Rare itemsets mining algorithm based on RP-Tree and Spark framework

Sainan Liu[a], Haoan Pan[b]

Chongqing Key Laboratory of Computational and Intelligence Chongqing University of Posts and Telecommunications Chongqing,China

[a]QUeensainan@163.com
[b]675445550@qq.com

Abstract. For the issues of the rare itemsets mining in big data, this paper proposed a rare itemsets mining algorithm based on RP-Tree and Spark framework. Firstly, it arranged the data vertically according to the transaction identifier, in order to solve the defects of scan the entire data set, the vertical datasets are divided into frequent vertical datasets and rare vertical datasets. Then, it adopted the RP-Tree algorithm to construct the frequent pattern tree that contains rare items and generate rare 1-itemsets. After that, it calculated the support of the itemsets by scanning the two vertical data sets, finally, it used the iterative process to generate rare itemsets. The experimental show that the algorithm can effectively excavate rare itemsets and have great superiority in execution time.

Key words: Component; Rare Item; Spark framework; RP-Tree algorithm; Vertical layout.

INTRODUCTION

Association rule mining techniques are used to extract useful information from databases. The set of association rules that can be extracted from a database can be divided using a support threshold into frequent and rare association rules. In many domains, events that occur frequently may be less interesting than events that occur rarely. For example, in the area of medicine, the expected, frequent responses to medications are less interesting than rare responses which may indicate unusual side-effects. This is also true in other domains, such as the detection of fraudulent financial transactions and network intrusions 1.

The earliest algorithm to solve the problem of rare items is MS-Apriori 2. It uses a bottom-up approach similar to Apriori 3. However, the value of multiple minimum support is too high. MS-Apriori is not able to identify rules which have low support and high confidence. Troiano et al.4 Proposed the Rarity algorithm that begins by identifying the longest transaction within the database and uses them to perform a top-down search for rare itemsets, thereby avoiding the lower layers that only contain frequent itemsets. Apriori-Inverse 5proposed by Koh et al. is used to mine perfectly rare itemsets. Sidney Tsang et al 6 proposed RP-Tree algorithm. RP-Tree avoids the expensive item set generation and pruning steps by using a tree data structure, based on FP-Tree 7, to find rare patterns. RP-Tree is based on FP-Growth 8, which is efficient at finding long patterns, since the task is divided into a series of searches for short patterns. So, RP-Tree algorithm is an improvement over these existing algorithms.

With the explosive growth of data, Big data requires strong resources for storage and processing,

Distributed computing is an effective way to solve large data. Spark is a better distributed computing framework, The programming interface is based on resilient distributed datasets9. It can effectively solve the problem of batch processing and interactive processing. At present, research on frequent itemsets mining based on Spark framework has been developed 10 11.

Based on the above analysis, In this paper, we propose a rare pattern mining algorithm based on RP-Tree and implement it on the Spark framework. The algorithm adopts the idea of vertical datasets and divides the datasets into frequent vertical datasets and rare vertical datasets, which are used to calculate the support degree of different itemsets. At the same time, in order to improve the efficiency of the algorithm, delete the combination that does not contain rare items, and do not duplicate the itemsets that have been stored in the rare vertical datasets. Thus, the number of candidate itemsets is reduced. The experimental show that the algorithm have great superiority in execution time.

RELATED TECHNOLOGY

Basic concept

Rare itemsets mining is from a given data set, through the support of the itemsets compared with the threshold to mine the rare itemsets. Among them, the itemset's support is the total number of all transactions in the dataset that contains the item set. An item is considered a rare item if its support is between minRareSup and minFreSup 6. The min Rare Sup threshold is a noise filter, whereby items that are below this threshold are considered as noise. Rare itemsets can be divided into types: rare-item itemsets which refer to itemsets that consist of only rare items and itemsets that consist of both rare and frequent items.

Rare itemsets mining

RP-Tree is an efficient algorithm for mining rare itemsets from transaction set. The core idea is to convert the database to a frequent pattern tree containing rare items. Then, Mining the rare items in this tree. RP-Tree algorithm is divided into two steps:

(1)Build frequent pattern tree that contain rare items: First, scan the transactional database, delete things that do not contain rare item, and arrange in descending order. Then, create the root node of the tree and mark it as null. For each transaction in the database, the items in the transaction are ordered first, then recursively inserted in the schema tree.

