2016 International Conference on Advanced Cloud and Big Data

A data streams analysis strategy based on hoeffding tree with concept drift on Hadoop system

Xin Song, Huiyuan He, Shaokai Niu
School of Computer Science and Engineering
Northeastern University
Shenyang, China
e-mail:bravesong@163.com

Jing Gao
School of Control Engineering
Northeastern University at Qinhuangdao
Qinhuangdao, China
e-mail: 75777041@qq.com

***Abstract*—The massive sensor data streams analysis in the monitoring application of internet of things is very important, especially in the environments where supporting such kind of real time streaming data storage and management. In order to support the classification of the massive sensor data streams, in this paper, a massive sensor data streams analysis strategy is proposed based on Hoeffding tree with concept drift for event monitoring application on Hadoop system. The proposed strategy is sufficient for sensor data streams classification tasks using map-reduce platform of Hadoop system. Finally, the possibilities of the strategy are demonstrated on spatial sensing data streams processing operations in comparison with existing solutions in the cloud computing environment. The simulation results show that the strategy achieves more energy savings and also ensures few amounts of sensor data retained in memory.**

Keywords—data streams classification; Hoeffding tree; Hadoop system; Hoeffding tree algorithm modification

I. INTRODUCTION

In recent years, an increasing number of emerging applications deal with a large number of heterogeneous sensor data objects in Internet of Things (IoT) because of a wide variety of sensor devices on sensing layer. Unlike traditional data sets, the sensor data streams flow in and out of a computer monitoring system continuously and with varying update rates. They are temporally ordered, fast changing, massive, and potentially infinite. In order to process the continuously changing sensor data streams, the application system terminal equipment must implement the data storage and the powerful computing ability for real-time collection, dissemination and extracting of sensor data to users and administrators anytime and from anywhere. Therefore, it is very necessary for managing massive and heterogeneous sensor data via combining the cloud computing and wireless sensor networks technology. Flexible computing resources and system services shared in network, omnipresent access and parallel processing are major features of Cloud Computing that are desirable for the smart event monitoring applications. For the small amount of effort when the spatial data processing task shift to distributed computing platform, Roberto Giachetta proposed a framework for processing large scale geospatial and remote sensing data in MapReduce environment. The porting of the framework to Hadoop requires some effort, but does not require complete overhaul [1].To discover knowledge or patterns form data streams, it is necessary to develop stream processing and analysis methods. In this paper, we proposed a massive sensor data streams analysis strategy using hoeffding tree with concept drift for cloud based monitoring application. The proposed strategy can realize the classification of original sensor data streams by distributed storage architecture to divide stream clustering and maintain such dynamically changing clusters.

The rest of this paper is organized as follows: in section 2, we briefly review some closely related works. The proposed data streams analysis strategy is derived and discussed in section 3. The validity analysis and performance evaluation are presented in section 4. Finally, the conclusions and future work directions are described in section 5.

II. RELATED WORKS

There have been a few of studies on the management of the sensor data streams using cloud computing. The conventional information system technology cannot manage the continuously changing properties of the sensor data streams. Ref.[2] formally presented a comprehensive framework for managing the continuously changing data objects with insights into the spatiotemporal uncertainty problem and presented an original parallel-processing solution for efficiently managing the uncertainty using the map-reduce platform of cloud computing. Michael Smit, Bradley Simmons and Marin Litoiu presented and implemented an architecture using stream processing to provide near real-time, cross-boundary, distributed, scalable, fault-tolerant monitoring [3]. For analyzing the data streams from city sensing infrastructures, Ref. [4] introduced an algorithmic architecture for kernel-based modeling of data streams. Recurring concept drift is one of the sub-types of concept drift. In recurring concept drift detection, it is very important to represent concepts and select the most appropriate classifier to classify. Feng Chao et.al proposed an algorithm conceptual clustering and prediction through main feature extraction (MFCCP) for classifying data stream with recurring concept drifts [5]. The algorithms for mining data streams have to make fast response and adapt to the concept drift at the premise of light demands on memory

