An Approach for Distributed Streams Mining Using Combination of Naïve Bayes and Decision Trees

Meng Zhang
School of Computer Science
Beijing University of Technology
Beijing, China
therealzhangmeng@gmail.com

Guojun Mao
College of Information
Central University of Finance and Economics
Beijing, China
maximmao@hotmail.com

***Abstract*—Nowadays we have various kinds of data generated at high speed in distributed environment. In many cases, it is difficult or unallowed to gather all the distributed data into a central place for processing. So we have to perform part of the work at the location where data is generated. In this paper, we present an approach for mining distributed data streams by using combination of naïve Bayes and decision tree classifiers. The method takes advantage of both distributed and stream mining characteristics. Each local site uses ensemble classifiers of decision tree to learn a concept of incoming stream and transmits local pattern to a central site. The central site combines the collected patterns to build a global pattern using an attribute-weighted strategy in order to relax the attribute independent constraint of naïve Bayes model. The experiment shows that by using this approach we can get comparable understanding of the global data while reducing transmission load and computation complexity.**

Keywords-data stream mining; distributed streams

I. INTRODUCTION

Nowadays, one of the main research directions in data mining field is data stream mining. Because it is common that real world applications produce streaming data, such as transactions in banking, stock market and e-commerce. Among these learning tasks distributed environment is also a common context due to the widely utilization of the Internet. Data stream mining has some characteristics different from traditional data mining. The data is generated at high speed, arrived continuously and potentially infinite. So it challenges the storage, computation and communication capabilities of the computer systems.

By examining previous related work in data stream mining, we found that the train of thought in many contributions is to get inspiration from traditional mining and extend the idea to fit the streaming context. Distributed mining context is a loosely coupled environment. Each element in the system can do some independent work without affect other ones. We think this trait is proper to combine separate classification methods and bring advantages of both methods into play.

Naïve Bayes [7] and decision tree [10] are two of the most widely used induction algorithms for classification tasks. Decision tree is a common learning technique which gives easy interpretation of data and performs well in many learning fields. Naïve Bayes classifier has a simple assumption that attributes are independent of each other. Although this assumption is often violated in real learning tasks, naïve Bayes classifier can get comparable prediction accuracy with other methods in many domains.

Since naïve Bayes classifier was introduced, there are many researches on improving this method. Langley and Sage [8] introduce an approach of using only a subset of the attributes to make predictions and they show it can improve accuracy in domains with redundant attributes. Elkan [3] applies Boosting to naïve Bayes classifier which builds several classifiers instead of only one. Many people have worked on combining decision tree with naïve Bayes to get better prediction results. Kohavi [6] builds decision tree containing naïve Bayes classifiers in the leaves which is called NBTree. He points out that this hybrid frequently outperforms both constituents in larger databases. Ratanamahatana and Gunopulos [11] propose a method selecting attributes that appear in the nodes of decision trees and using only these attributes to do naïve Bayes learning. Though their approach gets better results than C4.5 decision tree and naïve Bayes classifier on many datasets, it needs several scan of the training set.

Most of the improved methods are done under the condition of traditional mining on static datasets. Due to the characteristics of streaming data, it is necessary to use different strategies to do the mining task. Domingos and Hulten [2] describe a very fast decision tree (VFDT) learning algorithm based on Hoeffding trees for streaming data. They build a decision tree at the beginning and refine it incrementally when new data is coming. Street and Kim [12] propose an ensemble method called SEA to mining data stream. This method uses many relatively small decision trees instead of maintaining one large decision tree, so it can refine the model more efficiently than incrementally maintained single tree.

In distributed mining field, Parthasarathy et al. [9] introduce a distributed stream processing architecture. In this architecture, each distributed computing node learns a local model and a central site uses these local models to generate a final model. They point out that this method provides both parallelism and scalability. Sun et al. [13] present a hierarchical algorithm for summarizing several local patterns into a global pattern which requires only a single pass over the data.

In this paper, we extend the approach of combining decision tree and naïve Bayes classifier to distributed data

 ISBN:978-1-61208-115-1

streams mining environment. We build an ensemble of C4.5 decision trees at each local site and compute statistical summary of each data chunk. Then we select some attributes that are the split attributes at top levels of the C4.5 decision trees in the ensemble as key attributes. The local pattern is made up of these key attributes and the statistical summary. Each local site transmits its local pattern to the central site and the central site builds a new naïve Bayes classifier based on the local patterns to maintain the global pattern. By transmitting local patterns rather than raw data from local sites to central site, we can achieve a lower traffic cost in distributed environment. Besides, building naïve Bayes from statistical summary is more efficient than learning it from raw data.

