Ensemble based Data Stream Mining with Recalling and Forgetting Mechanisms

Yanhuang Jiang[1], Qiangli Zhao[2], Yutong Lu[1]

(State Key Laboratory of High Performance Computing, National University of Defense Technology, Changsha, 410073, China)[1]
(School of Computer and Information Engineering, Hunan University of Commerce, Changsha, 410205, China)[2]

***Abstract*—Using ensemble of classifiers on sequential chunks of training instances is a popular strategy for data stream mining. Aiming at the limitations of the existing approaches, we introduce recalling and forgetting mechanisms into ensemble based data stream mining, and put forward a new algorithm MAE (Memorizing based Adaptive Ensemble) to mine chunk-based data streams with concept drifts. Ensemble pruning is used as a recalling mechanism to select useful component classifiers for each incoming data chunk. Ebbinghaus forgetting curve is adopted as a forgetting mechanism to evaluate and replace the component classifiers in the memory repository. Experiments have been performed on datasets with different types of concept drifts. Compared with traditional ensemble approaches, the results show that MAE is a good algorithm with high and stable accuracy, less predicting time and moderate training time.**

Keywords- *data stream mining; Ebbinghaus forgetting curve; recalling mechanism; ensemble pruning*

I. INTRODUCTION

With the development of Internet and sensor techniques, more and more data are collected. Many new applications require the learning algorithms to work under dynamic environments, where data are come continuously with high speed as data streams [1]. Examples of such applications include social media mining, web log analysis, network stream monitor, spam categorization etc. These data streams are often characterized by huge volumes of instances, rapid arrival rate and drifting concept [1,2]. The learning system must have the mechanism to adapt to the recent data in order to provide continuous high predicting performance.

Adaptive ensemble is an important strategy for data stream mining [3-8]. It generates component classifiers sequentially from fixed-size blocks of training instances called data chunks. When a new data chunk arrives, existing components are evaluated and a new classifier is generated. SEA (Stream Ensemble Algorithm) [3] is the first method for creating adaptive ensembles, which uses C4.5 to generate component classifiers. When an ensemble reaches to a given size, the new classifier will replace the worst component of the ensemble. AWE (Accuracy Weighted Ensemble) [4] is also among the most representative methods, in which each component is associated with a weight, and the component with the lowest weight is discarded when component replacement is needed. ACE (Adaptive Classifiers Ensemble) [5] introduces a drift detector in its learning system, and new classifiers will be generated when concept drifts are detected. To achieve high accuracy, no pruning or replacement mechanism is provided in ACE, which always makes the ensemble clumsy after several periods of learning. Learn++.NSE [6] is also a chunk-based ensemble algorithm in which no pruning is used to limit the number of component classifiers. This makes Learn++.NSE requiring much memory and testing time. AUE (Accuracy Updated Ensemble) [7,8] is designed to use incremental algorithms to create component classifiers instead of static batch learners, so it can well adapt to gradual drifts as well as sudden drifts. While the capability of incremental learning limits the usable algorithms for component classifier generation.

All existing ensemble-based approaches for data stream mining have the following limitations: (1) the evaluation and replacement of component classifiers is taken place on the basis of the most recent data chunk. For some concept drifts, such as recurring drifts, the weakest component for current data chunk may not mean it is useless for the future. Thus, removing the weakest component for current data chunk is not a good idea, especially if the component is proved to be useful by many prior data chunks; (2) to predict unknown data, most ensembles use all component classifiers to derive the final result. According to research achievements of ensemble pruning [9-12], in most cases, some components of an ensemble may have no contributions or even worse effects on prediction tasks, which will degrade the performance and efficiency of the ensembles. In the context of concept drifts, using these bad components for prediction may result in low accuracy and slow responding to drifts