978-1-5090-3677-6/16 $31.00 © 2016 IEEE
DOI 10.1109/CBD.2016.59

resources. Ref.[6] proposed an ensemble classification algorithm for high speed data stream. The experimental results of the algorithm showed that the method not only classifies the data stream fast and adapt to the concept drift with higher speed, but also has a better classification performance. The occurrence of concept drift can affect considerably the performance of a data stream mining method, especially in relation to mining accuracy. Ref.[7] proposed a method of support approximation for discovering data stream frequent patterns. The method experimental results had shown that in several studied cased of concept drift, the proposed method not only performs efficiently in terms of time and memory but also preserves mining accuracy well on concept drifting data streams. A large amount of stream data is generated in real-time, which has to be processed in real-time as well. Zhao Gansen et.al proposed an on-demand framework (SRAStream) based on the concept drifting detection mechanism. The concept drifting detection algorithm is used to measure the distance of the new clusters for the current data and that of the existing clusters [8]. Yang Hong and Fong Simon proposed a singletree with an optimized node splitting mechanism to detect the drift in a test-the-training tree-building process. The method experimental results showed that the new algorithm performs with good accuracy while a more compact model size and less use of memory than the others [9]. For mining textual stream with the presence of concept drift, Song Ge et.al present a novel ensemble model named, Dynamic Clustering Forest (DCF). The DCF model outperforms other state-of-the-art classification method in most of the high-dimensional textual streams [10]. Sethi Tegjyot Singh et. al proposed a grid density based framework for classifying streaming data in the presence of concept drift. The entire framework is designed to be one-pass, incremental and work with limited memory to perform any-time classification on demand [11]. The primary disadvantage of the block-based ensembles lies in the difficulty of tuning the block size to provide a tradeoff between fast reactions to drifts. Ref.[12] put forward an online ensemble paradigm, which aims to combine the best elements of block-based weighting and online processing. The algorithm used the adaptive windowing as a change detector.

III. Implementation of the Data Streams Analysis Strategy Based on Hoeffding Tree with Concept Drift

Sensor data streams are produced continuously, in real-time and dynamic monitoring environment, with high-volume, fast-changing and infinite flow. The proposed processing strategy based on Hoeffding tree with concept drift on Hadoop system aimed to the classification of massive dynamic sensor data streams.

A. The Concept Drift Detection

The proposed method processing the concept drift detection based on the x^2 fitting testing of hypothesis test. The based idea is that if the classification accuracy of two adjacent data blocks appears the significant change in the same category data, then the data concept in new data blocks is changed. That is, the classification model needs to be reconstructed. The algorithm processed the hypothesis test for the classification situations of each category on the current data block, let $z = 2$ (classified correctly or not), H_0 is the average classification situation of the category on the former data block of concept smooth, the significance level $\alpha = 0.05$, if $x^2 \geq x_\alpha^2(z-1)$ on the new data block, the category classification accuracy appears the significant change, thus the concept drift generation, conversely the concept smooth distribution.

B. The Classification Strategy of the Dynamic Data Streams Based on Hoeffding Tree With Concept Drift

The Hoeffding tree algorithm is a decision tree learning method for high-speed stream data classification. The basic idea of Hoeffding tree is: it is incremental, which can be seen in Fig. 1. The figure demonstrates how new sampling data are integrated into the decision tree as they stream in. The only once scan is performed for each data stream sampling. It does not need the additional storage space after the data stream sampling processing.

