International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 02 Issue: 08 | Nov-2015 www.irjet.net p-ISSN: 2395-0072

Extended R-Tree Indexing Structure for Ensemble Stream Data Classification

P. Sravanthi
M.Tech Student, Department of CSE
KMM Institute of Technology and Sciences
Tirupati, India

J. S. Ananda Kumar
Assistant Professor, Department of CSE
KMM Institute of Technology and Sciences
Tirupati, India

Abstract
Stream data mining is an important data mining technique. Many stream data mining algorithms need only one pass to process data sets. More sophisticated state-of-the-art indexing techniques are needed to process, search, query the huge stream data. Ensemble stream data management technique is one of the stream data mining techniques. Ensemble-tree is an indexing data structure for storing classification rules of ensemble classifiers. Ensemble tree reduces time complexity of classifying a new stream record from linear to log. A new method is proposed for node splitting in the ensemble indexing tree construction. The new node splitting method is simple and easy to compute measures.

1. INTRODUCTION
In stream data mining, many data mining algorithms need many passes to process, compute and mine Data sets. But many real life problems need only one pass to mine large volumes of data streams. Fast, efficient, effective, correct, productive, stream data processing is a new, popular, latest research area. The system called datatStream is a state of the art system proposed for monitoring a set of large data streams. Credit Card transactions, cell-phone calls database details, Web browsing details, Document searching in the Internet are some of the examples for stream data. Stream data mining algorithms must be designed to work one pass of the stream data only.

More sophisticated state-of-the-art indexing techniques are needed to search, query the huge stream data. Hash indices and B-Trees are index structures that can handle only one dimensional data. To handle multidimensional data we need special indexing techniques. Spatial data is an example for multidimensional data. To handle spatial data, useful indexing techniques are – 1. K-d Trees, 2. Quad trees, 3. Octtrees, 4. R-Trees, 5. R+ Trees, 6. R*Trees, and 7. E-Trees etc. K-d Trees are two dimensional indexing structures. K-d-B tree is a multidimensional indexing structure and it is an extension to the K-d Tree. A quad tree is two dimensional indexing structures. An Octtree is 3-dimensional indexing structure. R-tree and its variants are very useful indexing structures for indexing objects such as points, line segments, rectangles, squares and polygons. An R-Tree indexing structure is a balanced tree structure with the indexed objects stored in leaf nodes. Each tree node is associated with one bounding box. The bounding box of a terminated node is the smallest rectangle parallel to the axes that contain the bounding boxes of its child nodes. Each leaf node stores the indexed objects and each internal node stores the bounding boxes of the child nodes along with the pointers to the child nodes.

Important operations of R-Tree are- 1.Search, 2. Insert, and 3. Deletion

1. Search overlap may occur between or among the bounding boxes associated with sibling nodes. For searching a specific point all the child nodes of a bounding box containing the point must be searched. For searching and querying multiple paths may have to be searched.

2. Insert into R-Tree- To insert a new object into the R-Tree, we have to find the leaf node that has the place for storing. Finding an exact leaf node for insertion is costly. Sometimes many tree traversals are needed. If the bounding box of the new object and the bounding box of any children of R-Tree are same then that common bounding box will be selected automatically as a heuristic. If the selected leaf node for insertion is full then the leaf node is split into two nodes and the corresponding bounding rectangle is adjusted. Sometimes heuristic quadratic splits are used for splitting.

3. Deletion- We can perform deletion in R-Tree similar to B^+-Tree or we can redistribute all the entries of under full nodes in order to improve clustering efficiency of the R-Tree. R-Trees allow better storage than K-d trees or quad trees, because the object is stored only once in R-Trees and also half full property is very easy to satisfy. Query processing is very slow because multiple paths must be traversed. R-Trees are better suitable for spatial data mining applications.

Spatial data may be represented using raster graphics format. Vector format is used to represent maps. Many real time objects such as roads , bridges , buildings and lakes, are represented by using graphics components such as points, lines, polygons and partitions. Many real time applications are Geographic databases based applications. Locations of telephones and electric cables, pipes, optical fiber optic cable locations, locations of cell towers are maintained using spatial data bases. Locations of hotels, restaurants, temples, vehicles, flights, trains, bridges etc are represented using spatial databases. Spatial cluster, spatial classification, spatial outliers are also some of the applications of spatial database. In order to organize data into multidimensional, we can use spatial data cubes. Spatiotemporal database is a type of spatial database that stores spatial data objects that change with time.