The rest of this paper is organized as follows. In Section 2, we describe our method for combining decision tree and naïve Bayes in distributed streams mining task. Section 3 shows the experiment results and analysis. Section 4 discusses the conclusions of our method and points out some future work.

II. LOCAL SITE MINING PROCESS

Each local site continuously receives data, and the received data is buffered for a special time interval to form a data chunk. When a chunk is available for processing, we will do three things on it as follows:

Firstly, a statistical information structure for this trunk, called *statistical summary*, is constructed. For a discrete attribute, we count the number of instances per attribute value per class. For a numeric attribute, we calculate sum and quadratic sum per attribute value per class. When computing the distribution of attribute values in a trunk, the number of instances per class in this trunk can also be gotten. Here we use a matrix to store the *statistical summary*.

$$\begin{bmatrix} & \begin{bmatrix} c_1 & c_2 & \cdots & c_n \end{bmatrix} \\ \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix} & \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1n} \\ s_{21} & & & \\ \vdots & & \ddots & \vdots \\ s_{m1} & & \cdots & s_{mn} \end{bmatrix} \end{bmatrix}$$

Figure 1. Matrix to store *statistical summary*.

In the matrix, each column represents a class value and each row represents an attribute. Each cell of the matrix is called a *summary record*. For a discrete attribute, e.g. attribute a_1 has *d* distinct values, the *summary record* s_{11} is an array of length *d* with each cell stores number of instances which have that specific value on a_1 and take the class value c_1. For a numeric attribute, e.g. attribute a_2, the *summary record* s_{21} is a structure made up of the sum and quadratic sum of attribute a_2's value of all the instances that take the class value c_1. Using matrix to store the *statistical summary* is convenient and proper because we can compute the global model by simply doing matrix addition of the *statistical summary* from all the local sites.

Secondly, a C4.5 decision tree is generated for the current data chunk, and use the current data chunk as a test set to validate the previous ensemble. During the process, we test each classifier in the ensemble on the newly arrived data chunk and also the newly constructed decision tree. Then we use the average classification accuracy of these classifiers and a threshold bound to decide whether the current tree is appropriate to add to the ensemble. Only if the accuracy of the current tree is higher than the average accuracy minus threshold bound, it can be added to the ensemble. If the ensemble is not full, we directly insert the new C4.5 decision tree into the ensemble. Otherwise, according to the testing result, the decision tree in the ensemble that gets the worst result on the current data chunk will be replaced by the new tree that is just constructed. Here, we take a simple replace tactic to refine the ensemble classifiers. This is because that the newest tree can represent the new changes of the data stream better.

Thirdly, the key attributes in the ensemble are selected. Given an integer *k*, the attributes locate on top *k* levels of all trees in the ensemble are considered as key attributes, and called as *Top-k-level-attributes*. Research like [11] has proved that the attributes that are located closer to the root are more important than others in a decision tree, and so we just need to pick those attributes appear at the top *k* levels of the decision trees. Take ensemble classifying into consideration, an *impact factor* need to be set for each attribute of each tree, which is inversely proportional to its location in the tree. That is, the root of a tree has the maximal *impact factor* value; the closer a node is located to the root, the larger its attribute's *impact factor* is. After key attributes are selected, if an attribute of them appears on several trees in the ensemble, we calculate the average value of the *impact factor* in these trees as the value of *impact factor* for this selected attribute. As far as an attribute *impact factor* is concerned, it is directly related to the levels that this attribute locates on all local decision trees. We can assign 2^k to the *impact factor* of the root attribute, 2^{k-1} to the *impact factor* of the attribute locates at second level of the tree, and so on. By doing this, the difference of *impact factor* between two levels is relatively obvious. For those attributes are not key attributes, their *impact factor* are assigned to 1.

In our design, the local pattern of a local site is made up of the *statistical summary* and the *Top-k-level-attributes*. Figure 2 gives description of mining a local pattern in the local site.