(2)Rare Itemsets Mining: based on the frequent pattern trees that are created that contain rare items.

PROPOSED RARE ITEMSETS MINING METHOD

On Spark distributed computing framework, a rare itemsets mining algorithm is proposed based on RP-Tree and dataset vertical layout mechanism, Call it RP-Tree Vertical layout Rare Item set Mining, RP-VRIM. RP-VFIM uses the concept of vertical layout of datasets, which consist of lists of itemsets and their Transaction Identifiers (TIDs), including frequent vertical datasets and rare vertical datasets. In each record of vertical data, the TIDs are sorted in ascending order. The proposed RP-VRIM algorithm can be divided into two major steps, the overall process shown in Figure 1, the specific steps described below.

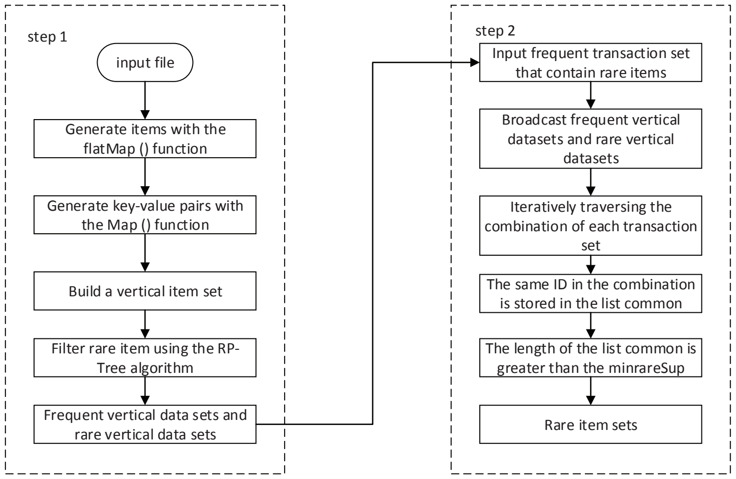
step 1
input file
Generate items with the flatMap () function
Generate key-value pairs with the Map () function
Build a vertical item set
Filter rare item using the RP-Tree algorithm
Frequent vertical data sets and rare vertical data sets
step 2
Input frequent transaction set that contain rare items
Broadcast frequent vertical datasets and rare vertical datasets
Iteratively traversing the combination of each transaction set
The same ID in the combination is stored in the list common
The length of the list common is greater than the minrareSup
Rare item sets

FIGURE 1. RP-VFIM algorithm of the basic process

Step 1

In step 1, a vertical dataset is generated in a vertical layout. Then, it is divided into frequent vertical data sets and rare vertical data sets according to the category of the item. Among them, frequent vertical datasets contain only frequent items, and rare vertical datasets contain rare items and subsequently updated itemsets.

This article uses the flatMap () function to generate all the items from the dataset. Then, the Map () function is applied to each item to generate key-value pairs, in which the key is an item and the value is a list of TID. The group ByKey () function is used to aggregate the items. In order to reduce the time required to traverse the itemsets in subsequent operations, the aggregated vertical datasets are classified as follows: If the number of TIDs contained in the item is greater than the minFreSup, it is classified into frequent vertical datasets. If the number of TIDs contained in this item is between the minFreSup and the minRareSup, it is classified as a rare vertical datasets. If the number of TIDs contained is less than the minRareSup, then it is discarded. Algorithm 2 is the pseudo code of step 1 process.

```
Input:  D=Transaction set
Output: vertical_rare_data= Rare vertical itemsets,
vertical_fre_data= Frequent vertical itemsets
1.  for each t in D//t: transaction
2.    flapMap(trans_ID,t);
3.      for each item I in t
4.        R1=map(I,trans_ID);
5.      end for
6.    end flapMap
7.  end for
8.  RP tree()
9.  R2=R1.groupByKey();
10. for each item m in R2
11.     if(length(m.getvalue()≥min_rare_sup∧
length(m.getvalue()≤min_fre_sup)
12.   vertical_rare_data= vertical_rare_data∧m
13.   else if
14.   length(m.getvalue()≥min_fre_sup
15.   vertical_fre_data= vertical_fre_data∧m
16.   else
17.     delete m
18.   end if
19. end for
20. Output(vertical_rare_data, vertical_fre_data);
```