Stream data size is very large: Video conference data, telecommunication data, bank transaction data etc are examples for stream data. Large volumes of data streams are generated in Internet traffic, communication networks, retail industry, remote sensors, and electric power grids, in scientific and engineering applications. It is very difficult to store large volumes of stream data and it is also very difficult to process the stream data many times. We need efficient and effective state of the art techniques and we have to develop single-scan, on-line, multi level, multidimensional stream processing techniques. Stream data sizes are increasing from terabytes to peta bytes rapidly. Many stream data analysis techniques are invented. Different types of stream data mining applications are listed below:

1. Multidimensional analysis of stream data
2. Stream data cube based modeling techniques
3. Stream data frequent pattern mining
4. Stream data classification
5. Stream data clustering

For accurate, efficient, effective and fast processing of stream data new data structures, analysis tools, processing techniques, models, tools and algorithms are needed. There must be tradeoff between accuracy and storage capacity because infinite storage is not available. We need efficient, robust, scalable, effective algorithms in terms of space and time complexity. The best time complexity for stream data processing is logarithmic. That is O(log n).

Synopsis data structures are useful to find approximate answers to the problems with the high probability such that the answer is within a factor of the actual answer. Random sampling is a technique where a set of samples are taken for mining instead of taking entire stream data. Reservoir sampling technique can be used to select an unbiased random sample of S elements without replacement. Here S number of sample sets is selected for mining. Sliding window model is another technique useful for stream data mining. This method uses only the recent data. Each time the number of tuples equal to the size of the window are processed, next time a new window is processed. Sliding window model is particularly useful in applications such as stocks, sensors, video conference, cell phone stream data etc.

Data streams are potentially infinite and it is impossible to process each element more than once. There may exist multiple parallel data streams in a stream data management system. We can also pass queries to stream data. Data stream query may be either one-time or continuous.

Data streams are dynamic, infinite, fast, continuous, multi-dimensional, combinations of different data patterns and different distributions of complexities. In order to find critical changes, unusual patterns and interesting patterns, multidimensional stream data analysis on aggregate measures required.

Sometimes there is a need to apply frequent-pattern mining on data streams. All the previous data mining algorithms must be modified in order to handle stream data. Stream data management is really a challenging task. Lossy counting algorithm is an efficient algorithm to find frequent items. Main characteristic of stream data is that they are time-varying. Most important point that must be considered in the case of stream data is that concept drift. In the stream data, data distributions change continuously. The change in the data distributions cause changes in the target classification model. In other words classification model is directly affects to the changes in data distributions in the stream data.

Different types of proposed methods for stream data classification are:

1. Hoeffding tree algorithm
2. Very fast decision tree (VEDT)
3. Concept adapting very fast decision tree (CVFDT)
4. Classifier ensemble approach

Ensemble classification approach is particularly suitable for stream data classification. An ensemble is a group of classifiers such as C4.5, ID3, Naïve bayes, Bayesian belief networks etc. Now a days stream data sizes are very large. Sometimes, it is not possible to process each record more than one stream data mining techniques must be fast, effective, efficient and accurate. When the stream data size is very large computational cost increases exponentially. The main characteristic of any stream data mining algorithm is that it must process the complete data in one pass only. Large volumes of data stream processing DM algorithms must be enriched with many latest trends and techniques. Most important stream data mining techniques are categorized as follows:

1. Stream data classification
2. Stream data clustering
3. Frequent stream data pattern mining
4. Indexing data streams
5. Dimensionality reduction in stream data
6. Stream data mining in distributed environment
7. Stream data mining in sensor networks
8. Join operations in stream data
9. Change detections in stream data
10. Data stream cube analysis

Indexing data streams

Data stream indexing is very useful for certain operations such as aggregate queries, transaction data analysis, whether simulation data analysis, war simulation data analysis, etc. Indexing usage is inevitable in analysis, computing, processing, querying and various other mining techniques of large data streams.

Indexing usage is inevitable for many data-centric applications such as trend analysis, network traffic management, web-click streams, finding location of a specific pervasive device, and intrusion detection etc. Indexing usage is both time & space efficient & also provides a high quality of answers to user queries. Indexing a dynamic stream data is very difficult. A dynamic indexing structure provides high query efficiency. A good index is always associated with parameters such as accuracy, speed, space and time.

