RESEARCH ARTICLE OPEN ACCESS

iOVFDT for Concept-drift Problem in Big Data

Rajeswari P. V. N [1], Radhika P. [2]
Assoc. Professor [1], Dept of CSE, Visvodaya Enginnering College, Kavali
Asst. Professor [2], Dept of CSE, Geethanjali Institute of Science & Technology, Nellore
AP – India.

ABSTRACT

The problem of How to efficiently uncover the knowledge hidden within massive and big data remains an open problem. It is one of the challenges is the issue of 'concept drift' in streaming data flows. Concept drift is a well-known problem in data analytics, in which the statistical properties of the attributes and their target classes shift over time, making the trained model less accurate. Many methods have been proposed for data mining in batch mode. Stream mining represents a new generation of data mining techniques, in which the model is updated in one pass whenever new data arrive. This one-pass mechanism is inherently adaptive and hence potentially more robust than its predecessors in handling concept drift in data streams. In this paper, we evaluate the performance of a family of decision-tree-based data stream mining algorithms. The advantage of incremental decision tree learning is the set of rules that can be extracted from the induced model. The extracted rules, in the form of predicate logics, can be used subsequently in many decision-support applications. However, the induced decision tree must be both accurate and compact, even in the presence of concept drift. We compare the performance of three typical incremental decision tree algorithms (VFDT [2], ADWIN [3], iOVFDT [4]) in dealing with concept-drift data. Both synthetic and real-world drift data are used in the experiment. iOVFDT is found to produce superior results.

Keywords:- Data Stream Mining; Concept Drift; Incremental Decision Tree; Classification.

I. INTRODUCTION

Big data has become a hot research topic, and how to mine valuable information from such huge volumes of data remains an open problem. Many research institutes worldwide have dedicated themselves to solving this problem. The solutions differ from traditional data mining methods, where the mining process must be efficient and incremental.

Processing big data presents a challenge to existing computation platforms and hardware. However, according to Moore's Law, CPU hardware may no longer present a bottleneck in mining big data due to the rapid development of integrated circuit industry. A well-designed algorithm is crucial in solving problems associated with big data.

A data stream model is usually defined as a model in which data move continuously at high-speed. Most big data can be considered as data streams, in which new data are generated continuously. Data streams contain very large volumes of data, which cannot be stored in either internal or external memory. A one-pass algorithm therefore forms the basis of data stream mining, which briefly stores a sufficient statistical matrix when new data passes, but does not require the full dataset to be scanned repeatedly.

A data stream also depicts an infinite big data scenario in which the underlying data distribution of newly arriving data may differ from older data in the real world: the so-called concept-drift problem. For example, click-streams of users' navigation patterns on an e-commerce website may reflect their purchase preferences as analyzed by the system. However, as people's preferences for products change over time, the old model is no longer applicable, resulting in concept drift.

Decision trees are one of the most important data classification techniques. These techniques are widely used because of their ability to interpret knowledge in different domains and present it as a tree-like graph. Decision trees can be distinguished into two categories according to their components: single-tree algorithms and multi-tree algorithms. A single-tree algorithm is lightweight and easy to implement and thus favored for data stream environments, although in some cases, a multi-tree algorithm may achieve slightly higher accuracy.

DATA STREAMS MINING PROBLEMS

To store continuous data stream is a great challenge for storage devices. To generate pattern or knowledge from stream data, algorithms with different techniques are needed. We don"t have enough amount of space to store stream data and problem occurs between accuracy of data pattern and storage. So we can classify into five categories as shown in table I. [10].

Table I. Classification of Challenges via Category

No	Issues	Challenges	Solution approach for these issues
1	Memory manage ment	Data arrival rate and variant data arrival rate over time are irregular and fluctuated	Summarization techniques
2	Data Pre-processin g	Quality of mining result and automation of pre-processing	Light-weight pre-processing technique
3	Data structure	Limited memory size and large volume of data stream	Incremental maintaining of data structure, novel indexing, storage and querying techniques
4	Resource	Limited resource like storage and computation capabilities	AOG
5	Visualiza tion of results	Problem in data analysis and quick decision making by user	Still is a research issue(one of the proposed approach is: intelligent monitoring)

In this paper, we investigate the performance of single-tree learning for concept-drift data streams. Three representative tree inductions are used in this evaluation: VFDT [2], a classic algorithm that pioneered the use of Hoeffding bound to build an incremental decision tree; ADWIN [3], a start-of-the-art tree model that uses an adaptive-window technique to handle concept drift; and iOVFDT [4,9], a model previously developed by the present authors that balances accuracy, tree size and learning speed.