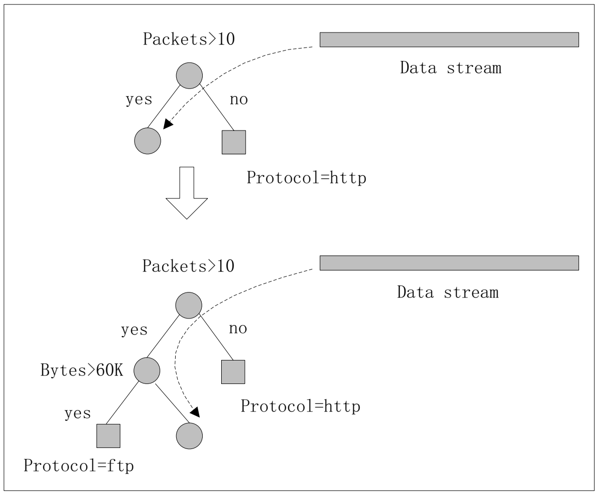

Fig. 1. The nodes of the Hoeffding tree are created incrementally as more samples stream in

The only object that must be stored in the memory is the decision tress itself with enough information saved in a leaf node for the tree growth. The decision tree can forecast the new data stream sampling in the processing of the training sample set at any time. The advantage of incrementally building the tree is that the system can execute it to classify data even while it is being built. The tree will continue to grow and become more accurate as more training data stream in.The pseudo code of Hoeffding Tree algorithm is shown in Fig. 2. Line 1-5 initializes the tree structure that there is only a root node in the decision tree at this time. Line 6-17 computes the information gain of the examples attributes and selects the node split according to the Hoeffding bound.

Suppose we make nl independent observations of a random variable l with range R, where l is an attribute selection measure. In the case of Hoeffding trees, l is information gain, X is the attribute, δ is the accuracy parameter where δ is user-specified and ε is the Hoeffding bound. In addition, the evaluation function G(X_i) is supplied, which could be information gain, gain ratio, Gini index, or some other attribute selection measure. Line 7 computes the maximize G(X_i) for one of the remaining attributes X_i. The goal of the Hoeffding algorithm is to find the smallest number of tuples n_l, for which the Hoeffding bound is satisfied. For a given node, let X_a be the attribute that achieves the highest G, and X_b be the attribute that achieves the second highest G. If **G**(X_a)- **G**(X_b)> ε , the strategy selects X_a as the best aplitting attribute with confidence 1- δ .

```
1: Let HT be a tree with a single leaf(the root)
2: for all training examples do
3:    Sort example into leaf $l$ using HT
4:    Update sufficient statistics in $l$
5:    Increment $n_l$, the number of examples seen at $l$
6:     if $n_l$ mod $n_{min}$=0 and examples seen at $l$ not all of same
          class then
7:        Compute G($X_i$) for each attribute
8:        Let $X_a$ be attribute with highest G
9:        Let $X_b$ be attribute with second-highest G
10:       Compute Hoeffding bound $\varepsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n_l}}$
11:       if $X_a \neq X_\varnothing$ and (G($X_a$)- G($X_b$)> $\varepsilon$ or $\varepsilon < \tau$ ) then
12:          Replace $l$ with an internal node that splits on $X_a$
13:          for all branches of the split do
14:                Add a new leaf with initialized sufficient
          statistics
15:          end for
16:       end if
17:    end if
18: end for
19: for each category do concept drift detection
20:  if the concept drift generation then
21:       delete the data classification model and reconstruct
          the new data classification model
22:  else save the classification model
23: end if
24: end for
```

Fig. 2. The pseudo code of Hoeffding Tree with concept drift

C. *The Workflow of the Sensor Data Streams Processing on Hadoop System*

To ensure real-time processing of a large-scale heterogeneous sensor data streams in cloud computing, the intermediate results of the preprocessing historical data were distributed at each cache nodes for reducing the duplication processing of the historical sensor data and avoiding the frequent transmission between nodes. Each node redundantly received data stream, so the pending processing data of the node were filtered by Map stage and operated Reduce calculation in the node local cache. When the existing node's local computing and storage resources cannot meet the real needs, the new increased node will be utilized for mobile cache data extension using the re-division technology. Finally, the local calculation results were synchronized to the distributed storage area. The Map-Reduce process on Hadoop system is shown in Fig.3.