Indexing data structures that are used for stream data mining are R-tree, R+ tree, R*-tree. Corresponding to new stream data records, the new features must be computed and inserted into the dynamic index structure and at the same time old features must be deleted from the index structure spatial data is indexed using R*-tree indexing structure. We need efficient indexing architecture for both time & space to extract features of stream data & then add these indexing features for improving query performance. We need low cost maintenance procedures for the index structure. Indexing technique definitely decreases streams record processing time & minimizes the space required for incremental computation.

Stream data classification is one of the most important data mining techniques. Ensemble learning is one of the state of the art data stream classification technique. Ensemble stream data learning technique manages large volumes of stream data and controls concept drifting. All the existing data stream management methods mainly concentrates only on constructing accurate ensemble models from the stream data. Existing stream data learning techniques can be applied to only a limited set of real world applications because prediction cost of stream data learning is very high. In many real time applications data streams arrive at a speed of Giga Bytes (GB) or Tera Bytes(TB) per

second but classification or clustering task must be completed within a limited time unit. Must indexing structures are proposed to speed up the stream data classification & each indexing structure has its own advantages & disadvantages.

Existing stream data classification techniques use a linear scan technique to process all the base classifiers. This linear scan technique is used to process all the base classifiers. That is, linear scan is costly for many real time applications. Hence, a new and low cost ensemble stream data classifications technique is needed for many modern real time applications that are based on ensemble learning.

A new indexing data structure called E-Tree is proposed to reduce the time complexity from linear to log. E-Trees are automatically and dynamically updatable in order to reflect stream data changes and pattern changes. Early ensemble streams mainly concentrated on building only accurate ensemble stream data models. The prediction cost of ensemble learning increases linearly as the number of classifiers in the ensemble learning increases. This linear time complexity is not suitable for many real life applications.

Classification is one of the most important data mining techniques. Ensemble learning is one of the most important data stream classification technique in data stream mining. Stream data mining has many real time applications such as ATM transactions management, Finding incorrect documents in website management, Spam filtering etc. Problems with stream data are-

1. Volume of stream data is very high
2. Changes in the features and patterns of stream data occurs so frequently

Many ensemble based stream data models have been proposed in the literature of stream data management. Ensemble classification, ensemble clustering, ensemble association, ensemble incremental classification and clustering, ensemble fuzzy classification and clustering, are proposed based ensemble management modelling techniques. Divide-and-conquer is the most popular ensemble stream data management technique to handle large volumes of stream data with concept drifting. In ensemble stream data classification stream data is divided into a predefined number of partitions and a new classifier is constructed for each partition and then these base classifiers are combined in different ways for predicting the class label of a new incoming stream record.

Advantages of ensemble learning design are

1. Ability to handle large volumes of stream data
2. High scalability in managing large streams of data
3. Capability to handle changes in patterns and trends that occur dynamically
4. Increased accuracy of classification and clustering
5. Reduced **errors** in classification
6. Easy to parallelize stream data processing and prediction
7. Reduced search times for obtaining accurate results in stream data processing

All the previous ensemble learning stream data modeling techniques are mainly concentrated in constructing accurate ensemble models and much importance is not given to efficiency in stream data processing. previous methods are sufficient when total number of base classifiers are less than 30 and prediction efficiency is not important. Previous ensemble learning models are not suitable for many real world applications, particularly to model dynamic changes in trends and patterns in stream data. Time complexity of ensemble learning is linear, that is O(n), which is not suitable for many real time applications. Many state of the art ensemble learning models have sub-linear (logarithmic) time complexity.

Ensemble approaches are stream data management tools particularly useful methods for handling concept drift. Ensemble approach is an up-to date dynamic stream data classification approach. It discards least accurate classifiers and updates all the remaining classifiers. It is incrementally updatable. Ensemble classification results are more accurate than any single classifier approach.

Stream Data

Stream data is dynamic. Main features of stream data are:

1. Data size is infinite
2. Change in the stream data occur dynamically
3. Fast response times are required
4. Very difficult to store completely and to process all data
5. Multiple scans of the same data is impossible
6. Data distributions within the stream data change dynamically and fastly

7. Aggregate measures of stream data changes dynamically
8. Multi dimensional analysis and modelling is needed
9. Sometimes multiple modelling techniques are needed
10. State-of-the-art structures, tools, methods indexing techniques and other processing ideas are required.