Step 2

Step 2 of RP-VFIM is used to generate rare K-itemsets, where K≥2. This step is an iterative process, It will generate K-rare itemsets in the Kth iteration. This article uses frequent vertical data sets and rare vertical data sets to calculate the support for each candidate.Create broadcast variables vertical_rare_data and vertical_fre_data and assign the vertical dataset's RDD. Scan this shared RDD and use it to calculate the support for each candidate set. Traversing the frequent vertical datasets and the rare vertical datasets in step 1, to find the transaction set that appear in the candidate set. All the same transactions in the combination are stored in the list common. The count of the TID, that is, the length of the common, is the support of the candidate set. If the same list of common length greater than or equal to the minRareSup, then the item set is to be mining rare items. And, all itemsets are loaded into Rare_itemset, Rare_itemset is a pair of RDDs, and algorithm 3 is the pseudocode of Step 2.

Algorithm 3: Step 2

```
Input: D_Fre= Frequent transaction sets that contain rare items, vertical_rare_data, vertical_fre_data, minRareSup, minFreSup, k:The number of recursions
Output: Rare_itemset
1.  rare_list= vertical_rare_data.getkey()
2.  for each t in D_Fre
3.    map(trans_ID,t)
4.  end for
5.  shared_fre_data=broadcast(vertical_rare_data), shared_fre_data=broadcast(vertical_fre_data);
6.  for evey item_list in t
7.    combinations= combiner(item_list,k);
8.  for item set IS in combinations
9.    if item∈fre_list
10.     search shared_fre_data item set IS get list_of_transIDS;
11.   else
12.     search shared_rare_data item set IS get list_of_transIDS;
13.   common=same list_of_transIDS;
14.   if(length(common)≥minRareSup)
15.     Rare_itemset=Rare_itemset∪list(C,common);
16.   end if
17. end for
```

According the new Rare_itemset to update shared_rare_data;

EXPERIMENTS AND ANALYSIS

Experimental design

Build a distributed environment, set a 5-node Spark cluster, Configuration Intel Core i5-4210 CPU, 8GB memory nodes. Connect all nodes with a 100Mbps Ethernet converter. The proposed RP-VRIM algorithm is implemented in Python compilation language. This article uses the T1014D100k and T40I10D100k two data sets.

TABLE 1. Attributes of data sets

data set	The number of items	The number of transactions
T1014D100k	870	10000
T40I10D100k	942	10000

Performance Analysis

First, analyze the impact of different min Rare Sup on algorithm performance. The threshold for setting the min Rare Sup of all transactions varies from 0.5% to 4%, The threshold of min Fre Sup is 10%, and the number of computing nodes is 5. The result is shown in Figure 2. It can be seen that the execution time of the algorithm decreases with the increase of the min Rare Sup threshold. This is because with the increase of the min Rare Sup threshold, the rare itemsets are mined less. In addition, the algorithm execution time decreases faster when the min RareSup is in the range of 0.5% to 1.5%.

