s11042-017-4620-2

MCRS: A course recommendation system for MOOCs

Hao Zhang[1] · **Tao Huang**[1] · **Zhihan Lv**[2] · **SanYa Liu**[1] · **Zhili Zhou**[3]

Received: 14 December 2016 / Revised: 28 February 2017 / Accepted: 16 March 2017

Abstract With the popularization development of MOOC platform, the number of online courses grows rapidly. Efficient and appropriate course recommendation can improve learning efficiency. Traditional recommendation system is applied to the closed educational environment in which the quantity of courses and users is relatively stable. Recommendation model and algorithm cannot directly be applied to MOOC platform efficiently. With the light of the characteristics of MOOC platform, MCRS proposed in this paper has made great improvement in the course recommendation model and recommendation algorithm. MCRS is based on distributed computation framework. The basic algorithm of MCRS is distributed association rules mining algorithm, which based on the improvement of Apriori algorithm. In addition, it is useful to mine the hidden courses rules in course enrollment data. Firstly, the data is pre-processed into a standard form by Hadoop. It aims to improve the efficiency of the basic algorithm. Then it mines association rules of the standard data by Spark. Consequently, course recommendation information is transferred into MySQL through Sqoop, which makes timely feedback and improves user's courses retrieval efficiency. Finally, to validate the efficiency of MCRS, a series of experiments are carried out on Hadoop and Spark, and the results shows that MCRS is more efficient than traditional Apriori algorithm and Apriori algorithm based on Hadoop, and the MCRS is suitable for current MOOC platform.

Keywords MOOC · Online course · Course recommendation · Apriori · Hadoop · Spark · Distributed computation

✉ Zhihan Lv
lvzhihan@gmail.com

[1] National Engineering Research Center for E-Learning, Central China Normal University(CCNU), Room 419,Science Hall, 152 Luoyu Road, Wuhan 430072, People's Republic of China

[2] Department of Computer science, University College London, London, UK

[3] Nanjing University of Information Science & Technology, Jiang su, Nanjing, China

Published online: 03 April 2017

1 Introduction

Over the past decade, recommendation systems (RSs) [26], have been widely used in social and shopping platforms with the increasing Internet information. In 2012, after the first year of MOOC, the its popularity spreads rapidly around the world. The top universities in the United States and their professors built several MOOC platforms, such as Udacity, Coursea and edX, and became the leaders in this field [11]. In recent years, a large number of MOOC platforms emerged in China, like NetEase cloud classroom, xuetangX, IMOOC and so on. Take xuetang X for an example, the number of its users has reached millions. Facing enormous course information, users cann't understand course information appropriately due to the deficiencies in their knowledge structure and cognitive level. In the complicated network information space, users are easily disoriented by the information overload [16]. Most online education platform basically offers courses retrieval based on text and knowledge points, instead of user's own knowledge ability, learning methods and other dimensions [8]. Course recommendation system can recommend appropriate courses to user by analyze user-related data, recommend appropriate courses to user, for example, K-Means and Apriori algorithm are used to analyze learning situation of users, and recommend relevant learning materials [3], and analyze the relationship between courses for course recommendation.

Applying EDM [17] in an education system can improve the quality of education, cluster algorithm can be used in group education, and makes education decisions more directional [20]. The essence of course recommendation is Educational Data Mining, in order to promote the process of EDM research, we need to optimize data mining algorithm, at present, there are many researches about course recommendation and recommendation algorithm. Zhou [28] proposes three types of recommendations, including course recommendation, learning resource recommendation and learning strategy recommendation. In closed educational environment, Wang [21] uses decision tree algorithm to analyze 5 Information of students: gender, personality, cognitive style, learning style, and the grades of last semester, and recommend most suitable course learning sequence for students, and improve the efficiency of education. In the early 21 century, Internet became popular, and also open education environment, for example, intelligent tutoring system and computer support collaborative learning system, students can communicate with each other. Aher [3] uses K-means and Apriori algorithm to analyze course enrollment data of student in MOODLE, the experimental results show that the recommended courses are in accordance with actual data of students, and data mining technology can help students to enroll courses. Salehi [18] proposes a method that recommends learning resources to students based on collaborative filtering and genetic algorithms, experiment show that a variety of learning resources can be recommended to users, the recommended content belongs to intelligent guidance system. In computer-supported collaborative learning, García [9] proposes a data mining method based on collaborative filtering and association rule, and recommends teaching improvement information to teachers after analyzing learning data. In paper [22], recommending the learning sequence of course according to the characteristics of students, however, these course recommendation systems should consider the change of data quantity in MOOC education environment. The number of users and courses is no longer fixed, in combination with the traditional research theory to build course recommended system of the current environment, be sure to consider the efficiency of calculation, and cloud computing is a way to improve the efficiency of computing [27].

In big data environment, the educational environment has been greatly changed. Since the first year of MOOC, the speed and quantity of learning data has increased a lot. Traditional

course recommendation system needs to combine big data framework, such as Hadoop, Spark and other distributed computing platforms. Wen [24] proposes that using N-gram model and association rules to mine the user behavior data on MOOC platform, for example, the user's online time statistics is related to user's login session, analyzing the user's mouse click behavior, obtain the user's personality characteristics, and providing personalized help. Wassan [23] gives a data mining model of association rules based on MapReduce, but it lacks further research, and concluded that construct education of big data platform needs to rely on Hadoop, MongoDB, Cassandra and other tools to improve the efficiency of educational data analysis. Many typical educational methods require educators to do some analysis, and then feedback to students, but in MOOC platform, education data will be produced at any time, the amount of data is no longer as before. Hou [10] analyzes the characteristics of students to recommend courses according to the students' context information, but it lacks of relationship analysis in different students and different courses. Denley [7] analyzes the grades of each student and gives a personalized recommendation in upcoming learning. West [25] mentions in big data environment, the education platform should have the ability of real-time analysis and feedback for student timely. In the current open educational environment, the key point of designing a course recommendation system is changing the way of calculation. It is significant to research the course recommendation algorithm based on distributed computing framework.