The results show that iOVFDT has good performance for both synthetic and real-world concept-drift data streams. The advantage of the adaptive tie threshold makes iOVFDT suitable for real-world applications.

The paper is organized as follows: Section 1 introduces the research topic; Section 2 reviews some related work; Section 3 presents the preconditions for the evaluation, including the platform, data sources and measurements; Section 4 analyzes the evaluation results and discusses the comparison; and Section 5 concludes the paper.

II. INCREMENTALLY OPTIMIZED DECISION TREE (IOVFDT) [11]

For noisy big data, a new decision tree induction proposes to use a multi-objective function to balance prediction accuracy, tree size and learning speed. New methods of functional tree leaf improve accuracy. Besides, intuitive graph visualizes tree structure dynamically for massive data analysis.

A. Introduction

How to extract meaningful information from big data has been a popular open problem. Decision tree, which has a high degree of knowledge interpretation, has been favored in many real world applications. However noisy values commonly exist in high-speed data streams, e.g. real-time online data feeds that are prone to interference. When processing big data, it is hard to implement pre-processing and sampling in full batches. To solve this trade-off, we propose a new decision tree so called incrementally optimized very fast decision tree (iOVFDT). Inheriting the use of Hoeffding bound in VFDT algorithm for nodesplitting check, it contains four optional strategies of functional tree leaf, which improve the classifying accuracy. In addition, a multi-objective

incremental optimization mechanism investigates a balance amongst accuracy, mode size and learning speed. iOVFDT is extension that can be integrated with the latest version of MOA. Besides, iOVFDT has a high extendibility that is able to combine with most VFDT-extended algorithms.

B. Implementation Platform

Massive Online Analysis (MOA) is a framework for datastream mining. It includes a collection of machine learning algorithms (classification, regression, and clustering) and tools for evaluation. Related to the WEKA project, MOA is also written in Java, while scaling to more demanding problems. In assification part, MOA has simulated decision tree algorithms that can be evaluated by built-in measurements. The well-defined experimental platform implements in two modes: graphic interface and command line. iOVFDT aims to train a decision tree with minimum error from big data, even if the data contain imperfect quality like noise and bias data. The incremental decision tree algorithm that inherits the use of Hoeffding bound in VFDT. Besides, four types of functional tree leaf are proposed in iOVFDT package, improving classifying accuracy. Suppose *nijk* is the sufficient statistic that reflects the number of attribute *Xi* with a value *xij* belonging to class *yk*. *i ,j ,k* are the index of attribute *X*, value of attribute *Xi* and class *y* respectively.

***Majority Class* functional leaf:**

$$\arg\max k = \{n_{ij1} \quad \cdots \quad n_{ijk} \quad \cdots \quad n_{ijK}\}$$

***Naïve Bayes* functional leaf:**

$$p_{ijk} = \frac{\mathrm{P}(x_{ij}|y_k)\cdot\mathrm{P}(y_k)}{\mathrm{P}(x_{ij})}, \quad \arg\max k = \{p_{ij1} \quad \cdots \quad p_{ijk} \quad \cdots \quad p_{ijK}\}$$

***Weighted Naïve Bayes* functional leaf:**

$$p_{ijk} = \omega_{ijk}\frac{\mathrm{P}(x_{ij}|y_k)\cdot\mathrm{P}(y_k)}{\mathrm{P}(x_{ij})} \quad where \quad \omega_{ijk} = \frac{n_{ijk}}{\sum_{k=1}^{K} n_{ijk}}$$

$$\arg\max k = \{p_{ij1} \quad \cdots \quad p_{ijk} \quad \cdots \quad p_{ijK}\}$$