This paper studies on new techniques to overcome the above limitations. Inspired by human memorizing functions, we propose a new ensemble-based algorithm, MAE (Memorizing based Adaptive Ensemble), for data steam mining. In this algorithm, we look upon each component classifier as a piece of knowledge. Each component classifier is associated with a value of memory retention, which reflects the usefulness of a component classifier in the system. The classifiers which are still memorized by the system are stored in a memory repository. When a new data chunk comes, the component classifiers related with the data chunk will be recalled and their memory retention will be increased. The classifier with the least memory retention will be forgotten when the memory repository is oversized. MAE uses Ebbinghaus forgetting curve to derive the memory retention for each component classifier, and setups an ensemble pruning mechanism to get a smaller target ensemble for prediction

978-1-4799-5148-2/14/$31.00 ©2014 IEEE

tasks. Experiments results show that, compared with other three approaches, MAE achieves better predictive accuracy and less testing time with moderate training time.

The remainder of this paper is organized as follows. Section 2 proposes the idea of memorizing based adaptive ensemble. Section 3 gives a detailed description of MAE algorithm. Experimental results and analysis are shown in Section 4. Finally, we draw conclusions and discuss future works.

II. IDEA OF MEMORIZING BASED ADAPTIVE ENSEMBLE

A. Human Memorizing Characteristics

As we know, human brain is capable of memorizing knowledge. It can recall the knowledge related with current event, and forget the knowledge which is never used or recalled for a long time. The more a piece of knowledge is recalled, the greater it is apt to be used. Similarly, once a piece of knowledge is learned, the less it is recalled, the greater possibility it will be forgotten.

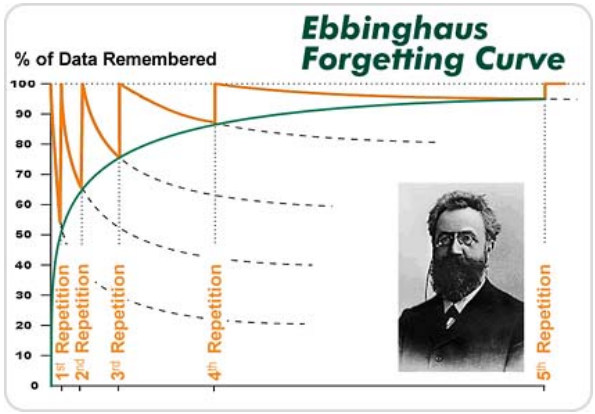

Figure 1. Ebbinghaus Forgetting Curve [13]

Hermann Ebbinghaus is a German psychologist who pioneered the experimental study of memory. He conducted experiments on himself to understand how long the human mind retains information over time [14]. The experimental results can be plotted on a graph what is now known as "Ebbinghaus forgetting curve" (see Figure 1). Ebbinghaus discovered that we rapidly forget the knowledge we have learned and the recall of the learned material over time makes the material to be more stable in memory. This extrapolated the hypothesis of the exponential nature of human forgetting.

B. Memorizing based Adaptive Ensemble

Inspired by the human memorizing function, we proposed a data stream mining algorithm, MAE (Memorizing based Adaptive Ensemble), with recalling and forgetting mechanisms as human beings. In MAE algorithm, Ebbinghaus Forgetting Curve is used to derive the memory retention for component classifiers and forget the least useful classifiers. Ensemble pruning technique is adopted to recall the useful classifiers and generate a smaller target ensemble for prediction tasks.

MAE algorithm has the following characteristics: (1) *Memory repository*: the system has a memory repository with a limited size called memory capacity. All the classifiers which are still memorized by the system are reserved in this repository. (2) *Memory retention*: each classifier has an evaluation value which represents its memory retention in a learning system. The memory retention of a classifier will be decline with the passage of time if it is not recalled. (3) *Learning knowledge*: When a new data chunk comes, a new classifier is generated. This new piece of "knowledge" will be added to the memory repository of the system. (4) *Recalling*: ensemble pruning is performed for each incoming data chunk as a recalling mechanism. The selected component classifiers are the recalled knowledge whose memory retention will be increased. While the pruned components are not eliminated from the memory repository, and they still have the chance to be recalled. (5) *Forgetting*: when the number of component classifiers in the memory repository reaches to the memory capacity of the system, the component with the lowest memory retention will be discarded. The discarded classifiers are forgotten by the system and will never be recalled any more. (6) *Predicting*: when a prediction task is coming, the recalled component classifiers are composed to a target ensemble and used for the prediction. Simple majority voting is adopted to give the final predicting results.

