International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-3 Number-1 Issue-9 March-2013

A Survey of latest Algorithms for Frequent Itemset Mining in Data Stream

U.Chandrasekhar[1], Sandeep Kumar. K[2], Yakkala Uma Mahesh[3]

Abstract

Association rule mining and finding frequent patterns in data base has been a very old topic. With the advent of Big Data, the need for stream mining has increased. Hence the paper surveys various latest frequent pattern mining algorithms on data streams to understand various problems to be solved, their short comings and advantages over others.

Keywords

Data Mining, Frequent Pattern Mining, Data Streams.

1. Introduction

In a data mining process, it is a tedious task to find the frequent patterns that are extracted from the Data Warehouse. Patterns are the set of rules behind the Mining frequent pattern of data, which comprises of Frequent Item Set Mining, Frequent Sequence Mining, Frequent Tree Mining, and Frequent Graph Mining. From that pattern, extracting knowledge is tedious. To propose this problem, so many algorithms have come up. Now days, the data sizes are in about Peta bytes. Hence it is hard to precede the problem whenever a worst case occurs. Selecting the suitable algorithm is a biggest task ever. In order to extract the accurate knowledge, the data should not be damaged. Each algorithm faces the trouble with the data collection and processing in the different view. Every algorithm is not suitable for the all kind of itemsets. This paper presents few latest algorithms used in various scenarios.

2. Illustrations and Results of Different Algorithms

A. An Algorithm for Mining Frequent Items on Data Stream Using Fading Factor:

In 2009, Ling Chen et al. [1] proposed that the fading factor model can be used to compute the frequent itemsets. The fading factor lm contributes more to the recent items than the older. It ranges between 0<lm<1, where lm is frequency. The value near to 1 is considered to be most frequent item. It has two major advantages. It takes the all old data items based on the frequency and the other is, changes in frequency vary by a small values. It uses Frequent-Item Counting algorithm with fading factor for finding out the e-approximate frequent data items occurrences with only $O(K+e^{-1})$ space with K as constant. Consider the input stream data as A= (a1, a2, a3…..), then its density of A at tm time is D(a, tm).

With the various conditions of the current time data and last time data. The density gets increased only when the same data gets received with the various proven formulae. Similar to the Lossy Counting, this also provides the results like

1. Items with the density greater than
   R / 1 – l,
   will be the output.
2. Items of density less than
   S – E / 1 - l
   is discarded or faded away.
3. The complete density is always greater than the estimated by
   E / 1 – l.

Here this algorithm uses list data structure. It holds the information about last data item and its density. The data items are mapped to the HASH function. Based on the density value the item gets added or removed from the hash table.

Framework of FC (Frequent Item Counting)

Input: A: the data stream.
lm: fading factor
e: density error bound
S: density threshold.
a: data item.
Output: Potential frequent data items.

Start
1. tm=0, $K=e^{-1}$
2. Loop if not terminated continue

Sandeep Kumar. K, MCA, VIT University, Vellore, India.
Yakkala Uma Mahesh, MCA, VIT University, Vellore, India.
U.Chandrasekhar, Assistant Professor-Senior, SITE School, VIT University, Vellore, India.

3. If new data item a has no entry in hash table then
4. Allocate new entry for a;
5. If density of the data item is less than the threshold value then
6. Delete that entry;
7. Replace that entry with the previous one;
8. End if
9. Insert a new entry to the end;
10. Else
11. Replace corresponding entry with the new density value;
12. End if
13. tm=tm+1;
14. End loop

End

This FC algorithm required only $O(K+e^{-1})$ memory space. It's not involved with any multiplication of density values with the fading factor. It just modify the new value whenever the similar data item comes. This algorithm is tested with comparing the loss counting in synthetic data set and real time data set. In both the data sets the time and the space required is very less for this algorithm when compared to LC.

B. An Efficient Algorithm for Mining Frequent Patterns over High Speed Data Streams:

In 2009, Cai-xia Meng [2] proposed the efficient algorithm for mining frequent itemsets over a high speed data streams. All the frequent pattern mining algorithms pose two steps. It involves calculations behind the arrival of every frequency of new item sets and formatting them into the output. In this algorithm, these two steps are blended together to reduce too much of time that arrives in LossyCounting (LC) and FDPM. The loss of data occurs in other algorithms is sorted out here. It uses DeferCounting (DC) method which delays the frequency calculation and provides gap between step 1 and step 2 to avoid the transaction missing problem. In step 1, DC includes the information that achieved the threshold value. In step 2, it frequent pattern from the stored information. Here the data structures used are List and Trie. List is for the frequent items and Trie for the frequent item sets. List is used with three-fields stating, id of each item, frequency of that item and error between the actual and the estimated frequency. Trie composed of two fields in which one points to counter in the list. The other field is to compute the frequency of the itemsets.