 ISBN:978-1-61208-115-1

D: the current data chunk in the data stream
E: an ensemble of C4.5 decision tree classifiers
T: a C4.5 decision tree classifier in the ensemble
T_c: the current learned C4.5 decision tree
T_w: the worst decision tree in the ensemble

repeat when there are more data chunks in the stream
  read one data chunk D
  compute *statistical summary* of D and store it in a matrix structure
  construct a decision tree T_c based on D
  use D to evaluate all decision trees in the ensemble E and also the T_c
  if T_c is appropriate to add to ensemble
    if E is not full
      insert T_c into E
    else
      find the worst tree T_w in E
      replace T_w with T_c
    end
  end
  for each tree T in E
    select out *top-k-level-attributes*
  end
  calculate the *impact factor* of these attributes
  transmit local pattern to central site
end

Figure 2. Local site process algorithm.

III. GLOBAL MODEL CONSTRUCTION

When a local site finishes processing a data chunk, it will transmit its own local pattern to the central site. As soon as the central site receives the local patterns from all the local sites for a data chunk, it will start generate the global pattern on this time. In our design, these local patterns have the same structure. Generating the global pattern is divided into two steps. Firstly, we need to build a naïve Bayes model from the *statistical summary* which is part of the local pattern. Secondly, we use attribute weights to adapt naïve Bayes model to improve classification performance.

We know that naïve Bayes classifier comes from the Bayes' rule of probability theory. It also has a core assumption that attribute is independent of each other. So based on these two conditions, we can describe naïve Bayes model as:

$$P(C_i \mid a_1 a_2 ... a_n) = \frac{P(a_1 \mid C_i)P(a_2 \mid C_i)...P(a_n \mid C_i)P(C_i)}{P(a_1 a_2 ... a_n)} \quad (1)$$

In the formula, C_i is for the value of class i and a_j is for attribute j. Since for a given data instance, the denominator is the same regardless of its class value, we can just consider only the numerator part of the formula. So our task is just to determine the probability of each attribute for a class value from local *statistical summaries.*

There are two types of attributes, discrete and numeric. For a discrete attribute, the probability $P(a_k|C_i)$ can be estimated by the proportion of instances that have the Attribute a_k to the number of instances in Class C_i. The probability $P(C_i)$ can be estimated by the proportion of instances in Class C_i to the size of investigated data. For numeric attribute, it is common to assume a normal distribution model. Thus, the probability $P(a_k|C_i)$ and $P(C_i)$ can be determined from the distribution function of Attribute a_k and Class C_i, respectively. All the information we need to calculate the probability is contained in the local *statistical summaries*.

In order to relax the attribute independent assumption, we introduce attribute weights, and adjust the numerator part of the Bayes formula as follows:

$$P(a_1 \mid C_i)^{w(a_1)} P(a_2 \mid C_i)^{w(a_2)} ... P(a_n \mid C_i)^{w(a_n)} P(C_i) \quad (2)$$

Where $w(a_i)$ is the weight of Attribute a_i. Note that the range of the probability is between 0 and 1. So the less the weight, the more impact the related attribute has. And we use a simple mathematical function to transform the impact factor of attribute to its weight:

$$f(x) = \alpha^x, \ 0 < \alpha < 1 \quad (3)$$

By doing this, the key attributes take greater value of impact factor, so they get smaller value of weight, which in turn have greater probability result and will blow their importance up in decision process. From another point of view, attribute dependencies can be partly eliminated by using attribute weights. Figure 3 describes the process of global model construction at the central site.

repeat receive local patterns from local sites
  sum the matrices of the local patterns up to get *statistical summaries* of the whole data at current chunk
  construct naïve Bayes model based on all the *statistical summaries*
  transform the *impact factor* of all the attributes to generate their weights in the model
  use this weighted Bayes model with to do classification on global data
end

Figure 3. Global model construction algorithm.

IV. EXPERIMENTAL RESULTS

We did the experiment in a simulated distributed environment and the sites are implemented as multiple threads in the program. Though we chose three local sites and one central site for keeping simplicity, our design can be applied to more complicated example easily. We used WEKA [5], the famous open source software for data mining,

 ISBN:978-1-61208-115-1

and two datasets from UCI repository [4] in the experiment. We chose the datasets based on some criteria, e.g. containing many attributes with both discrete and numeric, having large number of instances. The datasets that we selected both contain tens of thousands of instances.

The Adult dataset [6] has information for task of determining whether a person makes over 50K a year. It includes 14 attributes with 6 of numeric type and 8 of discrete type.