2. PROBLEM DEFINITION

Dynamic stream records are represented as $S=\{(x_i, y_i)\}$, where each x_i represents a set of k attributes and y_i is a class label with two values-yes or no. For simplicity a two class problem is considered. Stream data is divided into n partitions and 'n' decision tree base classifiers $c_1, c_2, c_3 \ldots\ldots c_n$ are constructed all 'n' base classifiers are combined into a single ensemble classifier E. Each base classifier c_i consists of x decision rules represented by if-then clauses. Total number of decision rules are n*x = nx. All these nx decision rules are represented in spatial database as nx spatial objects. Main goal is to construct best ensemble model to predict incoming stream record accurately within logarithm (sub-linear) time complexity O(log n). To achieve this goal each base decision tree classifier C_i is converted into a batch of spatial objects (SOs). This batch of spatial objects is called spatial database, SD. Entire ensemble model E is converted to a spatial database (SD) containing all spatial objects. With this idea given original problem is changed to classifying each incoming dynamic stream record r by systematically searching over the spatial database (SD).

3. ALGORITHMS

Proposed E-Tree Insertion Algorithm with heuristic split method

Algorithm 4: Inserting a node into E-tree

Input :

1. E- tree T
2. Classifier C (one or more rules)
3. m , minimum number of entries in a node
4. M, maximum number of entries in a node

Output : updated or modified E-tree T'

1. P←T .tree.root // get root of E-tree
2. Foreach decision rule R∈C do
3. L ← searchLeaf(R, P)
4. If(L.size < m) then
5. $L \leftarrow L \cup R$
6. T'← UpdateParentnode (L)
7. Else
8. $<P_L,P_R> \leftarrow$ ***ModifiedsplitNode(L, R)***
9. T'← adjustTree(T, P_L, P_R)
10. Endif
11. Endfor
12. Insert the classifier, C into table structure
13. Output the tree, T', after inserting the new classifier, C

Explanation of Modified E-Tree insertion operation

Node split occurs during the insertion of dynamic stream data records into the ensemble tree indexing structure. Proposed node splitting method reduces the time complexity of node split task from $O(n^2)$ to O(n log n). New method is easy and simple to split the nodes. Always minimum and maximum node sizes are specified clearly at the beginning.

Advantages:

1. Node splitting process is very fast.
2. It is a very simple and efficient heuristic technique for node splitting.
3. Entries in the node are sorted first and then divided into two halves, left node and right node.
4. Proposed method is reasonably very good asymptotically.

Whenever new trends and patterns are found during dynamic stream data classification, the E-tree insertion algorithm insert new base classifiers into the ensemble model and this model is automatically converted into the spatial space data object and inserted into the E-tree indexing structure. Whenever a new classifier, C, is created, a new entry associated with the newly created classifier is inserted into the E-table structure and similarly all the decision rules, R, belonging to the newly created classifier are inserted into the E-tree structure one after the other, and linked all the rules together by the respective pointer entries.

Inserting decision rules of a new classifier into E-tree is very similar to the insertion operation of R-tree indexing structure starting from the root search proceeds downwards in order to find a leaf node that covers the rules of the new classifier. Once a leaf node is found, it is checked to find whether space for insertion is available or not. If the leaf node contains less than the maximum number of entries, M, then new set of rules are inserted into the leaf node. Parent node is updated appropriately. After searching, if it is found that there is no space for new insertion, then the current leaf node is split into two nodes

and then new set of rules are inserted. Node splitting in E-tree is a difficult step.

A decision classifier C4.5 is used to generate decision rules from the dynamic stream data. All decision rules are "hard" decision rules only. That is decision rules are not fuzzy. Various measures used for online query evaluation are:

1.Time cost- computational cost of E-trees is much lower than the computational cost of traditional ensemble models because E-trees are height balanced indexing tree structures that are used to index all classifiers in the ensemble.

2.Memory cost- Through E-trees consumes larger memory during stream data record classification, the memory size is in affordable range only.

3.Accuracy- prediction accuracy of E-trees is high and it is equal to the accuracy of original ensemble models.

E-Tree ensemble learning

Architecture of ensemble models on dynamic data streams using E-tree indexing structure is shown in the fig 4. Training module maintains and controls all the insertion and deletion operations of E-trees. Training module is responsible to monitor E-tree operations for constant updating of E-tree. Prediction module is responsible to predict the class label of newly arriving dynamic stream record x by using synchronized copy of E-tree received from training module.