The DC algorithm:

Start

1. Fix the list length;
2. For every transaction x in R
3. Update the list with transaction x;
4. Extract only the items that are in the list
5. Insert the transaction x in Trie;
6. If query about frequent-patterns then
7. Return result with growth of Trie and support p.
8. End if
9. End for

End

This paper compared the DC algorithm with the LossyCounting and FDPM. Comparing with the LossyCounting (LC) algorithm, it takes more time in the step 1 and occupies very less memory. It supports well for the increase in the transaction length than LC. Comparing with FDPM, DC takes less time in the step 1 for the same item sets. But the space complexity of the FDPM is not mentioned clearly about calculating the frequency of item sets. These algorithms are compared practically with the real time data. All comparison proved that DeferCounting is very efficient on the LossyCounting and FDPM.

C. Kaal – a Real Time Stream Mining Algorithm:

In 2010, Varun Kumar et al. [3] proposed this algorithm which has an ability to hold the various sizes of the batch rather than the fixed in other. But the time has been fixed for segregating the Batches. In the previous algorithms, the infrequent items get removed. If those items become frequent later, the data cannot be bagged again. Moreover they concentrated only on the frequent item sets, but not on the extracting knowledge from it. These kinds of problems are clearly solved by this paper. It uses an extension of trie structure with the log-time window as its data structure. It constitute three columns namely tilted-time, frequency and size of the batch. The recent data holds big space whereas the old one holds the less only. It follows two different types of tail pruning in examining whether the superset needs to be dropped or not based on the different batch sizes and time.

This algorithm involves three major steps

1. Initializing the maximum allotted execution time.
2. Initializing the support value for the starting phase.
3. Initializing the trie data structure to empty.

Get all the transactions arrived in a fixed period of time.

I. Very recent batch processing
   1. The candidate patterns in the search frontier are made as global pool.
   2. Make border set to null.
   3. Arrange the patterns in descending order of length.
   4. Take the best pattern and arrange it in a block bk.
   5. Expand the block bk till the last level.

II. Modification of the summarized data/structure.
   1. Mine the new itemsets
   2. Check which type of pruning is suitable.
   3. Perform pruning.

III. Scanning the trie.
   1. Check if all itemsets are updated.
   2. If not update it.
   3. Get the new batches and continue the same.

The processing rate for all the various sized batches took around same which provides the major advantage than any other algorithms. The preprocessing steps work well to differentiate well. It starts the batch processing as soon as it starts rather than delaying like FP-Stream. It follows the heuristics model during the less support requirements to give frequencies.

D. A Linear Regression-Based Frequent Item set Forecast Algorithm for Stream Data:

In 2009, Sonali Shukla et al. [4] proposed this algorithm with the regression based methodology to find out the frequent item sets continuously that are streaming regularly. The 2-Dimensional stream data is preprocessed and converted into sampling value. With these values, regression analysis is carried out. It bags the data using sliding window model and then it applies FIM-2DS algorithm to compute with the item set.

Then it is processed to the sampling value for the further process with least square method. Each data is paired (mi, ni) to find the arrival time difference between them. It is calculated like t, t-1, t-2, t-3… t-n. if the pair is (m,n) then it is mean that 'm' is an independent variable of 'n' and 'n' is dependent variable on 'm'. With the help of the pairs of Data Sets, the dependent and independent variable values are calculated. Later the regression line is drawn from fragment slope values. Moreover, the regression analysis is also used to find out the functional relationships between the paired data items. After finding out the regression model the following steps are followed.

Algorithm:

Step-1: The real time stream data, threshold value and window size is given as input.

Step-2: The support value is calculated which is greater than threshold.

Step-3: Consider time and support as independent and dependent variable respectively.

Step-4: The co-efficient of the regression model is calculated.

Step-5: Find the slope value to draw a regression fit line.

Step-6: Finally calculate the error.

Here the error found is less than FTP-DS algorithm. The less amount of error has a major impact on the frequent itemsets mining. But this algorithm doesn't deal with the noise removal present in the huge and fast changing data and also about the execution time.

E. A More Accurate Space Saving Algorithm for Finding the Frequent Items:

In 2010, ZHOU Jun et al. [5] proposed this algorithm by considering the space as an important factor. They used an improved LRU (Least Recently Used) based algorithm. This algorithm omits the infrequent items before taken for the processing. It increases the stability and the performance. This is used to find out the frequent items as well as the frequency of those items. It has 6 properties to follow the algorithm.