III. MAE ALGORITHM

As traditional chunk-based approaches, MAE also divides a data stream into equally sized data chunks. A new classifier is generated for each new data chunk, and the evaluation of all classifiers is performed after processing all instances from the data chunk. The memory retention of a classifier is decided by two factors: (1) the number of recall times for the classifier; (2) the time interval from the last recall. In MAE algorithm, the memory retention of a classifier c is calculated as follows:

$$w_c = e^{-f_c \cdot (t-\tau_c)} \tag{1}$$

Where w_c denotes the memory retention of classifier c, f_c is the forgetting factor of c, τ_c denotes the time of the last recall for c (or the creating time if c got no recall), and t is the current time. The forgetting factor for a new generated classifier is initialized to an initial forgetting factor α. The forgetting factor of a classifier should be reduced after each recall, which makes the classifier harder to be forgotten. The forgetting factor of classifier c is calculated as follows:

$$f_c = \frac{\alpha}{R_c + 1} \tag{2}$$

Where R_c is the number of recall times for classifier c.

Algorithm I shows the pseudo code of MAE algorithm. When a new data chunk comes, MAE generates a new classifier and adds it to the memory repository *MS*. Then an ensemble pruning process (recalling mechanism) is carried out on all classifiers, which uses the new data chunk as the validation set. The selected classifiers are composed to the target ensemble E_t. In MAE, the classifiers being selected mean they are recalled by the system. The related parameters of the recalled classifiers, such as the number of recall times, the last recall time, and forgetting factors, should be updated.

At last, the forgetting mechanism is applied on the classifiers in the memory repository. Their memory retention

values are updated. If the size of the memory repository is larger than the "memory capacity" m, the classifiers with the lowest weights should be removed from the memory repository, which means it is forgotten by the system.

ALGORITHM I MAE ALGORITHM

Input:
S: data stream of instances
m: memory capacity

(1) Initialization: $MS \leftarrow \emptyset$; $\alpha \leftarrow 1$; $t=0$;
(2) For each data chunk $D_t \subset S$ do
  (2.1) $c \leftarrow$ a new component classifier built on D_t ;
  (2.2) Initialize parameters for classifier c: $w_c \leftarrow 1$; $R_c=0$; $\tau_c=t$; $f_c=\alpha$;
  (2.3) Add c to the memory repository: $MS \leftarrow MS \cup \{c\}$;
  (2.4) E_t=ensemble-pruning (MS, D_t);
  (2.5) **For each** classifier $c_i \subset E_t$
    (2.5.1) $\tau_{ci}=t$;
    (2.5.2) $R_{ci}= R_{ci}+1$;
    (2.5.3) compute the forgetting factor of c_i based on equation (2);
  (2.6) **For each** classifier $c_j \subset MS$
    (2.6.1) update the memory retention of c_j based on equation (1);
  (2.7) **if** $|MS| > m$ **then** remove the classifier with the lowest memory retention from MS;
  (2.8) $t=t+1$;

MAE algorithm adopts n-of-m method for prediction tasks. Instead of using all component classifiers to predict instances as traditional approaches, MAE applies the current target ensemble E_t on prediction tasks. We use simple majority voting to combine the predictive results of all classifiers in E_t to get the final predicting result.

IV. EXPERIMENTAL RESULTS

A. Experimental Setup

We compared MAE algorithm with three traditional ensemble-based algorithms, SEA, AWE and ACE, on 13 large datasets. Ten of them are artificial datasets. We use different generators of MOA framework [15] to generate these artificial datasets, where each of them generates two different datasets. The other three datasets, Elec, Cover and Airlines, are from real applications and publicly available. Table I gives a brief description of each dataset.