As big data processing is concerned, newly built cloud clusters meet the challenges of performance optimization focusing on faster task execution and more efficient usage of computing resources [14]. In the process of applying data mining algorithm into MOOC platform, it is necessary to consider the characteristics, large number of data and the fast generation. The Apriori algorithm mentioned in paper [3] used for course recommendation is based on closed educational environment. The number of data is small, and produced in a fixed time, do not need to use distributed computing framework, after combined with the current educational platform, the related recommendation algorithms mentioned in paper [17] and [18, 21, 28], applied to the MOOC platform based on the distributed computing environment. The course recommendation algorithm proposed in this paper is based on Apriori algorithm on Spark platform. In the process of data mining, firstly, Sqoop imports data into HDFS, and then preprocesses the data into a standard form by Hadoop, Spark will read data from HDFS to memory for calculating, and the efficiency of the memory-based computation model is higher than Hadoop computation model. The analysis of course enrollment information is divided into two steps, the first step is to mine 1-frequent item sets as the initial candidate data set, and the second step is to mine k-frequent item sets according to the support until the condition is not satisfied, the results of analysis are stored in database as course recommendation rules. Experiments show that the appropriate courses can be recommended for some users. The main contributions of this paper are as follows:

1) Based on distributed theory and traditional course recommendation model, propose the course recommendation model oriented MOOC platform, MCRS, which greatly improves the data storage level and efficiency of calculation, and can be better applied to a course recommendation in MOOC platform.
2) According to the parameters, the relationship between courses can be analyzed reasonably and efficiently, and the experiment shows that the algorithm is more efficient than the traditional Apriori algorithm and Apriori algorithm based on Hadoop [6].

3) Analyze the data of the actual course enrollment data, mining rules of the course recommendation, and verify the rules with the history course enrollment data. The result shows that the rules are valid.

2 MOOC oriented course recommendation model

2.1 Recommendation models

As information consumer, it is difficult to accurately locate the satisfactory information. Information producers are also difficult to directly push their own information to the user who needs this information. Recommendation system is helpful to solve this problem, the evaluation dimension of recommendation system includes three aspects: user dimension, item dimension, user preference dimension. Recommendation model deals with the related data, and recommends appropriate items to the appropriate user, the working principle of the recommended system is shown in Fig. 1.

Learning on MOOC platform will produce a lot of log information, as the user's behavior log can be summarized into the user's behavior data, in traditional recommendation system, paper [1] gives a formal definition of recommendation system: assuming C is a set of users, S is the set of recommending resources, S and C can be very large, function u () is used to calculate the effectiveness of S on C, u:C × S → R, R is a range of all nonnegative numbers, the problem of recommendation is to find the objects S with the max R.

$$\forall c \in C, s_c' = \underset{s \in S}{\operatorname{argmax}} u(c, s) \tag{1}$$

The practical problem solved by recommendation algorithm is the calculation of u, selecting S according to the properties of resources, recommendation algorithms can be classified into content-based recommendation algorithm, collaborative filtering recommendation algorithm, knowledge -based recommendation algorithm, combined recommendation algorithm [2]. The story of beer and diapers is widely used in a recommendation system.

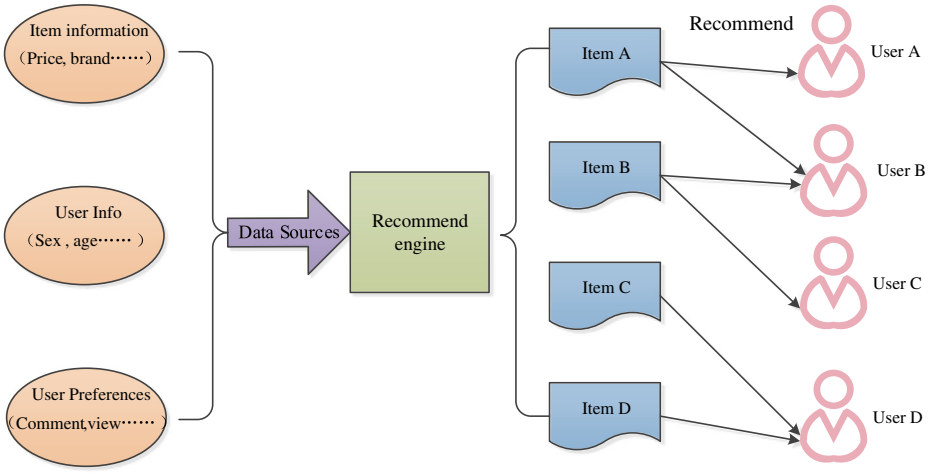


Fig. 1 The principle of recommendation system

Dangdang analyzes the user's shopping cart, and finds the rule that is similar to "users who bought book A are always would like to buy book B ". When users browse the book A, the recommendation system will recommend the book B if other users purchased the book A, this recommendation algorithm based on the analysis of user's behavior is generalized to the recommendation algorithm of collaborative filtering.

The collaborative filtering recommendation algorithm is divided into two steps, the first step is to find other user that is similar to current user C, and calculates effectiveness of resource S. The second step is to sort S by the effectiveness or other operations, and find the suitable resources to recommend. The recommendation system based on collaborative filtering spontaneous recommends resources according to the behavior of users, the user does not need to set parameters to find relevant information, the user's recommendation is implicitly obtained from the information that the user browsed. Recommendation algorithms can be divided into two categories: heuristic methods (Memory-based methods) and model-based methods.