Algorithm:

1. Initialize the value of data items monitored;
2. Initialize LRU value and its length;
3. Initialize the stream summary value and bucket values to 0;
4. Let K be set of data item to be monitored over current transaction;
5. If Stream- Summary monitors K then Counter of K is incremented and steps into higher bucket;
6. Else if LRU monitors K then Consider the L is a set monitored by stream-summary.
7. Replace the L with K;
8. Remove K from LRU;

9. K counter is incremented and step into higher bucket;
10. Else If the LRU is overflow then
11. Delete the last item in it;
12. Insert the K at the head;
13. End if.

This algorithm mostly concentrates on the space. But the time taking for replacing the counter value and the changing of bucket levels results in extra time. Since it is hashing-based algorithm drop has made at the tail and the insert is made at the head of the LRU. Its complexity is remains *O(l)* where l is the length of the data stream. Here the efficiency is measured in space complexity. It occupies less memory space than any other algorithms. This is suitable for net bone packet stream and high speed network packet stream. It mostly applicable only on the network monitoring and it's not suitable on applying it in the real time streaming data.

F. An Algorithm for Predicting Frequent Patterns over Data Streams Based on Associated Matrix:

In 2012, Yong-gong Ren et al. [6] proposed this algorithm in order to predict the future data based on the new method called AMFP-Stream (Associated Matrix Frequent Pattern-Stream), it predicts the frequently occurred item sets over data streams efficiently. It also has a capability to predict that which item set will be frequent with high potential. It takes the data in the form of 0-1 matrix and then it updates the values by doing logical bit operations. Based on this it will find out the item sets that will frequently occur in the future. It uses the associated matrix for the further manipulation. Experimental results says that this algorithm is how much feasible.
Predicting the frequent itemsets in this algorithm involves three steps.

1. Use the transactional data and generate the associated matrix that is dogged out.
2. Update and delete the associated matrix for every new frequent itemsets.
3. Predicting the frequently occurred data in the associated matrix based on the current window data.

At first associated matrix $Mij(1<i,j<n)$ needs to be generated. If the itemsets are repeated, then the value change from 0-1 in the matrix. For each set, it determines the existence. At each time the first column and the last row of the matrix holds the number of occurrences of the itemsets in the particular row and column.

It involves four different algorithms. The first algorithm discuss about the generation of the associated matrix by giving the transaction dataset as an input.

First Algorithm:
Step-1: Store the incoming data in a separate memory;
Step-2: Initialize the Mij to be 0;
Step-3: Evaluate each value;
Step-4: Loop verify till the last element
Step-5: If itemsets are repeated then
Step-6: Record the statistical information;
Step-7: Store the support number;
Step-8: End If.
Step-9: End loop.

The next algorithm involves in updating the associated matrix values with frequent item sets. It removes the item sets that are infrequently occurred.

Second algorithm:
Step-1: Consider all the items.
Step-2: Check for the minimum support
Step-3: If the transaction value not repeated then
Step-4: Reduce the frequency of that item.
Step-5: Delete the items which are infrequent.

Now the matrix holds the items that are only of the frequent items. Now it should be mined with the help of another algorithm. It involves in extracting the frequent item sets from it for the gain of knowledge. It takes the last updated associated matrix as an input and produces the frequent itemsets as the result.

Third Algorithm:
Step-1: Consider all the elements from the associated matrix.
Step-2: If the occurrences is 0 then do nothing
Step-3: Else itemsets and supporting data are determined.
Step-4: End if
Step-5: If more items then continue from step-1 again.
Step-6: Else exit the routine after updating the frequent itemsets.

Based on the values from the associated matrix, the itemsets may occur frequently by the previous window values in the current transaction. These problems are get solved by removing the very old transactions before the new transaction takes places and updated with the new frequency of the itemsets. The last fourth algorithm predicts the itemsets that will occur frequently occur in the future. It takes the current sliding window as an input and produces the

next sliding with maximum possible frequent itemsets.

Fourth Algorithm:

Step-1: Consider the data of the current sliding window.

Step-2: Consider the already exiting frequently occurred itemsets.

Step-3: Find out the common itemsets.

Step-4: This results in the itemsets frequency measure in upcoming sliding window.

From these algorithms we can conclude that the different ways are followed to predict the frequent itemsets. But it fails to explain about the space and time complexity even though it poses full accuracy in the frequent itemsets. We can conclude that these algorithms are suitable for mining the any small size databases.

G. A New Adaptive Algorithm for Frequent Pattern Mining over Data Streams:

In 2011, Mahmood Deypir et al. proposed this algorithm based on the different kind of sliding window based model. It don't need entire data that are in streaming. It takes an advantage of the already existing item sets. To enhance the feature of sliding window concept, it reduces the amount of space occupying and time taken to calculate based on the fixed size of the window.