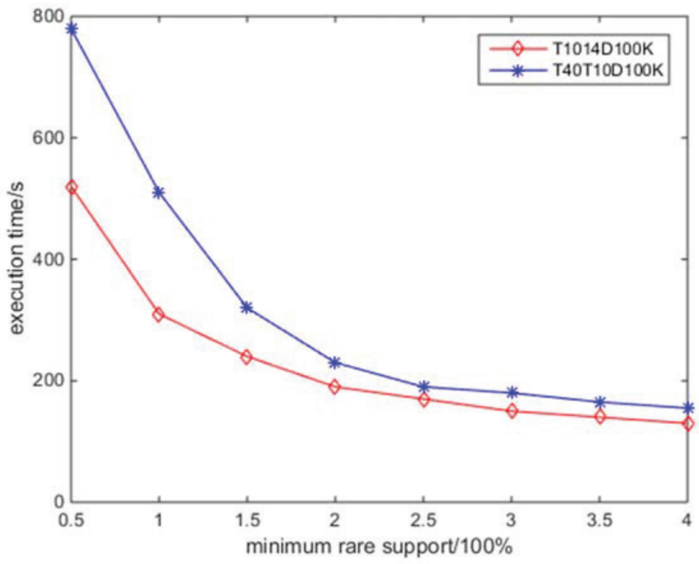

FIGURE 2. Different minRareSup

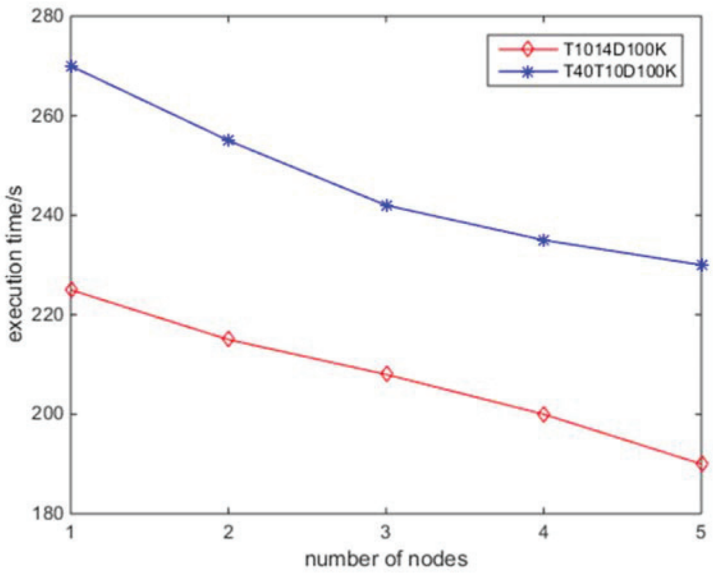

FIGURE 3. The number of different cluster nodes

Then, to verify the extensibility of the RP-VRIM method, Set the number of computing nodes in the Spark cluster from 1~5, The minRareSup is set to 2%. The result is shown in Figure 3. It can be seen that as the number of compute nodes in the cluster increases, the total execution time of the algorithm decreases and almost linearly.