The Forest CoverType dataset [1] has information for predicting forest cover type from cartographic variables. It includes 54 attributes and most of them are discrete. It has 7 distinct classes.

In the experiment, we first randomly repeated the data and shuffled it to generate sufficient amount of data. Then we split the dataset into three parts, each for a local site. We continuously read the dataset at a different rate to simulate the data stream. We tested the classification accuracy at the central site. As a comparison, we centralized all the data at the central site to do a single stream mining using ensemble C4.5 decision tree classifiers only.

We did the experiment with some parameters. Each data chunk is composed of 500 instances and the minimal interval between the arrivals of two instances is set to 10 milliseconds. Take into account the number of attributes in each dataset, we set the parameter *k* of *top-k-level-attributes* to 2 with Adult dataset and 3 with CoverType dataset.

Figure 4 and 5 shows the accuracy result of our experiment with Adult dataset and CoverType dataset respectively. In these plots, we can see the accuracy of distributed mining method is lower than the accuracy of centralized mining method most of the time. Sometimes the distributed result can outperform the centralized method. On all accounts the difference between them is not much. So we can say the performance of our approach is comparable with the centralized method.

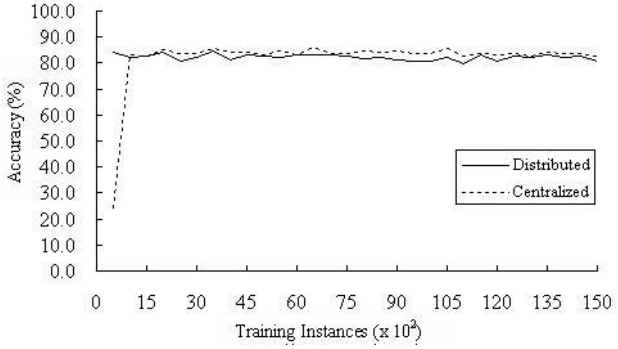

Figure 4. Result of Adult dataset with 500 instances per chunk.

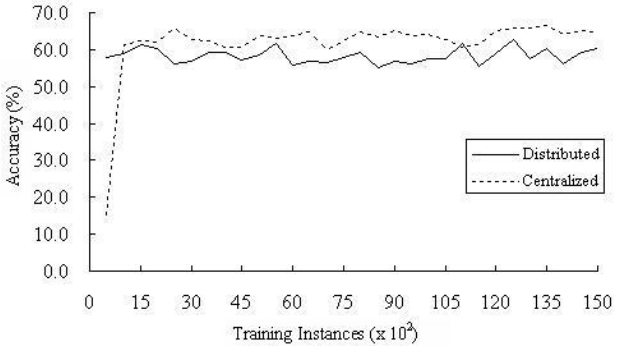

Figure 5. Result of CoverType dataset with 500 instances per chunk.

We also record the number of *top-k-level-attributes* that selected from each data chunk. Figure 6 indicates that on average only 7 out of 14 attributes for Adult dataset were selected as key attributes. Figure 7 shows that on average 25 out of 54 attributes for CoverType dataset were selected as key attributes. Concentrating on these key attributes improves our global classification model.

V. CONCLUSION

We have described a method of using C4.5 decision tree and naïve Bayes classifiers in distributed stream mining. This is to extend the combination of decision tree and Bayes learning. It takes advantage of the characteristic of both decision tree and naïve Bayes model. Decision tree is easily interpreted and has hierarchical structure which can distinguish some important attributes from others. By using ensemble of decision trees, we can eliminate the tree pruning work which will slow down the processing efficiency. And it will not degrade the performance because the ensemble method can combine several weak classifiers to get good results. Naïve Bayes classifier can be trained efficiently given the *statistical summary*. It is particularly suited when the dimensionality of the dataset is high like in our experiments. The experimental result shows that it is an applicable approach for its simplicity and efficiency.

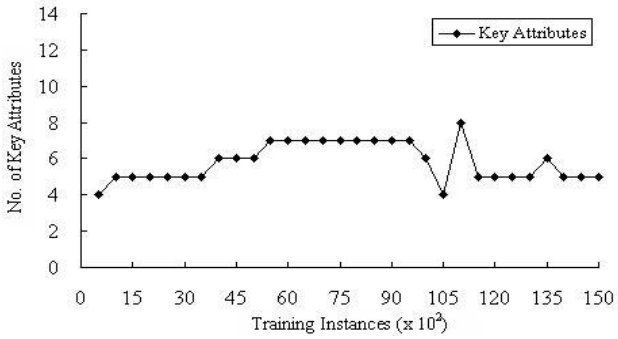

Figure 6. Key attributes selected from Adult dataset with 500 instances per chunk.