This paper introduces the pane based algorithm, which is names as paWin has a good result over the execution time and memory occupying. It adds the features of the transaction window-based algorithms like estWin. Here, all the new newly arriving data are made up to the fixed sized panes. All the panes holds the information about the transaction identification number sets. As soon as the each pane arrives the itemsets are mined using Eclat algorithm. This itemsets are stored in the form of prefix tree structure. Each node in a tree holds the various information like item's id, actual pane window counts, id of the pane, potential count and the pointers to hold the children. The next phase is window sliding in which panes addition and deletion takes place. The new itemsets get updated from the new pane. Each of the new *Tridsets* are added to the child node. Based on the parent node consideration the new node is inserted. The depth of the tree is considered as a major to find out the frequency of the items. The impact of the minimum support threshold is vary for all the algorithms. The performance is measured for the algorithms like estWin and SWIM algorithms and conclude that the proposed algorithm's performance is much better. With the help of the batch processing transactions the high speed data streams are processed in a short period of time which facilitates the performance of this algorithm.

3. Performance Analysis

Table 1: Comparison Chart

Alg.	Benefits	Drawbacks
A	Faster than λ-LC	Not mentioned about size factor
B	Provides good Support rate for long data streams.	More space but valuable.
C	Considers old frequent data.	Approximate frequency on low support.
D	Error optimized	Loss in support on Huge Stream data
E	Less Memory and Good for online processing high-speed network packet data.	Does not support for the removal of data.
F	High support rates and Good Accuracy	Not mentioned about time and space complexities.
G	More Accuracy, Less memory and Run-time.	Removing old infrequent items may come frequent later.

4. Conclusion

In algorithm [1] the data is processed at $O(K+\varepsilon^{-1})$ memory where M is a constant and the time taken to process $O(m)$. The next algorithm [2] takes a DeferCounting in an effective manner. It takes more time in calculating the frequency in step 2. Since the first step allows taking only important information from every transaction the throughput is higher. The next algorithm [3] has includes the entire frequent pattern from the beginning itself rather than skipping it like in other. The algorithm [4] is not analyzed yet in the factors like huge data, fast changing with the noisy and incomplete data. The real time data streaming is suited well in this algorithm [5]. The algorithm [6] is used to predict not only the current window's frequent pattern but also it predicts the next. The algorithm [7] facilitates in adding the new frequent items and in removing the old items in the periodic manner. But this algorithm might fail if there

is a necessary usage in the older frequent data.

Considering the accuracy, space and execution time as factor, the algorithms are developed in a way that can give the frequent patterns. By comparing these algorithms, we will be able to select which kind of algorithm is suitable for the given scenario. The varying size of the database might also predict the favoring algorithm. The methodology followed is same, but throughput is varying for each. Still now no algorithm has proven that the knowledge gained from the available data is more accurate than others.

References

[1] Ling Chen, Shan Zhang,Li Tu, "An Algorithm for Mining Frequent Items on Data Stream Using Fading Factor".33rd Annual IEEE International Computer Software and Applications Conference.172-179,2009.

[2] Cai-xia Meng, An Efficient Algorithm for Mining Frequent Patterns over High Speed Data Streams. World Congress on Software Engineering, IEEE 2009, 319-323.

[3] Varun Kumar,Rajanish Dass. Proceedings of the 43rd Hawaii International Conference on System Sciences, 2010 IEEE, 978-0-7695-3869-3.

[4] Sonali Shukla, Sushil Kumar, Bhupendra Verma,A Linear Regression-Based Frequent Itemset Forecast Algorithm for Stream Data. International Conference on Methods and Models in Computer Science, 2009.

[5] ZHOU Jun, CHEN Ming, XIONG Huan A More Accurate Space Saving Algorithm for Finding the Frequent Items.IEEE-2010.

[6] Yong-gong Ren,Zhi-dong Hu,Jian Wang. An Algorithm for Predicting Frequent Patterns over Data Streams Based on Associated Matrix. Ninth Web Information Systems and Applications Conference, 2012. 95-98.

[7] Mahmood Deypir, Mohammad Hadi Sadreddini, A New Adaptive Algorithm for Frequent Pattern Mining over Data Streams, ICCKE,2011, 230-235 FLEX Chip Signal Processor (MC68175/D), Motorola, 1996.

Sandeep Kumar. K, born in Bengaluru on August 8th, 1989. Completed UG and proceeding MCA in VIT University, Vellore.

Yakkala Uma Mahesh, born in Dachetalli, Hyderabd. on June 6th 1988. Completed UG and proceeding MCA in VIT University, Vellore.