***Error-adaptive* functional leaf:**

$$\arg\min \mathrm{F} = \{Err(\mathrm{F}^{MC}, y_k), \quad Err(\mathrm{F}^{NB}, y_k), \quad Err(\mathrm{F}^{WNB}, y_k)\}$$

FMC, *FNB* and *FWNB* require memory proportional to *O (N·I·J·K)*, where *N* is the number of nodes in tree model, *I* the number of attributes; *J* is the maximum number of values per attribute; *K* is the number of classes. *FNB* and *FWNB* are converted from that of *FMC*. So we don't require extra memory for *FEA* respectively. When required, it can be converted from *FM*.

C Extension of MOA(Massive Online Analysis):

Extension of MOA platform, iOVFDT package supports both GUI and command-line mode. What's more, this package adds new functions of ROC statistics and tree visualization to improve the experimental tasks.

1

iOVFDT applies a multi-objective optimization model to control the node-splitting. After normalizing those three dimensions, the area of this triangle model is defined as Φ(*TRt)* , where *TRt* is the decision tree structure at timestamp *t*. The range of Φ*(TRt)* is within a min-max model that ensures the variances of statistics mean and true mean isn't too big to maintain, where Min.Φ(*TRt)*<Φ(*TRt*)*<Max.Φ(*TRt)*. If Φ(*TRt)* goes beyond this constraint, the existing model is not suitable to embrace new data that the algorithm should be updated. Therefore, the nodesplitting condition is adaptively optimized that: *ΔH(Xi)>HB* or Φ(*TRt)>Max.*Φ(*TRt)* or Φ(*TRt)<Min.*Φ(*TRt)*.

iOVFDT Package Built-in MOA

After downloading iOVFDT package (*iovfdt.jar*) and required MOA packages (*moa.jar* and *sizeofag.jar*), GUI can be run by typing the command in console:

java -cp iovfdt.jar:moa.jar -javaagent:sizeofag.jar moa.gui.GUI

Three new components are included in *iovfdt.jar*:

- Family of iOVFDT algorithm (four types of optional functional tree leaf)
- Model Evaluation Method (with ROC statistics and tree structure buffer output)
- Decision Tree Visualization (by *prefuse.jar* open source visualization tool)

Example 1a: evaluate model by GUI mode

1. Configure the task as EvaluateModel_ROC;
2. Select iOVFDT as the learning method;
3. Select the training data *nursery_train.arff* and testing data *nursery_test.arff*;
4. Select the location to save tree model buffer to *IOVFDT_SampleTree.txt*;
5. Output the result to *IOVFDT_MOA_2012.txt*;
6. Press button "RUN".

The related output results are shown in Fig 1 and Fig 2.