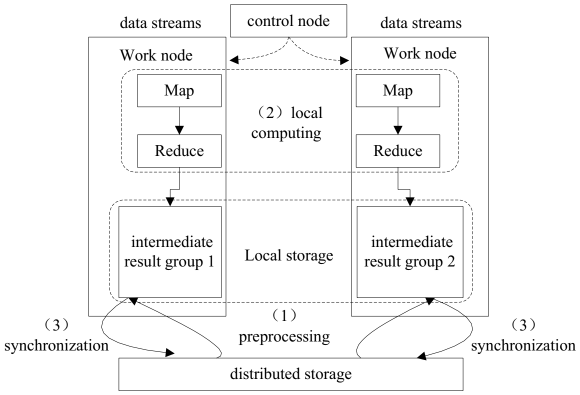

Fig. 3. The workflow of the sensor data streams processing

IV. Performance Evaluation

In this section, the obtained results are evaluated for the proposed strategy on various performance indicators. Along with the increase in the sensor samples, the Hoeffding bound value drops rapidly. When n=10000, the bound value is about 0.03, that is, **G**(X_a)- **G**(X_b)>=0.03. In this experiment, the bound value ε =0.05. The data flow velocity is 1MB/s (that is to say, the data is sent by 200B each, and 5000 sequence /s), and the scale of the intermediate result is 50GB. Conducting each test for 10 times, and each time for 10 min, the experimental result is the average value. Fig.4. shows the performance compared with those of the recent research RTMR algorithm [13] and the proposed strategy (HTCD). Fig.5. shows memory hit rate performance compared with those of the LRU algorithm (Least Recently Used), RTMR algorithm and the proposed strategy (HTCD). The memory read/write performance is improved by 8.72% in the proposed strategy. The external storage read/write and entirety read/write performance are improved by 19.2% and 3.79% respectively. The memory hit rate performance is improved by 10.9% and 21.3% compared with LRU and RTMR.