The basic idea of heuristic algorithm is to predict the effectiveness of the resource S on the new user C by evaluating the effectiveness of the resource S on the user C′ who is similar to the new user C. The core of heuristic algorithm consists of two points: calculating the similarity between users, selecting similar users and calculating the effectiveness of resources S on new users. The similarity sim (c, c ') of users can be evaluated by the difference of the user's interest and behavior. The basic calculation method is based on correlation and cosine distance. The method based on correlation, calculates the association between users according to the difference of the reviews on resource S. The method based on cosine distance, regards reviews as a vector to calculate the cosine distance to get similarity [19]. Equation 2 introduces the weight of user's similarity in Fig. 2.

$$r_{c's} = k \sum_{c' \in C} sim\left(c, c'\right) \times r_{c's} \tag{2}$$

The recommendation principle of both is based on the theory of recommendation system, analyzing user's behavior, course information and user preferences, and then recommend the most effective course to user. The difference of both is that the traditional course recommendation system is based on closed educational environment, the number of course and user is basic fixed, course recommendation system does not require the use of big data framework. But the MOOC oriented course recommendation system, the number of course presents as PB-scale. In the process of transplanting traditional course recommendation model into MOOC environment, which needs to combine with big data framework. Based on the above analysis,

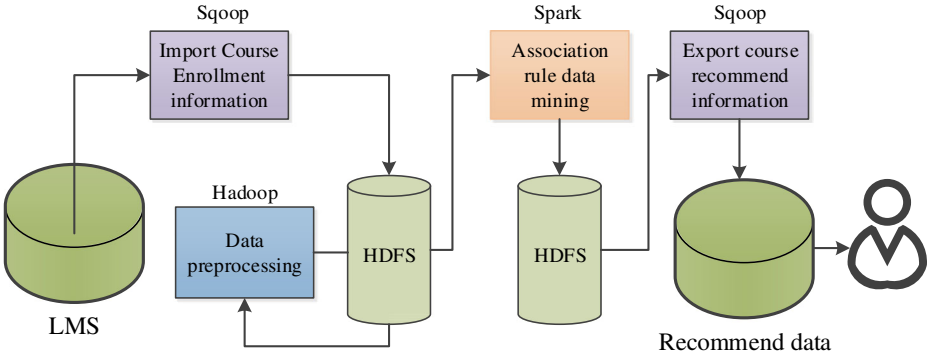


Fig. 2 Course recommendation model

MCRS is proposed in this paper, which is MOOC oriented Course Recommendation System, the whole process of the system consists of three parts:

1) Data collection and preprocessing. Course enrollment information of student imported into Hadoop file system through Sqoop, the form of original data may be not uniform, cannot be efficient processing, and preprocessing the data in Hadoop, simplified data Stored in HDFS;
2) Analysis of association rules. Spark can directly read HDFS data to memory for calculation, combined with the advantages of both, it is easy to analyze the association rules of formatted data, and the results stored in HDFS.
3) Recommending course according to the parameters. Filtering the results according to confidence, and exporting the results to relational database through Sqoop, as the course recommendation information, if there are students in accordance with the rules of courses, and the course is recommended.

Course recommendation model is the core of course recommendation system, the essence of the model is course data mining, in paper 4–8, the recommendation algorithm mentioned above includes collaborative filtering, association rule, decision tree algorithm, clustering algorithm, genetic algorithm, which are commonly used in data mining. In paper 4, the recommendation model is combined with Apriori algorithm and K-means algorithm, the course enrollment data calculated by this model are most consistent with historical data, it shows that the course recommendation system can help students to select courses. The interpretation of system model is also based on the traditional educational environment, the design of course recommendation system is based on the principle of the recommendation system, the core is still the analysis of user and course content, Eq. 1 calculates the maximum of R and recommending most suitable courses to user. In MOOC environment, analyzing the mass of user's course enrollment data, extracting the common characteristics of course enrollment data.

One of the theory of collaborative filtering algorithm is association rules mining. The difference is that collaborative filtering is used for user's preferences, Eq. 2 of heuristic method calculating the weighted effectiveness of resources by the different similarity of users. The association rule is used for transactions, and has stronger constraints on data. The similarity is regarded as the same, regarding users as a subset, analyzing the association rules, and mining K-frequent item sets.

By the data pipeline, Sqoop imports the processed data to traditional database, such as MySQL, sorting by confidence, confidence level represents the relationship between courses, according to confidence, MOOC platform can recommend most related courses, this automatically recommended function can improve the efficiency of retrieving related learning resources on MOOC platform.

2.2 Calculation model

Spark (http://spark.apache.org/) is designed by Scala, the resource allocation and scheduling are managed by Resource Manager, Sparker is different from Hadoop, but stores data in memory, which can improve the efficiency of calculation. So spark is very suitable for iterative and interactive computing, such as association rules mining algorithm, Apriori.

In the process of implementing Spark program, there is an important distributed data structure. Resilient Distributed Dataset. RDD, it is a distributed memory abstraction, not only

can be created in batched, but also can read and write to any location in memory. It can achieve efficient fault tolerance, and does not need to rollback operation for the failure calculation, only need to recalculate the missing part of RDD partition. BlockManager abstracts data into data blocks, stored in memory or disk, if the data is not in this node, it can copy the data from the remote node for calculation. Physically, RDD is actually a metadata structure, and is similar to the data exist in Hadoop namenode, the form of storage as show in Fig. 3. RDD is created in 4 ways, and the common way is input from Hadoop file system or other persistent storage system, such as Hive, Cassandra, Hbase and others. Operations on RDD involve two types of operators: Transformation operator and action operator. In order not to run out of system memory quickly, Spark using the implementation of delay, and only the operation accumulated to action operator. Transformation operator will transform one RDD to another RDD. Action operator will really trigger the execution of the entire sequence of operations, the middle of the result will not re-allocate the memory, but all of the operations are in the same data block.

As shown in Fig. 4, the Spark application consists of seven basic components. The Driver Program is the main function of the Application and creates SparkContext. RDD is the core structure of Spark. Jobs triggered by the Action operator are submitted to the Spark through the RunJob method in SparkContext. Each job is divided into a number of stages according to wide dependency of RDD, and each stage contains a set of same Task, called TaskSet. A partition corresponds to a Task, the Task executes the operators contained in the corresponding Stage of the RDD, the task is packaged by the operator and is placed in the thread pool of Executor.

The Spark execution mode includes Standalone, YARN and Mesos, by default, Spark cluster runs the applications in Standalone mode, and uses FIFO order in scheduling, each application will occupy the resources of all available nodes, you can configure ‘spark.cores.max’ to control the number of CPU cores that an application can apply in the entire cluster. In the Mesos mode, you need to set the “spark.mesos.coarse” to true and change the it to coarse-grained scheduling mode, when Spark runs on the YARN platform, users can set the number of ‘-num-executors to’ the number of application-allocated Executors during program execution. Different operating modes achieve the same basic functions, but use different names in specific parameter settings, and the scheduling mechanism is different.