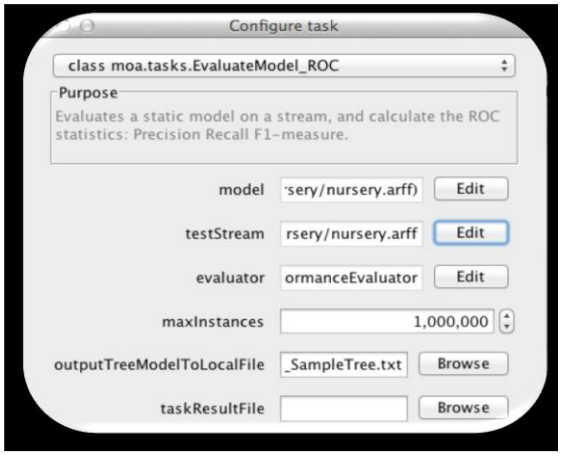

Fig1: Configuration task

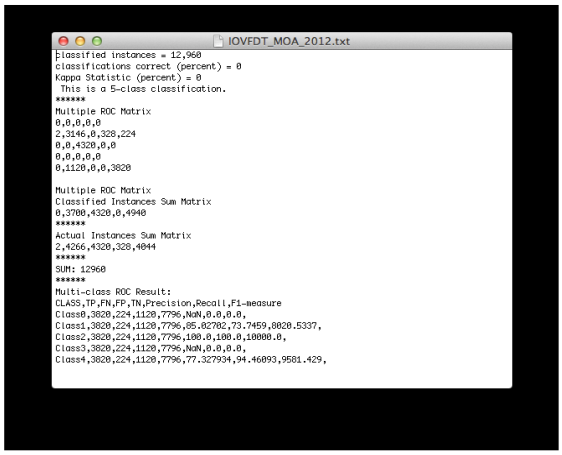

Fig 2: Classifying process

Example 1b: evaluate model by command-line mode

- java –cp iovfdt.jar:moa.jar –javaagent:sizeofag.jar moa.DoTask "EvaluateModel_ROC
- -m (LearnModel -l iOVFDT -s
- (ArffFileStream -f
- /Users/data/uci_nursery/nursery_train.arff)) -s (ArffFileStream -f
- /Users/data/uci_nursery/nursery_test.arff) -T /Users/IOVFDT_SampleTree.txt" >
- "/Users/IOVFDT_MOA_2012.txt".

The related output results are shown in Fig 3, Fig 4 and Fig 5.

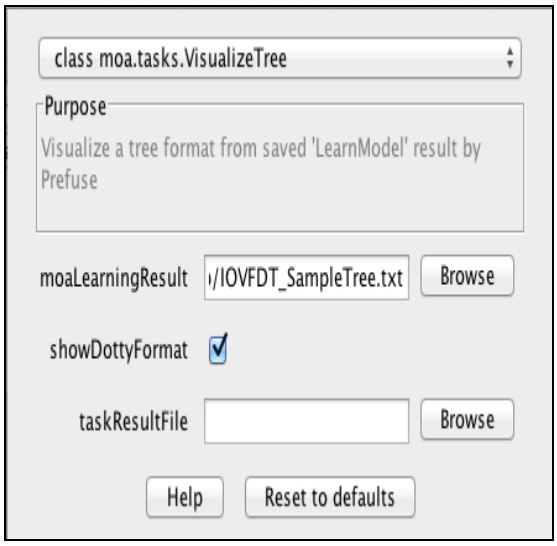

Fig 3: Visualization of Decision Tree

Example 2: visualize decision tree

1. Configure the task as VisualizeTree;
2. Select the saved tree buffer *IOVFDT_SampleTree.txt*;
3. *Optionally: show dotty format converting;
4. Press button "RUN".

Integration With Other VFDT-extended Algorithms

In source code part, we write comments for each place of modification based on *HoeffdingTree.java*. Generally, seven-part modifications are proposed in *iOVFDT.java*. In each of them, it includes some new class, variables and functions designed for iOVFDT algorithm. When you want to integrate it to other extension of decision tree that uses Hoeffding bound as node-splitting criteria, just add these seven modifications to appropriate places in source codes. It is very easy.