In our experiments, the chunk size was set to 500 for all algorithms. A fast BPNN algorithm, RPROP [16], was adopted to generate component classifiers for all tested algorithms. The stop condition for training a BPNN was set to MSE<0.05 or the number of 3 epochs reach to 50. Insufficiently trained BPNNs improve the diversity in classifier ensembles, and which makes the comparison of ensemble methods in improving accuracy be easier.

TABLE I. DESCRIPTION OF DATASETS

Dataset	No. Inst	No. Attrs	No. Cls	Noise	No. Drifts	Drift Type	MOA Generator
Hyp_S	1 M	10	2	5%	1	slow incremental	Hyperplane
Hyp_F	1 M	10	2	5%	1	fast incremental	Hyperplane
RBF_{GR}	1 M	20	4	0%	4	gradual	Radial basis function
RBF_{ND}	1 M	20	2	0%	0	none	Radial basis function
SEA_S	1 M	3	4	10%	3	slow sudden	SEA
SEA_F	2 M	3	4	10%	9	fast sudden	SEA
$Tree_S$	1 M	10	4	0%	4	slow recurring	Random tree
$Tree_F$	100 k	10	6	0%	15	fast recurring	Random tree
LED_M	1 M	24	10	10%	3	mixed	LED
LED_{ND}	10 M	24	10	20%	0	none	LED
Elec	45 k	7	2			unknown	
Cover	581 k	53	7			unknown	
Airlines	539 k	7	2			unknown	

To make the comparison more meaningful, we set the reserved number of component classifiers be the same value m=25 for all compared algorithms since lower values would decrease the accuracy of all tested algorithms. For ACE, we remove the component classifiers with the lowest weight to keep the maximum number of classifiers be 25 instead of keeping all classifiers as the original algorithm in reference [5]. MAE uses a cluster-based ensemble pruning algorithm [9] as the recalling mechanism. All tested algorithms were implemented in C++ language and run on a computer equipped with two 4-core 2.2GHZ Intel processors, 32 GB RAM and Linux operating system.

We evaluate the performance of the tested algorithms from three aspects: predicting accuracy, training time and testing time. In our experiments, all algorithms worked as the test-then-train paradigm for each data chunk. That is, when a new data chunk comes, it is used to test the performance of current target ensemble at first, and then used to generate a new component classifier and update the evaluation values of all component classifiers. The results of chunk predicting accuracy, training time and testing time are the average values of the corresponding results among all data chunks.

B. Predicting Accuracy

Table II lists the average chunk accuracies. The last row lists the mean results of all datasets for each algorithm. From Table II, we can see that MAE outperforms other algorithms on 7 out of 13 datasets on the accuracy results, including 5

artificial datasets and 2 real datasets. MAE also achieves the highest mean result of all datasets. ACE outperforms other algorithms on 5 artificial datasets, and its average result is the third one. AWE achieves the best result on Elec dataset, while its mean result is the lowest.

TABLE II. AVERAGE CHUNK ACCURACIES IN PERCENTAGE [%]

Dataset	SEA	AWE	ACE	MAE
Hyp_S	86.67	85.88	**86.89**	86.75
Hyp_F	87.90	87.09	**88.56**	87.77
RBF_{GR}	54.64	50.48	**56.46**	55.13
RBF_{ND}	72.70	70.57	72.66	**73.27**
SEA_S	81.03	80.11	**82.95**	79.31
SEA_F	86.66	84.69	**87.01**	86.82
$Tree_S$	41.08	35.88	39.78	**44.05**
$Tree_F$	35.73	32.20	32.37	**39.98**
LED_M	47.24	49.36	49.06	**50.10**
LED_{ND}	46.84	46.71	45.99	**47.65**
Elec	62.51	**63.66**	61.36	63.28
Cover	58.95	57.89	56.98	**59.23**
Airlines	63.21	62.74	63.75	**64.51**
Average	63.47	62.10	63.37	**64.45**

Figure 2, Figure 3 and Figure 4 illustrate the accuracy results of corresponding data chunks for three typical datasets, RBF_{ND}, $Tree_S$ and Elec, respectively. In each figure, the vertical axis shows accuracy and the horizontal axis shows the number of processed instances.