By default, Spark’s scheduler executes the job in FIFO. Each job is divided into several stages. The first Job occupies all of the available resources, and the second Job occupies the remaining free resources, and so on. In order to achieve the parallel operation of several jobs,

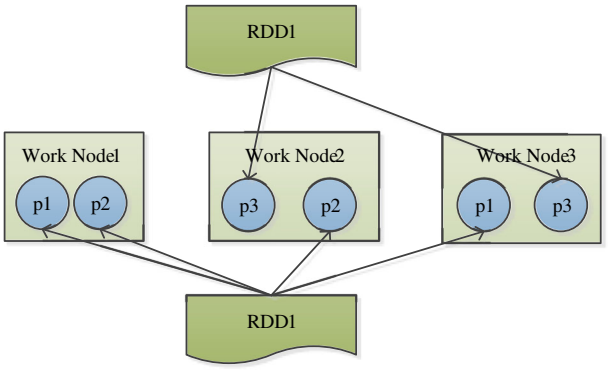


Fig. 3 RDD data management model

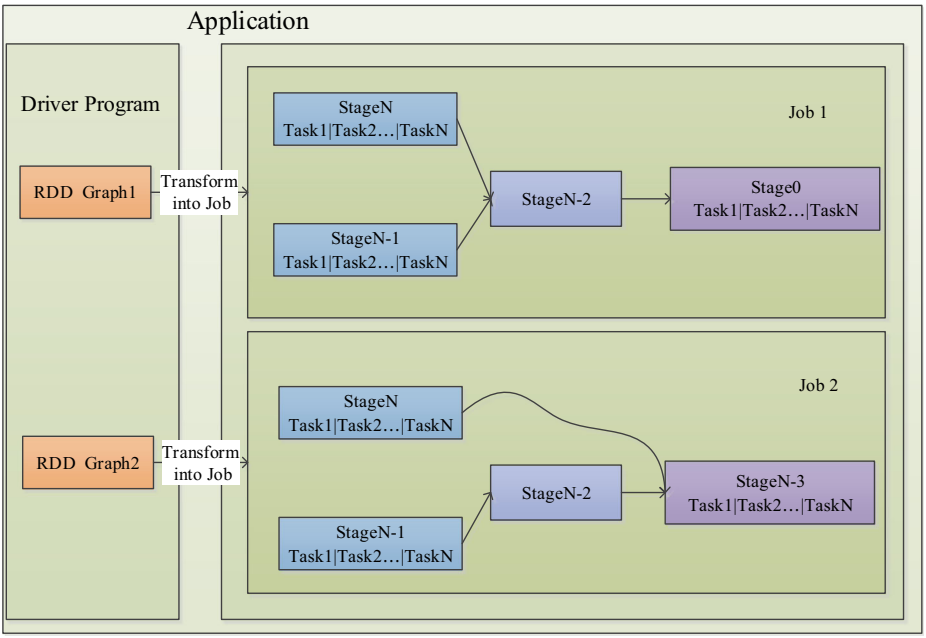


Fig. 4 Spark Application component

after the 0.8 version, you can configure the FAIR scheduling mode, and the resources is obtained in polling between Jobs. You can use FIFO or FAIR mode to con-figure the scheduling, and configure the different weight for jobs to get different number of resources, other para-meters can be set through the 'fairschedular.xml'.

3 The analysis of algorithm

3.1 The principle of algorithm.

Apriori [13] is an association rule mining algorithm proposed by Rakesh Agrawal and Ramakrishnan Srikant in 1994. The basic idea is a recursive algorithm based on the idea of two-phase frequent item sets. The purpose of association rules is to find the relationship between items and items in a data set, also known as the Market Basket Analysis.

AP algorithm includes the following basic concepts:

1) Transaction Database: storage of two-dimensional structure of the record set. Defined as: D
2) Items: Collection of all items. Defined as: I
3) Transaction: a record in the database. Defined as: T, $T \in D$
4) Item set: the emergence of the items collection. Defined as: k-item set
5) Support: defined as sup(X) = occur(X) / count(D) = P(X)
6) Confidence/Strength: defined as conf(X- > Y) = sup(X∪Y) / sup(X) = P(Y|X)

7) Candidate item set: The Item set resulting from merging downwards. Defined as C [k]
8) Frequent item set: An item set whose support is greater than or equal to the specified minimum support. Denoted as L [k].
9) Pruning: filtering through the support.

The concrete calculation process is shown in Fig. 5. Firstly, the dataset is scanned and stored as the original data set which read each item then, and calculate the number of the same item. 1-Frequent item set is obtained by pruning according to the support. 2-candidate item set is obtained from the permutation and combination of the 1-frequent item set's keys. 2-frequent item set is obtained by pruning the 2-candidate item set according to the support. By this recursive analogy, K-frequent item sets are calculated.

In big data environment, the course recommend- ation system can also be divided into three phases. The course recommendation model is shown in Fig. 2. Compared with the traditional course recommendation system, Hadoop ecosystem is used during data collection and preprocessing. The original data is stored in MySQL or Mongo, imported into HDFS through Sqoop (http://sqoop.apache.org/). In pre-processing phase, the data format is mainly arranged and simplified. This process can be easily implemented by the MapReduce program in Hadoop, defining the special key / value pairs [5] types, transforming the data into the type as Student_id: course_id, course2_ id, course_id .....

The algorithm can be divided into two phases: the first phase is a simple WordCount process that calculates 1-frequent item sets according to the support; the second phase is to iterate the original data, and output all the K -frequent item sets that satisfied the support:

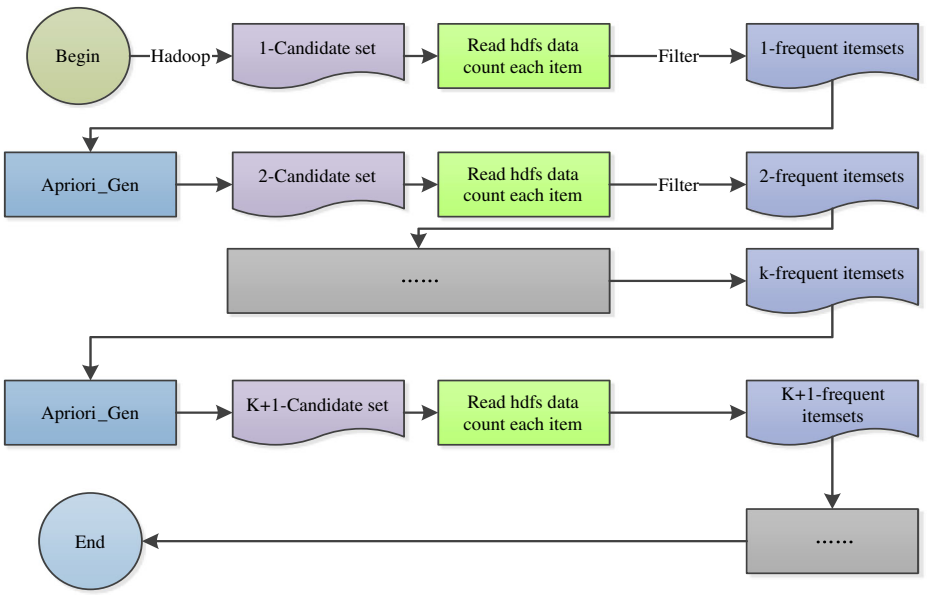


Fig. 5 The principle of apriori

Input parameters: data set D, Support nonsupport, output path output path
Output parameters: K-frequent item sets
1. read the data into memory, form the original data set RDD
2. program begin with the run method The first calculation only calculates the 1-frequent item sets
3. iteratively calculating the data sets, and calculating 1-frequent items according to the support
4. in the second phase, k-frequent item sets are calculated according to k-1 frequent item sets
5. calculate 2-candidate sets according to the key of 1- frequent item set
6. iteratively calculating the 2-frequent item sets according to the 2-candidate sets and the support
7. if 2-frequent item set exists, then calculate 3-candidate sets
8. similarly, if k-1 frequent item sets exist, then calculate k-candidate set
9. if the k-1 frequent item sets does not exist, the calculation is stopped

3.2 The implementation of algorithm

The implementation of algorithm is divided into two phases. In the first phase, frequent item sets are calculated as base data for calculation as shown in Fig. 6. The second phase is a cycle computation of the entire K-frequent item sets. K-frequent item sets add the process of mining multi-item sets in map process as shown in Fig. 7. Each map processes in accordance with the K item sets .

The first phase:

Input: Data set D,
Output: K-frequent item sets
Parameters: data set path, support, output path
1) Read data from hdfs
2) RDDItem = Transactions.flatMap(String line)
Foreach item in line
list.add(item)
End Foreach
Return list
3) RDDItemPair = RDDItem.mapToPair(String s)
Return Tuple2(s,1)
4) reduceItem = RDDItemPair. reduceByKey(int a,int b)
return a + b
5) filterItem = ReduceItem.filter(minsuport)
6) format filterItem
return filterItem.1 and filterItem.2/transactions.count
7) filterItem.save(outputpath)
8) itemSet = hashSet.add(filterItem.1)/* The distributed RDD data is converted to the traditional set by the collect operator. *

The second phase:

Input: Dataset transaction
Output: K-Frequent item sets (K > 1)
Parameters: support, output path
1) go = true
2) itemSetk = apriori_gen(itemSetk-1)
3) While(go)
4) RDDItem = Transactions.flatMap(String line)
Foreach item in itemSetk
If line contains(item)
list.add(item)
End If
End Foreach
5) RDDItemPair = RDDItem.mapToPair(String s)
return Tuple2(s,1)
6) reduceItem = RDDItemPair. reduceByKey(int a,int b)

```
    return a + b
7)    filterItem = ReduceItem.filter(minsuport)
8)    format filterItem
       return filterItem.1 and filter Item.2/transactions.count
9)    If filterItem.size > 0
      filterItem.save(outputpath)
      End If
      Else
          go = false
      End Else
10) itemSet = collect hashSet.
      add(filterItem.1.splitBy(“,”))
11) End While
```

Time complexity analysis:

Time consumption is mainly cost in reading data. Frequent mining is a series of simple accumulation and filtering operations. The following assumptions basic parameters, the basic parameters are assumed below:

1) x,the number transactions, that is, the number of rows of data;