Fig 4: Converted Dotty format

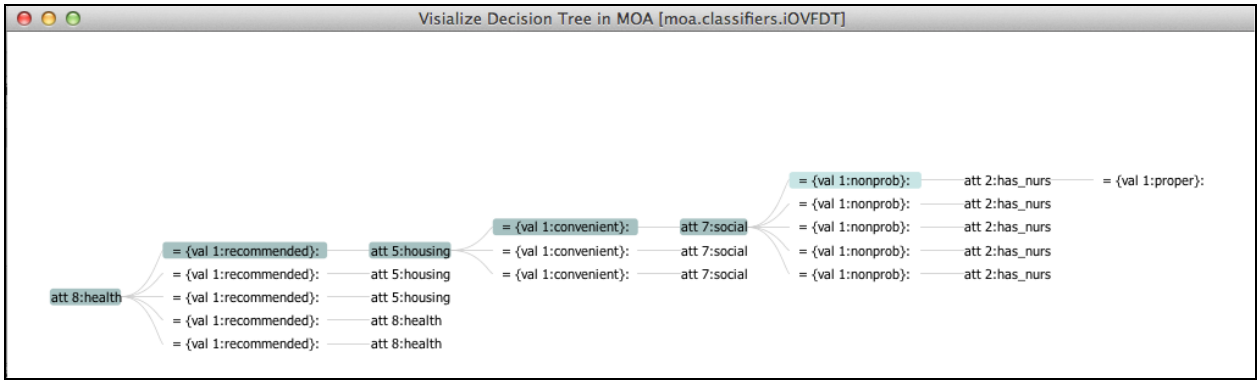

Fig 5: Exploring Decision Tree

III. EVALUATION COMPARISON WITH OTHER ALGORITHMS

In this evaluation, we focus on the one-pass learning performance of various incremental decision trees, in which the algorithm learns and updates the tree model as each new sample arrives. This learning approach is suitable for large volumes of data, or even infinite data. We use both synthetic and real-world data in this evolution. We can configure the level of concept drift in the synthetic data, while we have known concept drift in the real-world data. The concept drift is visualized by a feature selection method that shows the ranked relationship of each attribute in a class.

A. Datasets and Platform

In the real world, big data may comprise millions of instances that arrive at high speed. To simulate a big data environment, MOA [1] provides a platform in which data arrives continuously. The running environment is a Windows 7 PC with an Intel 2.8 GHz CPU and 8 G RAM.

We simulate the *Waveform* data using the MOA Data Generator. The data comprise the records collected from distributed sensors containing 9 numeric attributes, and three types of waveforms as the targeted class. The number of instances is 200,000. We configure the drifting percentage as 0% (no attribute drifts), 25% (5 of 9 attributes drift), and 50% (10 of 9 attributes drift).

The real-world data are from UCI machine learning. *Cover Type* data were released in 1999, comprising forest land inventory data for natural eco-system management. Forty-two categorical attributes and 12 continuous attributes are used in the two predictive models to predict seven different cover lands. This is an open dataset for analyzing the concept-drift problem.

B. Evaluated Algorithms

Three typical incremental decision-tree algorithms are tested: VFDT[2], ADWIN[3], and iOVFDT[4].

VFDT pioneered the use of HB for node splitting, but has no criteria for handling concept drift. Its improved version, CVFDT [5], uses a fixed sliding-

window technique with a user pre-defined tie-breaking threshold to deal with concept drift instead of a fixed window size.

ADWIN provides an adaptive-window solution for the concept-drift problem. As ADWIN performs as well as or only slightly worse than the best window for each rate of change in CVFDT [3], CVFDT was not compared in this test.

iOVFDT uses an adaptive tie-breaking threshold and an optimized node-splitting condition. It supervises the model error using a min-max model that reflects the concept drift when a leaf splits into an internal node.

Functional Tree Leaf (FTL) [6] is an improvement for these algorithms, and it is claimed that the error-adaptive FTL obtains better performance than the majority class and naïve Bayes algorithms [1,4]. Hence, we only test the tree inductions with error-adaptive FTL in this paper.

VFDT and ADWIN require a tie-breaking threshold setup, where $K \in (0,1)$. The configurations for VFDT, ADWIN and iOVFDT are δ = 10-6 and *nmin* = 200.

C. Meansurement

To evaluate the performance of the algorithms we use some common measurements for decision tree learning, as listed in Table II.

TABLE II.MEASUREMENTS

Measure	Specification
Accuracy	The accuracy of classification: #Correctly Classified /#Total InstancesX100%
Kappa	A measure of classifier performance in unbalanced class streams [7].
#Node	The number of internal nodes.
#Leaf	The number of leaves. A rule is a branch from the root to a leaf in the expression of a set of if-then-else rules. #Leaf indicates how many rules are included in a tree mode.
Depth	The number of nodes that exist in the longest path from the root to a leaf.
MemKB	The memory (KB) used for tree building.
Time	Model training time in seconds. It also reflects the learning speed of an algorithm.