RBF_{ND} dataset was created by radial basis function generator. It has two decision classes and no drift. From Figure 2 we can see that the accuracy results of each algorithm are very stable. The reason is that, for an application without concept drifts, all component classifiers are almost the same important for all data chunks, so the weights of component classifiers have little effect on the predictive accuracy. MAE got slightly better results for this dataset possibly because it uses ensemble pruning process to remove some useless classifiers from the target ensemble.

The $Tree_S$ dataset contains four sudden recurring drifts evenly distributed over 1000000 instances. Figure 3 plots the chunk accuracies of all algorithms on this dataset. MAE has quick reaction to the sudden drifts, and achieved the best accuracy on $Tree_S$ dataset. This is possibly due to the recalling and forgetting mechanisms in MAE algorithm which considers the history importance of component classifiers and selects most useful classifiers for prediction when drift occurring.

Electricity (Elec) [17] is one of the most widely used dataset in data stream classification. It consists of energy prices from the electricity market, which were affected by market demand, supply, season, weather, and time of day. Elec contains 45312 instances each described by seven features. This data has a lot of small drifts since the price was varied by many different factors. Figure 4 shows that SEA, AWE and MAE got relatively stable accuracies, while the results of ACE have a lot of wide fluctuations. The possible reason is that the drift detector of ACE is very sensitive to small drifts, and the new component classifier generated for a detected drift may be a classifier with low predicting accuracy.

The above accuracy results have shown that MAE achieves good predicting accuracy for the data streams with different type of drifts or without a drift. Moreover, our experiments also show that MAE is an algorithm with stable predicting ability.

C. Training Time

Table III reports the average training time results in 10^{-2} seconds for all compared algorithms. What is worth mentioning is the training time of Airlines dataset. Though the dataset has only 7 attributes, two of them are discrete attributes with more than 300 values each. This made the trained BPNNs have huge number of nodes since each value of a discrete attribute corresponds to a binary-valued input node in a BPNN. Training big BPNNs consumed large amount of computing time, and the average results of all datasets in the last line of Table III are dominated by the results of Airlines dataset.

TABLE III. CHUNK TRAINING TIME IN 10^{-2} SECONDS (CS)

Dataset	SEA	AWE	ACE	MAE
Hyp_S	4.35	**4.17**	21.82	6.86
Hyp_F	4.73	**4.63**	23.59	7.25
RBF_{GR}	**9.45**	9.54	37.14	11.76
RBF_{ND}	5.81	**5.76**	24.46	7.69
SEA_S	**1.84**	1.99	10.71	4.31
SEA_F	**1.83**	1.98	10.52	4.20
$Tree_S$	**9.26**	9.34	77.78	10.10
$Tree_F$	**9.37**	9.44	79.84	10.33
LED_M	**16.34**	16.46	124.72	17.94
LED_{ND}	**17.74**	17.97	138.71	19.13
Elec	4.31	**4.28**	35.73	5.51
Cover	**20.19**	20.42	201.35	22.32
Airlines	**1080.38**	1089.93	1462.35	1097.82
Average	**91.20**	91.99	172.98	94.25

Table III shows that SEA is the fastest algorithm, followed by AWE. MAE requires moderate training time, while ACE is a time consuming algorithm relatively. Considering that generating component classifiers consumes almost the same amount time for all compared algorithms, the differences between the training time results of the compared algorithms come from the different drifts processing methods in these algorithms. SEA and AWE are the algorithms with fast training process because both of them use very simple methods to evaluate the component classifiers. MAE algorithm uses recalling and forgetting mechanisms to evaluate component classifiers in the experiments, which