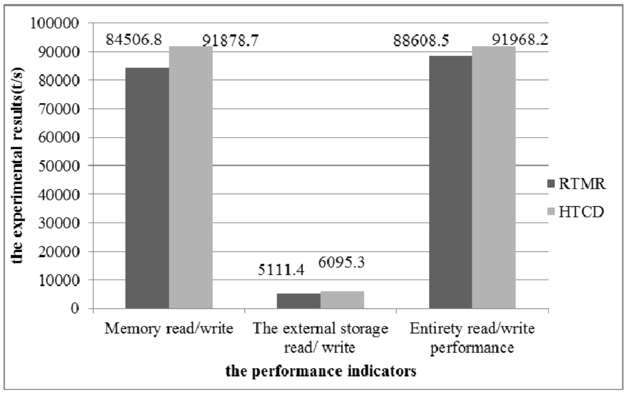

Fig. 4. Thet performance compared with RTMR and HTCD

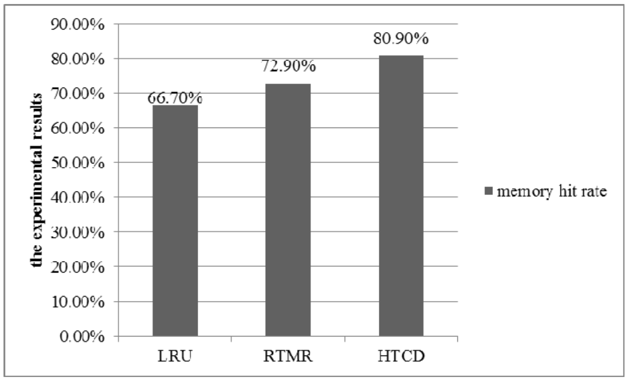

Fig. 5. The memory hit rate results compared with LRU, RTMR and HTCD

V. Conclution

This paper proposes and describes a data streams processing strategy based on Hoeffding tree with concept drift on Hadoop environment. The newly proposed method can restructure the data classification model when some categories appeared the concept drift. Furthermore, the Map-Reduce Parallel Processing Strategy of Massive Sensor Data Streams is presented for improving the real-time processing performance. The proposed design project is an efficient strategy for the massive sensor data streams analysis and mining. However, the proposed strategy spends a great deal of time with attributes that have nearly identical splitting quality. In addition, the memory utilization can be further optimized.

Acknowledgment

The research work was supported by the National Natural Science Foundation of China under Grant No. 61403069 and 61473066, the Natural Science Foundation of Hebei Province under Grant No. F2014501055, the Program of Science and Technology Research of Hebei University No. ZD20132003, the Fundamental Research Funds of the Central University under Grant no. N120323009, and the Fund of Northeastern University at Qinhuangdao under Grant No. XNK201402.

References

[1] R. Giachetta, “A framework for processing large scale geospatial and remote sensing data in MapReduce environment,” Computers &Graphics, vol.49, pp.37-46, June 2015.

[2] B. Yu, R. Sen, D. H. Jeong, “An integrated framework for managing sensor data uncertainty using cloud computing, ” Information Systems, vol.38 no.8, pp.1252-1268, 2013.

[3] M. Smit, B. Simmons, M. Litoiu, “Distributed, Application-level monitoring for hetero- geneous clouds using stream processing,” Future Generation Computer Systems, vol.29 no.8 , pp2103-2114, 2013

[4] C. Kaiser, A. Pozdnoukhov, “Enabling real-time city sensing with Kernel Stream Oracles and MapReduce,” Pervasive and Mobile Computing,vol. 9, pp.708-721, 2013.

[5] C. Feng, Y.M. Wen, L.B. Tang, “Algorithm of recurring concept drift based on main feature extraction,” Journal of Data Acquistition and Processing, vol.31 no.2, pp.315-324, March 2016.

[6] N. Li, G.D.Guo, “Ensemble classification algorithm for high speed data stream,” Journal of Computer Application,vol.32 no.3, pp.629-633, 2012.

[7] C.W. Li, K.F.Jea, “An approach of support approximation to discover frequent patterns from concept-drifting data streams based on concept learning,” Knowledge and Information Systems, vol.40 no.3, pp.639-671, September 2014.

[8] G.S. Zhao, Z.J. Ba, J. H. Du, “Resource constrained data stream clustering with concept drifting for processing sensor data, ” International Journal of Data Warehousing And Mining ,vol.11 no. 3, pp. 49-67,July-September 2015.

[9] H. Yang, F. Simon, “Countering the concept-drift problems in big data by an incrementally optimized stream mining model,” Journal of Systems And Software ,vol.102,pp. 158-166, April 2015.

[10] G. Song, Y.M. Ye, H.J. Zhang, et al. “Dynamic clustering forest: An ensemble framework to efficiently classity textual data stream with concept drift,” Information Sciences, vol.357, pp.125-143, August 2016.

[11] T.S. Sethi, M. Kantardzic, H.Q. Hu, “a grid density based framework for classifying streaming data in the presence of concept drift,” Journal of Intelligent Information Systems , vol.46 no. 1, pp. 179-211, February 2016.

[12] Y.G. Sun, Z.H. Wang, H.Y. Liu, et al.“Online ensemble using adaptive windowing for data streams with ConceptDrift.,” International Journal of Distributed Sensor Networks , article ID:4218973, 2016.

[13] K. Y. Qi, Z. F. Zhao, J. Fang, Q. Ma, “Real-Time Processing for High Speed Data Stream over Large Scale Data,” Chinese Journal of Computers, vol.35, no.3, pp.477-490, Mar. 2012.