E-tree search operation is applied to make online predictions. We assume that stream is coming with un-labeled data. Initially the buffer in the training module is filled with incoming stream records. Records in the buffer are labeled by human experts or intelligent labeling machines. This labeling process is very slow, costly and time consuming and only a small percentage of incoming records at regular intervals regularly in order to provide uniform labeling. As soon as buffer is full, automatically a new classifier is constructed and then it is inserted in the ensemble. New classifiers of E-tree are constructed regularly. Once the maximum capacity of the E-tree is reached automatically old or outdated or un-useful classifiers are deleted from the E-tree by executing E-tree deletion operation. Updated and latest E-tree will be synchronized and E-tree copy is passed to the prediction module.

4. EXPERIMENTAL RESULTS

Initially a set of decision tree classifiers are constructed. Each classifier represents a set of decision rules. All these decision rules are transformed into spatial data objects. Spatial data objects are stored in the ensemble tree indexing data structure. Main operations of ensemble tree data structure are – insert, search, deletion and classification operations. Previously these trees are used only for searching. In this paper, we use ensemble tree data structures for classifying the newly arriving stream record.

R-tree is the best indexing data structure for indexing spatial database objects. Rectangles are used to represent rules of decision tree classifiers. A set of decision tree classifiers are generated and then these classifiers are combined for ensemble classification. Ensemble classification is particularly useful for dynamic stream data management. A mapping function is defined to transform all classifier rules into corresponding spatial database objects.

When the numbers of classifiers in stream data classification are very small then normal indexing structure is sufficient. When the set of classifiers is very big we have to use a new state of the art indexing structure for stream data management. Ensemble tree indexing structure is one such state of the art indexing data structure. Ensemble tree indexing structure is an extension of the R-tree like structure. B-trees are only one dimensional indexing structure. They are suitable for indexing spatial database objects.

TABLE 1 Input Data (Classifier Rules)

20, 90, 30, 100, 1
40, 110, 50, 120, 1
40, 95, 50, 105, 1
60, 30, 70, 130, 1
35, 60, 65, 85, 1
80, 60, 90, 120, 1
75, 90, 85, 100, 1
10, 10, 20, 20, 1
25, 15, 55, 40, 1
100, 30, 130, 50, 1
110, 10, 120, 70, 1
115, 25, 125, 45, 1
80, 70, 100, 80, 1
40, 20, 50, 30, 1
50, 70, 60, 80, 1
60, 110, 70, 130, 1
70, 70, 90, 90, 1

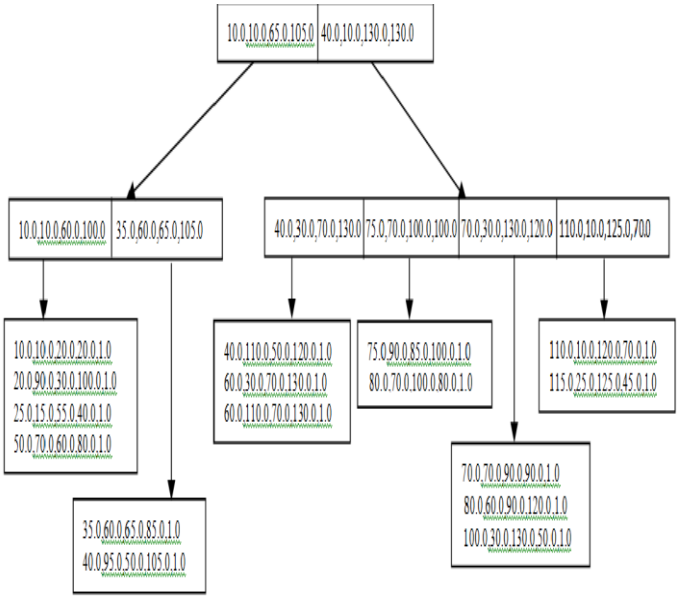

Fig -1. Enhanced ensemble tree for the input data given in TABLE1

5. Conclusions

Ensemble classification is common and more popular in stream data classification. Ensemble tree indexing data structure is constructed for ensemble classification. Node splitting occurs during ensemble tree construction. One heuristic new method is found for node splitting. In this new method, time complexity of node splitting is O(n log n) and new method is very easy to apply. First values are ordered and then split is applied. In future we will try to find other node splitting methods with linear time complexity.

6. REFERENCES

[1] Peng Zhang, Chuan Zhou, Peng Wang, Byron J. Gao, Xingquan Zhu, and Li Guo E-Tree: An Efficient Indexing Structurefor Ensemble Models on Data Streams "IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 27, NO. 2, FEBRUARY 2015"

[2] J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, third ed. Morgan Kaufmann, 2011.

[3] J. Quinlan, C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.