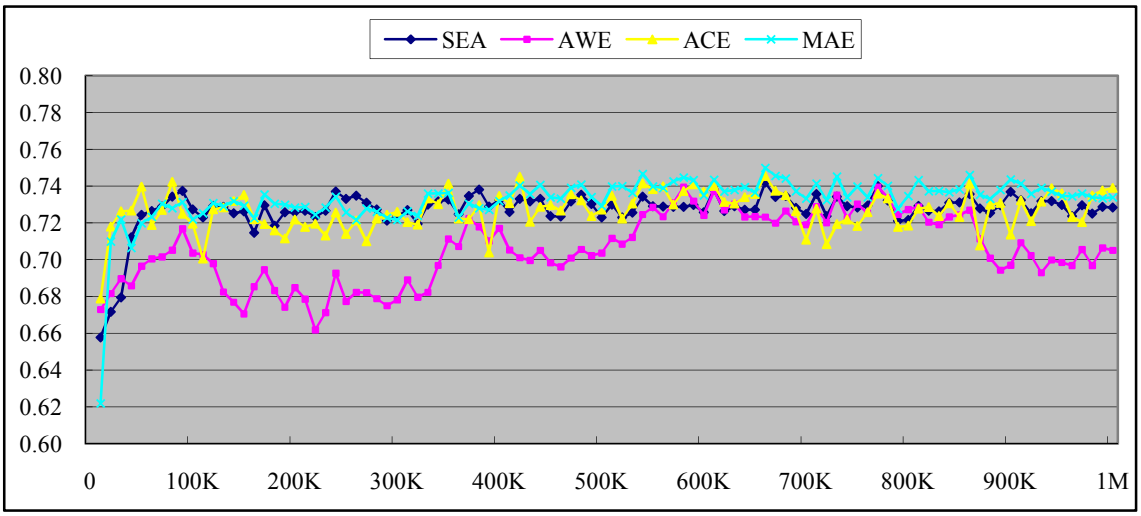

Figure 2: Average chunk accuracy results on RBF_{ND} dataset

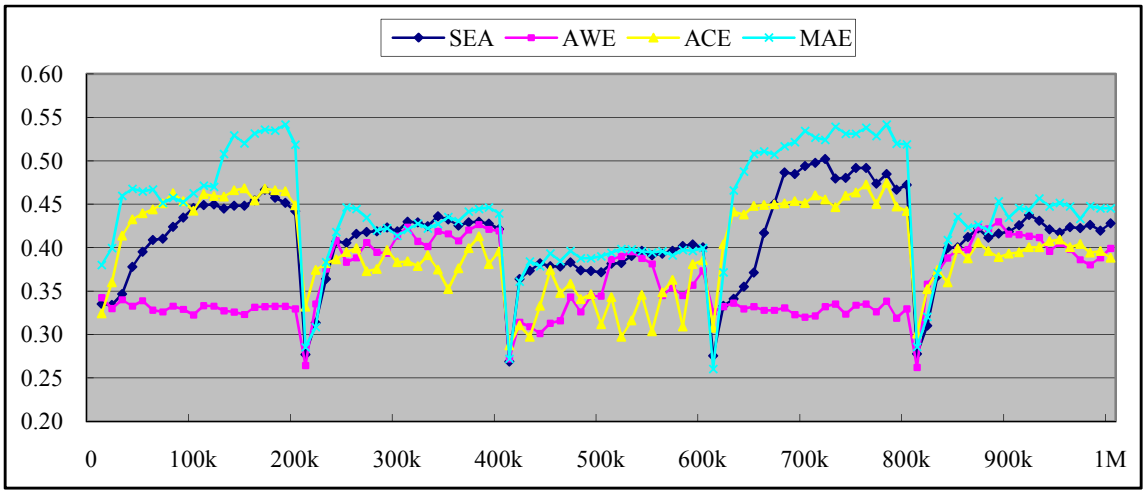

Figure 3: Average chunk accuracy results $Tree_S$ dataset

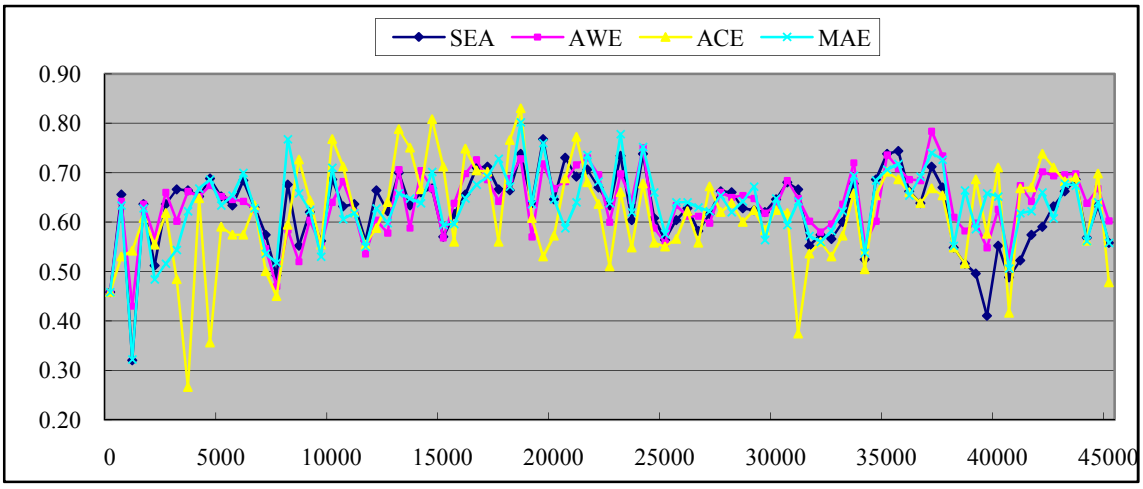

Figure 4: Average chunk accuracy results on Elec dataset

requires a little more time than those of SEA and AWE. The training time results of ACE are much higher than those of the other algorithms. This is because the drift detector of ACE executes drift detecting mechanism when each instance comes, which requires large amount of training time.

D. Testing Time

Table IV lists the average testing time results in 10^{-5} seconds for all algorithms. From Table IV we can see that MAE obtained the least testing time results since some component BPNNs did not participate predictions. In our experiments, about 15 to 20 component BPNNs were selected for most of predictions in MAE algorithm.

TABLE IV. AVERAGE TESTING TIME IN 10^{-5} SECONDS

Dataset	SEA	AWE	ACE	MAE
Hyp_S	1.50	2.00	3.35	**0.90**
Hyp_F	1.95	1.95	4.15	**1.55**
RBF_{GR}	**2.30**	2.35	7.05	**2.30**
RBF_{ND}	1.60	1.55	2.35	**0.75**
SEA_S	0.80	0.75	0.80	**0.05**
SEA_F	**0.38**	0.40	0.78	0.40
$Tree_S$	3.90	4.65	4.70	**3.50**
$Tree_F$	4.65	3.90	6.25	**3.55**
LED_M	5.60	5.45	5.35	**4.40**
LED_{ND}	7.15	6.45	6.90	**6.20**
Elec	**34.07**	**34.07**	35.16	**34.07**
Cover	1.38	1.38	1.38	**0.05**
Airlines	502.32	499.54	499.54	**459.68**
Average	43.66	43.42	44.44	**39.80**

MAE algorithm got the results of "0.05" on SEA_S and Cover datasets, which are much less than the other results. The reason is that, for these two datasets, MAE only recalled very small number of component classifiers when each data chunk came, which makes its testing process very fast.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a memorizing based adaptive ensemble algorithm, MAE, for data stream mining. The main contributions of MAE algorithm include: (1) it introduces human recalling and forgetting mechanisms into data stream mining; (2) it adopts ensemble pruning technique as the recalling mechanism; (3) it uses Ebbinghaus forgetting curve to emulate the forgetting mechanism. The proposed MAE algorithm was compared with other state-of-art algorithms, including SEA, AWE and ACE. The experimental results have shown that, with moderate chunk training time, MAE outperforms other algorithms in chunk predicting accuracy and testing time under different concept drifting scenarios. The experiments proved that MAE is a good and stable algorithm for data stream mining.