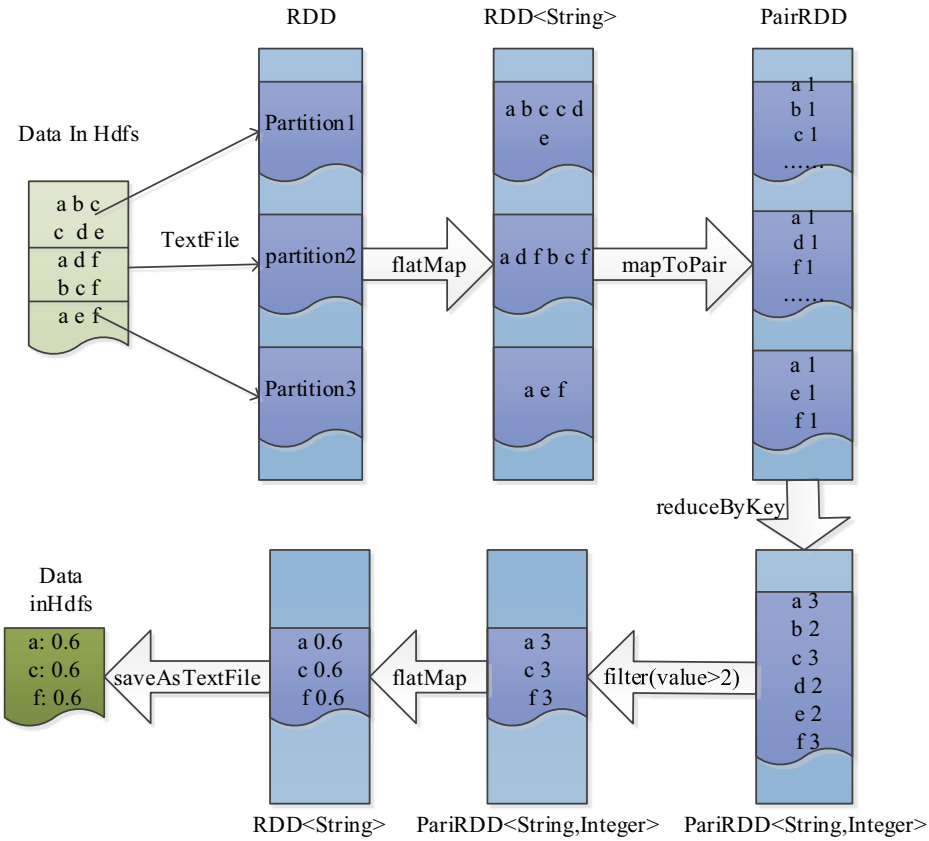


Fig. 6 Mining 1-frequent item sets

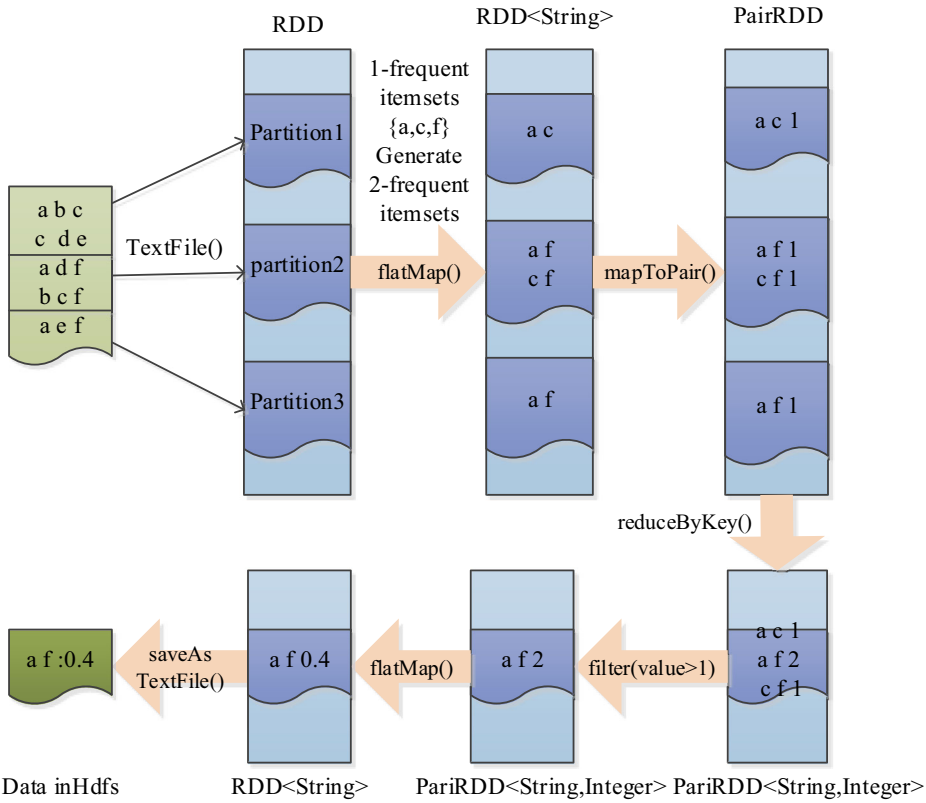


Fig. 7 Mining 2-frequent item sets

2) m,the number of mapper assigned
3) g,the average amount of data contained in each transaction
4) f, the number of parameters contained in the candidate set from the last frequent mining
5) the time it takes to find an element in a hashtree

The whole calculation process can be divided into the following time periods:

1) the time for K + 1 candidate sets generated from K candidate sets: $T_g = O(f^2)$
2) the time cost to save candidate sets into the HashSet: $T_h = O(f^2)$
3) the time consumed by the Transformation operator:

$$T_f = \frac{x}{m} \times \frac{tg(g-1)}{2} = O(x/m.tg^2).$$

Table 1 Data set

Name	The number of item	The number of transaction
HMXPC13	5	119,024
T10I4D100K	870	100,000
T25I10D10K	990	4900

Table 2 Course information

Course number	Course ID	Course name
1	HarvardX/CB22x/2013_Spring	The Ancient Greek Hero
2	HarvardX/CS50x/2012	Introduction to Computer Science
3	HarvardX/ER22x/2013_Spring	Justice
4	HarvardX/PH207x/2012_Fall	HealthStat
5	HarvardX/PH278x/2013_Spring	HealthEnv

The total time cost: $T_g + T_h + T_f = O(f^2) + O(f^2) + O(x/m.tg^2) \approx O(f^2 + x/m.tg^2)$, it can be seen that the performance of the experiment is most closely related to the number of transaction.

4 Experimental analysis

The experiment, compared with the Apriori algorithm based on Hadoop [12], the basic algorithm of MCRS is an association rule mining algorithm based on Spark. Experiment platform was established on windows7, 8G memory Lenovo computer, and set up three 64-bit Centos6.7 virtual machine as a cluster, 1 master node and 2 slave node, each node's memory is allocated 2G.

4.1 Data set

Three benchmark datasets were used during the experiment, IBM provides the dataset for Apriori algorithm,T10I4D100K (http://fimi.ua.ac.be/data/),T25I10D10K (http://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php)] (generated by a random iteration of the database), Harvard and MIT published the edX learning data in 2012–2013, HMXPC13_DI_v2_5–14–14.csv (https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/26147)]. The information of three data sets is shown in Table 1, the edX dataset has 135,205 learning data, and contains 21 attributes, but association rule analysis only selects student course enrollment data. The data is preprocessed before analyzed, customizing the special Key Pair, taking student ID as a unique identifier, counting each student's course enrollment as Value, and input to the HDFS in the form of Key, Value, as the input data of experiment. The data's course information is shown in Table 2. After processing the data in Hadoop, the data has 119,024 transactions.