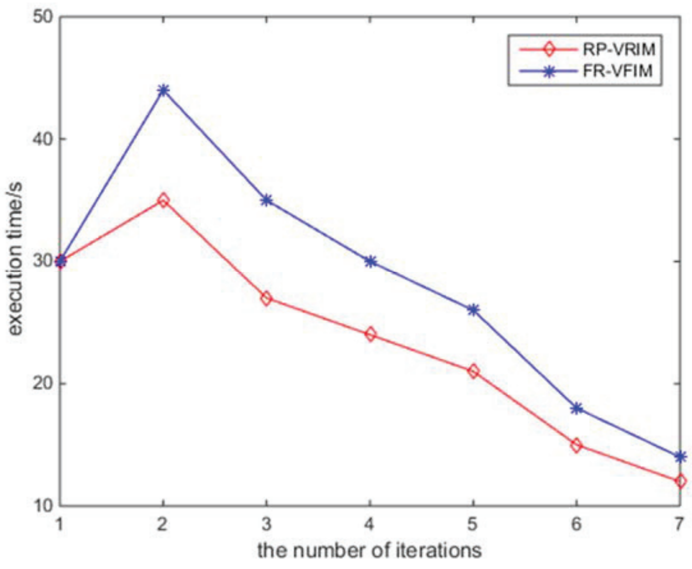

FIGURE 4. T1014D100k data set

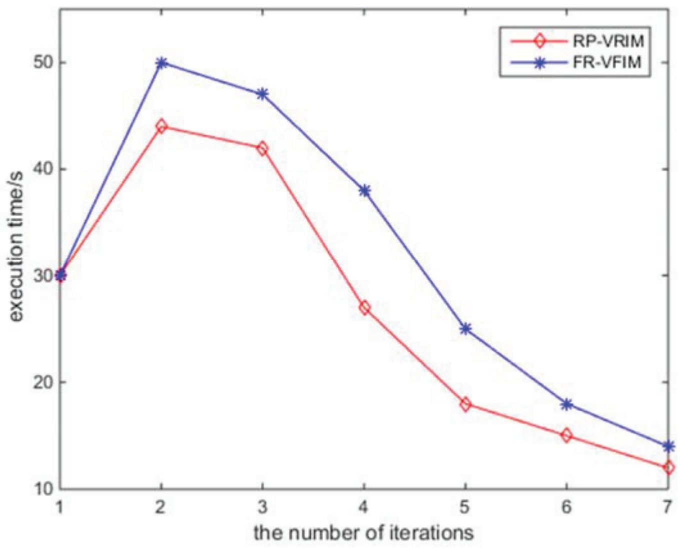

FIGURE 5. T40I10D100k data set

Comparison of execution time between FP-VFIM algorithm11 and RP-VFIM algorithm based on Spark framework. The number of nodes in the cluster is five. Among them, two thresholds are set for the FP-VFIM algorithm, which is the same as that of the RP-VFIM algorithm. The experimental results Figure 4 and Figure 5 can be seen, The execution speed of RP-VRIM on the data set is faster than that of FP-VFIM. The two algorithms are almost the same in the execution of the iteration 1.Due to the vertical data set used in RP-VFIM, the vertical datasets are divided into frequent vertical datasets and rare vertical datasets. Reduced traversal combination of the use of time, so the execution time is less.

CONCLUDING REMARKS

The RP-Tree algorithm is one of the efficient algorithms for mining rare itemsets from a transactional set. This frequent pattern is further used for association rules mining and has a wide range of applications. For the issues of the rare itemsets mining in big data, need a lot of memory and resource processing. Therefore, this paper proposes a RP-Tree Rare Itemsets Mining Algorithm on the distributed computing Spark framework. The algorithm uses frequent vertical datasets and rare vertical datasets to solve over-scan problems. Experiments show that this method and the method of not splitting vertical datasets have advantages in execution time.

REFERENCES

1. Adda M, Wu L, Feng Y. Rare Itemset Mining [C]// International Conference on Machine Learning and Applications. IEEE, 2007:73-80.
2. Liu B. Mining association rules with multiple minimum supports [C]// Knowledge Discovery and Data Mining. 1999:337-341.
3. Dandu S, Deekshatulu B L, Chandra P. Improved Algorithm for Frequent Itemsets Mining Based on Apriori and FP-Tree [J]. Global Journal of Computer Science & Technology, 2013, 13 (2).
4. Troiano L, Scibelli G, Birtolo C. A Fast Algorithm for Mining Rare Itemsets [C]// International Conference on Intelligent Systems Design & Applications. IEEE Computer Society, 2009:1149-1155.
5. Yun S K, Rountree N. Finding Sporadic Rules Using Apriori-Inverse [C]// Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining. Springer-Verlag, 2005:97-106.
6. Tsang S, Yun S K, Dobbie G. Finding Interesting Rare Association Rules Using Rare Pattern Tree[M]// Transactions on Large-Scale Data- and Knowledge-Centered Systems VIII. Springer Berlin Heidelberg, 2013:157-173.
7. Dongmei Fu, Zhiqiang Wang. Mining algorithm of association rule based on FP-tree and constrained concept lattice and application research [J]. Application Research of Computers, 2014, 31 (4):1013-1015.
8. Karthikeyan T, Vembandasamy K, Raghavan B. An Intelligent Type-II Diabetes Mellitus Diagnosis Approach using Improved FP-growth with Hybrid Classifier Based Arm [J]. Research Journal of Applied Sciences Engineering & Technology, 2015, 11(5):549-558.
9. Chen Bian, Tong Yu, Changtian Ying,. Self-Adaptive Strategy for Cache Management in Spark [J]. Acta Electronica Sinica, 2017, 45(2):278-284.
10. Lukui Shi, Xin Zhang, Shengli Shi. Parallelization and optimization of FP_Growth algorithm based on Spark [J]. Computer Engineering and Applications, 2017.
11. Shao Liang,He Xingzhou,Shang Junna. Frequent itemsets mining algorithm for big data based on FP-Growth and Spark framework [J/OL]. Application Research of Computers, 2018, (10):1-6(2017-10-10).