There are a lot of things waiting us to do in the future. Firstly, we need more experiments to optimize the values of the parameters in MAE algorithm, such as memory capacity, forgetting factor and the selection of ensemble pruning methods. Secondly, we plan to implement more algorithms, such as AUE [8] and drift detecting algorithms, in our system and do a more sufficient performance comparison.

ACKNOWLEDGEMENTS

This work is supported by the National Natural Science Foundation of China under Grants No. 61272141, 60905032 and 61120106005.

REFERENCES

[1] J. Gama, "Knowledge Discovery from Data Streams", 1st ed. London, U.K.: Chapman & Hall, 2010.

[2] A. Bifet, G. Holmes, B. Pfahringer, P. Kranen, H. Kremer, T. Jansen and T. Seidl, "MOA: Massive Online Analysis, a Framework for Stream Classification and Clustering," *Journal of Machine Learning Research - Proceedings Track* , 11: 44-50, 2010.

[3] W. N. Street and Y. Kim, "A streaming ensemble algorithm (SEA) for large-scale classification," in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2001, pp. 377–382.

[4] H. Wang, W. Fan, P. S. Yu, and J. Han, "Mining concept-drifting data streams using ensemble classifiers," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2003, pp. 226–235.

[5] K. Nishida, K. Yamauchi, and T. Omori, "ACE: Adaptive classifiers-ensemble system for concept-drifting environments," in *Proc. 6th Int. Workshop Multiple Classifier Syst.*, 2005, pp. 176–185.

[6] R. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *IEEE Trans. Neural Netw.*, 22:10, pp. 1517–1531, 2011.

[7] D. Brzezinski and J. Stefanowski, "Accuracy updated ensemble for data streams with concept drift," in *Proc. 6th HAIS Int. Conf. Hybrid Artif. Intell. Syst.*, II, 2011, pp. 155–163.

[8] D. Brzezinski and J. Stefanowski, "Reacting to Different Types of Concept Drift: The Accuracy Updated Ensemble Algorithm," *IEEE Transactions on Neural Networks and Learning System*, 24:7, 2013

[9] A. Lazarevic,, Z. Obradovic, "The effective pruning of neural network classifiers," In *IEEE/INNS International Conference on Neural Networks*, IJCNN2001, pp. 796–801

[10] G. Martinez-Munoz, A. Suarez, Pruning in ordered bagging ensembles. In: 23rd International Conference in Machine Learning (ICML-2006), 2006, pp. 609-616.

[11] I. Partalas, G. Tsoumakas and I. Vlahavas, "Pruning an ensemble of classifiers via reinforcement learning", Neurocomputing 72(7-9), 2009, pp. 1900-1909.

[12] Q. Zhao, Y. Jiang, M. Xu. A Fast Ensemble Pruning Algorithm based on Pattern Mining Process. Data Mining and Knowledge Discovery (ECML/PKDD2009). 19:227-292, 2009

[13] http://www.phase-6.com/en/vocabulary-tool/scientific-background /scientific-background.html

[14] H. Ebbinghaus. "Memory: A Contribution to Experimental Psychology," (Translated by Henry A. Ruger & Clara E. Bussenius). http://nwkpsych.rutgers.edu/~jose/courses/578_mem_learn/2012/readings/Ebbinghaus_1885.pdf

[15] http://moa.cms.waikato.ac.nz/

[16] Riedmiller M., Heinrich B. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In *Proceedings of the IEEE International Conference on Neural Networks* (ICNN). San Francisco, 1993: 586-591

[17] M. Harries, "SPLICE-2 comparative evaluation: Electricity pricing," School of Computer Science and Engineering, Univ. New South Wales, New South Wales, Australia, Tech. Rep. 9905, 1999.