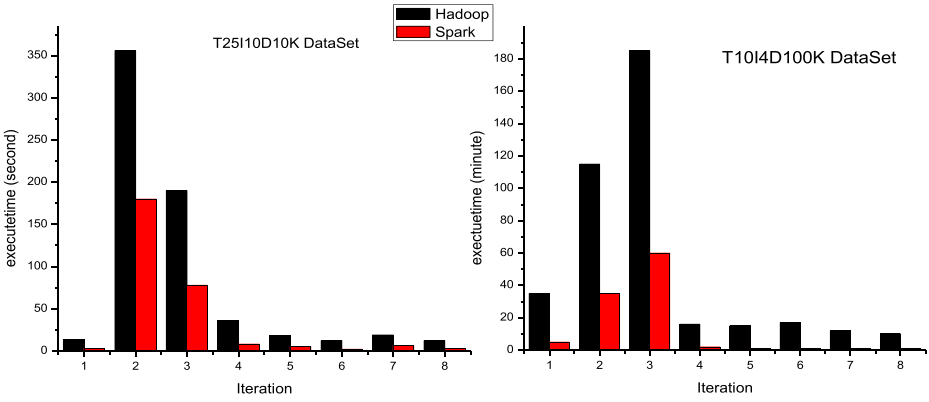
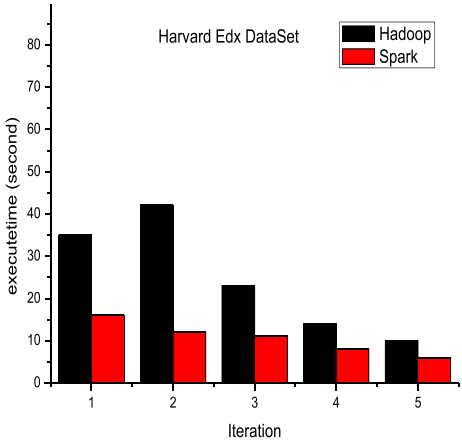


Fig. 8 The execution time on Spark and Hadoop

Fig. 9 HMXPC13



4.2 The result of experiment

4.2.1 Algorithm efficiency experiment

The minimum support is set to 100/transaction in the experiment. It starts with the first item, until the items no more than 100. The maximum frequent item set is set to 8 items, and each experiment was carried out three times to reduce the deviation of the experiment, collecting the results of the experiment, recording the time cost for each experiment and mining the k-item set. The left side of Fig. 8 shows the experimental results of T25I10D10K data sets. Spark execution time is 5 min, and Hadoop execution time is 10 min. The difference in the small number of data is not obvious. The right side of Fig. 8 shows the experimental results of T10I4D100K. Spark execution time is 106 min, and Hadoop execution time is 405 min. The efficiency increases from 2 to 4 times. The advantages of Spark will become more obvious in large clusters. Experimental results show that the efficiency of association rules mining algorithm proposed in this paper is higher than Hadoop based association rule mining algorithm.

4.2.2 Course recommendation experiment

Fig 9 and Table 3 are the experimental results of HMXPC1. The experiment mines association rules in real course enrollment data, and verifies the feasibility of the algorithm in the course recommendation. Respectively, the experiments were carried out on Spark and Hadoop

Table 3 support

Course combination	support
2	0.76188
4	0.19302
2, 4	0.03248
5, 4	0.0297
3, 2, 1	0.00925
3, 1, 5	0.00725
3, 2, 1, 5	0.00391
3, 1, 5, 4	0.00117

Table 4 confidence

Course rule	Confidence	Recommend efficiency
1,3- > 2	0.4928	26.6%
1,2- > 3	0.4632	25.2%
3- > 1,2	0.1257	Not recommend
2- > 1,3	0.0121	Not recommend
3- > 2	0.4036	24.4%
1- > 3	0.3983	25.3%
2- > 5	0.0213	Not recommend
2- > 1	0.0262	Not recommend

platforms, the program was executed for 53 s on Spark and 124 s on Hadoop platform, furthermore, validating the program on Spark was more efficient than Hadoop. The experiment takes 100,000 data as the training data to analyze the rules between the courses, the remaining 19,024 course enrollment data used for check. The experiments show that the highest support of 1-frequency item set is course 2, indicating that the students of this course has the largest number. The highest support of 2-frequency item set is the course 2 and 4, the highest support of 3-frequency item set is the course 1, 2 and 3, the highest support of 4-frequency item set is the course 1,2,3 and 5.

According to the support of the experimental results, and calculating the confidence between the courses, rule A- > C: Confidence = Support (A, C}) / Support ({A}). The results of analysis are shown in Table 4, it can be seen that the association between courses 1, 3 and 2 is highly, the association between courses 2 and 1 is lowly. According to these data, it is clear that which courses are more likely to be chosen, and make a more reasonable education arrangement. Calculating the rate of course enrollment for the recommended course in the predicted course enrollment data, according to the highest confidence of the course rules. Recommending course 2 to those who enrolls the course 1 and 3, recommending course 3 to those who enrolls the course 1 and 2, and then verify the efficiency of the rules with 19,024 historical data, the first course enrollment rule. The confidence of course 2 after courses 1 and 3 is 0.4928, there are 284 users enroll courses 1 and 3 in the historical dataset, 60 users among them enroll course 2 in subsequent courses, the efficiency of the course recommendation is 26.6%. There are 242 users enroll courses 1 and 2 in the historical dataset, 61 users among them enroll course 3 in subsequent courses, the efficiency of the course recommendation is 25.2%. There are 1149 users enroll courses 3 in the historical dataset, 280 users among them enroll course 2 in subsequent courses, the efficiency of the course recommendation is 24.4%. There are 671 users enroll courses 1 in the historical dataset, 170 users among them enroll course 3 in subsequent courses, the efficiency of the course recommendation is 25.3%. Thus, the MCRS can recommend the appropriate course for some users.

5 Conclusion

With the popularity of MOOC platform, the generation of learning data will be faster and bigger. The feedback for users is also more and more important. MCRS is a course recommendation system that satisfies the demands of today's big data environments, and the basic algorithm is based on Apriori algorithm and Spark calculation model, in terms

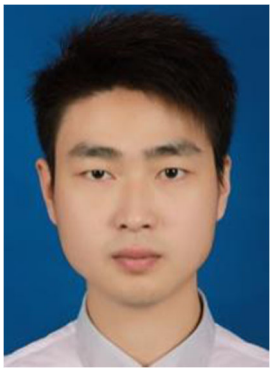
of calculation efficiency and real-time performance compared to Hadoop algorithms, which can make up for its shortcomings, and suitable for big data calculation. As more and more data, the calculation of the advantages will become more obvious, it can flexibly increase the nodes of its calculation to mine association rules on MOOC platform, and according to the information of current users' course enrollment data, recommends the appropriate courses for some users. Based on the MCRS, we can implement different algorithms on the basic data mining algorithms, we can use Spark to implement the algorithms such as genetic algorithm, collaborative filtering, decision tree, etc. A lot of the work of education data mining can be carried out on MOOC platform, not only the course recommendation, but also learning methods, other learning resources recommendation to promote personalized learning [15]. In addition, using advanced technologies such as deep learning to research advanced Education Resource Recommendation Systems. EDM can be used to design better and smarter education system [4], and how to more efficiently recommend resources to users in real time, need to keep exploring.