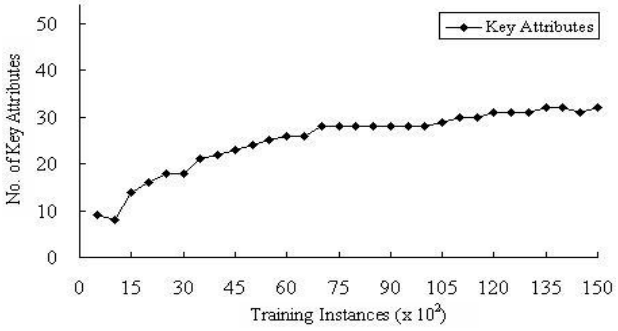

Figure 7. Key attributes selected from CoverType dataset with 500 instances per chunk.

The information technology develops and changes fast. This work indicates that researching on some of the classical mining tasks can give us inspirations and we can use them for reference when the target or context changes.

In future research, we will focus on making our approach more general to other datasets. That is, datasets with less amount of data, less number of attributes, etc. Besides, handling the drift of concept is another issue to think about.

ACKNOWLEDGMENT

This research has been supported by Beijing Municipal Key Laboratory of Multimedia and Intelligent Software Technology and the National Science Foundation of China under Grants No.60873145.

REFERENCES

[1] J. Blackard and D. Dean, "Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables," Computers and Electronics in Agriculture, vol. 24(3), Elsevier Press, Dec. 1999, pp. 131-151, doi:10/1016/S0168-1699(99)00046-0.

[2] P. Domingos and G. Hulten, "Mining high-speed data streams," Proc. The 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 00), ACM Press, Aug. 2000, pp. 71-80, doi:10.1145/347090.347107.

[3] C. Elkan, "Boosting and naïve bayesian learning," Technical report, Department of Computer Science and Engineering, University of California, San Diego, 1997.

[4] A. Frank and A. Asuncion, "UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]," Irvine, CA: University of California, School of Information and Computer Science, 2010.

[5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten, "The WEKA data mining software: an update," SIGKDD Explorations, vol. 11(1), 2009.

[6] R. Kohavi, "Scaling up the accuracy of naïve-bayes classifiers: a decision-tree hybrid," Proc. The 2nd International Conference on Knowledge Discovery and Data Mining (KDD 96), AAAI Press, Aug. 1996, pp. 202-207.

[7] P. Langley, W. Iba, and K. Thompson, "An analysis of bayesian classifiers," Proc. The 10th National Conference on Artificial Intelligence, AAAI Press, Jul. 1992, pp. 223-228.

[8] P. Langley and S. Sage, "Induction of selective bayesian classifiers," Proc. The 10th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Press, Jul. 1994, pp. 399-406.

[9] S. Parthasarathy, A. Ghoting, and M. Otey, "A survey of distributed mining of data streams," Data streams: models and algorithms, C. C. Aggarwal, Springer Press, 2007.

[10] J. Quinlan, "C4.5: programs for machine learning," Morgan Kaufmann Press, 1993.

[11] C. Ratanamahatana and D. Gunopulos, "Scaling up the naïve bayesian classifier: using decision tree for feature selection," Proc. Workshop on Data Cleaning and Preprocessing (DCAP 02), at IEEE International Conference on Data Mining (ICDM 02), Maebashi, Japan, 2002.

[12] W. Street and Y. Kim, "A streaming ensemble algorithm (SEA) for large-scale classification," Proc. The 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 01), ACM Press, Aug. 2001, pp. 377-382, doi:10.1145/502512.502568.

[13] J. Sun, S. Papadimitriou, and C. Faloutsos, "Distributed pattern discovery in multiple streams," Proc. The 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 06), Springer Verlag Press, Apr. 2006, pp. 713-718.

[14] L. Wang, X. Li, C. Cao, and S. Yuan, "Combining decision tree and naïve bayes for classification," Knowledge-Based Systems, vol. 19(7), Elsevier Press, Nov. 2006, pp. 511-515, doi:10.1016/j.knosys.2005.10.013.

 ISBN:978-1-61208-115-1