IV. RESULTS AND DISCUSSION

A. Synthetic Waveform Concept-drift Data Streams

First, we examine the negative effect of concept drift on decision tree learning. VFDT, with the default value of K, is applied to the waveform data with different percentages of concept drift. Figure 6 shows the test results implemented every 10,000 samples. The figure clearly shows the effect of the concept-drift problem on the accuracy of classification. In Figure 2, we zoom-in on the tests from the 10^4 to the 10^2 samples and drift-down to the samples between the $10{,}000^{th}$ and the $20{,}000^{th}$ data records. The accuracy clearly fluctuates when concept drift exists in the data streams. Zooming in reveals some details, such as a crossover between the 25% drift and 50% drift in Figure 7. However, the overall trend shows that concept drift gradually reduces the accuracy.

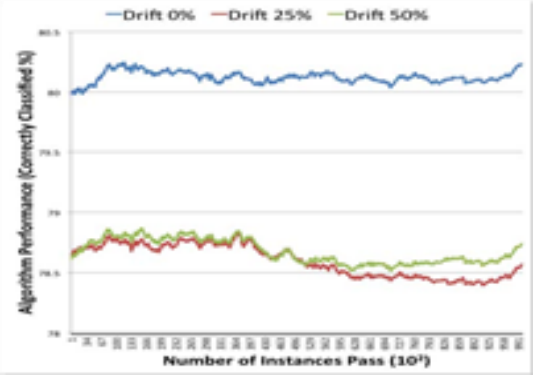

Figure 6. Negative Effect of Concept Drift on Accuracy.

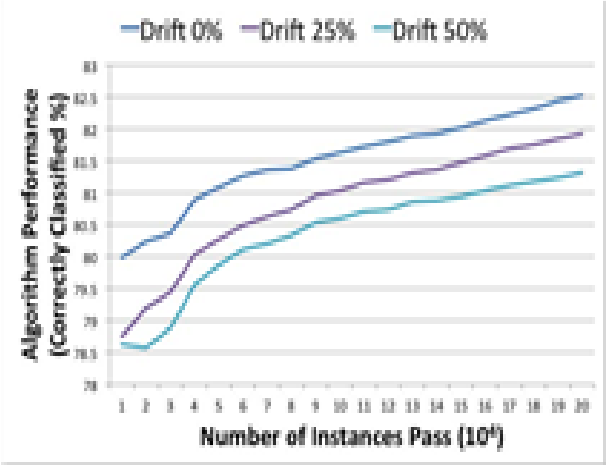

Figure 7. Zoom-in on the Negative Effect of Concept Drift on Accuracy.

Second, we illustrate the performance in terms of accuracy and changing tree size as new data arrive. The results are visualized in Figure 5. Due to the page limitation, the figures are attached at the end of the paper. We use K = 0.05 (a small value and the default setup of MOA), K = 0.50 (the mean value), and K = 0.95 (a large value) in this test. We can see that in general, iOVFDT obtains better classification accuracy than ADWIN and VFDT. The tree size is stable in iOVFDT but varies with different K in ADWIN and VFDT. When K is configured as a small value, the tree sizes of ADWIN and VFDT are smaller than that of iOVFDT, but in the other cases, iOVFDT produces a more compact tree size.

Third, we assess the accuracy, tree structure and memory cost when different K values are applied in VFDT and ADWIN. However, we do not know which value of K is the best until all possible values have been tried. This is not practical in real-time applications. In this test, we use K = 0.05, 0.10, 0.15,…, 0.95. In addition to accuracy, the Kappa statistic [7] and the structural features of the decision tree model, such as the number of nodes, number of leaves and tree depth, are important for evaluating the models. Each path from the root to a leaf represents a rule in the decision tree model, thus the number of leaves reflects the number of patterns in the tree model. In addition, the amount of memory consumed reflects the computation cost of a tree-learning algorithm in the MOA platform. For VFDT and ADWIN, we show the average result for different K values in Table III. iOVFDT generally outperforms VFDT and ADWIN in this test. We use the iOVFDT result as the benchmark, and Table IV shows the improvement compared to VFDT and ADWIN. In this table, we find that iOVFDT improves accuracy by 1-2%, and improves the tree size by more than 15% and 35% compared to VFDT and ADWIN. In addition, iOVFDT consumes less memory and thus reduces the computational cost.