Acknowledgements This study was funded by the National Programs for Science and Technology Development (grant number 2015BAK07B03), the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD), Jiangsu Collaborative Innovation Center on Atmospheric Environment and Equipment Technology (CICAEET), and specific funding for education science research by self-determined research funds of CCNU from the colleges' basic research and operation of MOE ((grant number CCNU17QN0004)).

References

1. Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems : a survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge & Data Engineering 17(6):734–749
2. Adomavicius G, Zhang J (2012) Stability of recommendation algorithms. ACM Transactions on Information Systems (TOIS) 30(4):23
3. Aher SB, Lobo LMRJ (2013) Combination of machine learning algorithms for recommendation of courses in E-learning system based on historical data. Knowl-Based Syst 51:1–14
4. Baker RS (2014) Educational data mining: an advance for intelligent Systems in Education. IEEE Intell Syst 29(3):78–82
5. Dean BJ, Ghemawat S. (2010) MapReduce: Simpli_ed data processing on large clusters. in OSDI
6. Dean J, Ghemawat S (2008) MapReduce: simplified data processing on large clusters[J]. Commun. ACM 51(1):107–113
7. Denley, Tristan. (2013) Degree compass: a course recommendation system."EDUCAUSE review. Online
8. Gaikwad T, Potey MA. (2013) Personalized course retrieval using literature based method in e-learning system. In Technology for Education (T4E), 2013 I.E. fifth international conference on. 2013: IEEE
9. García E et al (2011) A collaborative educational association rule mining tool. Internet & Higher Education 14(2):77–88
10. Hou, Yifan, et al. (2016) Context-Aware Online Learning for Course Recommendation of MOOC Big Data. arXiv preprint arXiv:1610.03147
11. Kang YQ (2014) An analysis on SPoC: post-MooC era of online education. Tsinghua Journal of Education 35(1):85–93
12. Li N, et al. (2012) Parallel implementation of Apriori algorithm based on MapReduce. In Acis international conference on software Engineering, artificial intelligence, NETWORKING and parallel & distributed computing

13. Lin M, Lee P, Hsueh S. (2012) Apriori-based frequent itemset mining algorithms on MapReduce. in Proceedings of the 6th international conference on ubiquitous information management and communication. 2012: ACM
14. Liu Q, Cai W, Shen J, Zhangjie F, Liu X, Linge N (2016) A speculative approach to spatial-temporal efficiency with multi-objective optimization in a heterogeneous cloud environment. Security and Communication Networks 9(17):4002–4012
15. Luo Dan, Yunxiang Zheng. (2015) The Research of Online Teaching Pattern Based on MOOC. 2015 International symposium on educational technology (ISET). IEEE
16. Rendle S (2012) Factorization machines with libfm. ACM Transactions on Intelligent Systems and Technology (TIST) 3(3):57
17. Romero C, Ventura S (2010) Educational data mining: a review of the state of the art. IEEE Transactions on Systems Man & Cybernetics Part C Applications & Reviews 40(6):601–618
18. Salehi M, Kamalabadi IN, Ghoushchi MBG (2013) An effective recommendation framework for personal learning environments using a learner preference tree and a GA. IEEE Trans Learn Technol 6(4):350–363
19. Sarwar B, et al. (2001) Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on world wide web. 2001: ACM
20. Tanai Mirwais, Jongwan Kim, Joong Hyuk Chang. (2010) Data Mining and Educational Decisions. (2010): 260–263.
21. Wang Y, Tseng MH, Liao HC (2009) Data mining for adaptive learning sequence in English language instruction. Expert Syst Appl 36(4):7681–7686
22. Wang Y, Tseng MH, Liao HC (2009) Data mining for adaptive learning sequence in English language instruction. J Expert Syst Appl 36(4):7681–7686
23. Wassan JT (2015) Discovering big data modelling for educational world. Procedia - Social and Behavioral Sciences 176:642–649
24. Wen M, Rose CP. (2014) Identifying latent study habits by mining learner behavior patterns in massive open online courses. in Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management
25. West DM, (2012) Big Data for Education: Data Mining, Data Analytics, and Web Dashboards. 2012: Hrvatska znanstvena bibliografija i MZOS-Svibor 11
26. Yang X, Guo Y, Liu Y et al (2014) A survey of collaborative filtering based social recommender systems. J. Comput Commun 41:1–10
27. Zhangjie F, Sun X, Liu Q, Lu Z, Shu J (2015) Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing. IEICE Trans Commun E98-B(1):190–200
28. Zhou Q (2015) M.C.Y.D., research progress on educational data mining: a survey. Ruan Jian Xue Bao/ Journal of Software 26(11):3026–3042 http://www.jos.org.cn/1000-9825/4887.htm



Hao Zhang is Assistant Professor of Central China Normal University. His main researches are multimedia resources recommendation, the privacy of big data, security of cloud computing based on virtualization and learning behavior analysis based on machine learning and Deep Learning.

Tao Huang is associate professor of Central China Normal University. His main research is multimedia resources recommendation system, cloud computing, and online learning technology.

Zhihan Lv is an engineer and researcher of virtual/augmented reality and multimedia major in mathematics and computer science, having plenty of work experience on virtual reality and augmented reality projects, engage in application of computer visualization and computer vision.

Sanya Liu is a professor and deputy director in NERCEL, CCNU. His research interests include artificial intelligence, computer application and educational data mining and quantified learning technologies.

Zhili Zhou is an Assistant Professor with the Nanjing University of Information Science and Technology. His current research interests include near-duplicate image/video detection, image/video copy detection, coverless information hiding, digital forensics, and image processing.