Drift%	Method	Acc%	Kap%	#Node	#Leaf	Depth	MemKB
0	iOVFDT	82.61	73.92	1029	515	23	6,115
	VFDT	81.78	72.67	1229	615	18	7,358
	ADWIN	81.05	71.58	1651	825	22	11,550
25	iOVFDT	81.24	71.87	1057	529	19	5,857
	VFDT	80.73	71.10	1248	625	17	6,918
	ADWIN	80.03	70.04	1647	824	21	10,469
50	iOVFDT	81.33	71.99	1049	525	18	5,886
	VFDT	80.77	71.16	1252	626	16	7,007
	ADWIN	80.07	70.10	1650	825	21	10,807

TABLE III.COMPARISON OF AVERAGE PERFORMANCE OF WAVEFORM

TABLE IV.IOVFDT IMPROVEMENT SUMMARY

Drift%	Method	Acc%	Kap%	#Node	#Leaf	Depth	MemKB
0	VFDT	1%	2%	-16%	-16%	26%	-17%
	ADWIN	2%	3%	-38%	-38%	5%	-47%
25	VFDT	1%	1%	-15%	-15%	12%	-15%
	ADWIN	2%	3%	-36%	-36%	-10%	-44%
50	VFDT	1%	1%	-16%	-16%	10%	-16%
	ADWIN	2%	3%	-36%	-36%	-16%	-46%

V. CONCLUSION

Advances in technology have resulted in the development of new methodologies specifically designed for handling big data problems. Although improvements in computer hardware are fundamental to such developments, a flexible way of solving the hardware bottleneck lies in the improvement of software algorithms. Stream mining is intended to tackle high-speed and changing data on the fly. The one-pass process makes it possible to handle massive and big data, and even infinite data.

A decision tree is an important classification model, which turns the output of mining results into useful intelligence in the form of a tree-like graph. In the past decade, incremental decision trees have become popular for solving big data problems. There are two major types: single-tree algorithms and multi-tree algorithms.

In our previous work, we developed iOVFDT [4, 9], a new incremental tree induction with an optimized node-splitting mechanism for adaptive learning. In this paper, we investigate the phenomenon of concept drift using three representative single-tree induction algorithms.

The evaluation results show that iOVFDT obtains good accuracy with a compact model size and less use of memory.

REFERENCES

[1] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: Massive Online Analysis," Journal of Machine Learning Research, 11, pp. 1601-1604, 2010.

[2] P. Domingos and G. Hulten, "Mining high-speed data streams," in Proc. of the 6th ACM SIGKDD, pp. 71-80, 2000.

[3] A. Bifet and R. Gavalda刱"Learning from time-changing data with adaptive windowing," in Proc.

of the SIAM International Conference on Data Mining, pp. 443–448, 2007.

[4] H.Yang and S.Fong, "Incrementally optimized decision tree for noisy big data," in Proc. of the BigMine Workshop of the 18th ACM SIGKDD, pp. 36-44, 2012.

[5] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in Proc. of the 7th ACM SIGKDD, pp. 97-106, 2001.

[6] J. Gama, R. Rocha and P. Medas, "Accurate decision trees for mining high-speed data streams," in Proc. of the 9th ACM SIGKDD, pp. 523-528, 2003.

[7] C. Jean, "Assessing agreement on classification tasks: The kappa statistic." Computational Linguistics, 22(2), pp. 249–254, 1996.

[8] B. Kevin, Pratt and G. Tschapek, "Visualizing concept drift," in Proc. of the 9th ACM SIGKDD. pp. 735-740, 2003.

[9] H. Yang, S. Fong, Incremental Optimization Mechanism for Constructing A Decision Tree in Data Stream Mining, International Journal of Mathematical Problems in Engineering, Hindawi press, 2013, Ready for publishing. DOI:10.1155/2013/580397

[10] Aggarwal, C., Han, J., Wang, J., and Yu, P.S., (2004): On Demand Classification of Data Streams. In Proceedings of 2004 International Conference On Knowledge Discovery and Data Mining (KDD '04). Seattle, WA.

[11] Hang Yang and Simon Fong, Incrementally Optimized Decision Tree & Tree Explorer, Version 1.0.