2009 International Symposium on Computing, Communication, and Control (ISCCC 2009)
Proc .of CSIT vol.1 (2011) © (2011) IACSIT Press, Singapore

An Agent Based Method to Predict the Subsequent Concept in Concept Shifting Data Streams

Hossein Morshedlou [1] and Ahamad Abdollahzadeh Barforoush[1+]

[1] Computer Engineering and Information Technology Department
AmirKabir University of Technology, Tehran, IRAN

Abstract. The advent of new application areas leads to intensive research on data streams. Due to concept shifts in the underlying data, maintaining a timely updated model has become one of the most challenging tasks. The data streams originate from the events of the real world so we think the associations among these events should exist among the concepts of the originated data stream too. Extraction of hidden associations among the concepts can be useful for prediction of subsequent concept in data stream. In this paper we present an agent-based method for classification of data streams with concept shifts. The agent creates a history from the concept changes and uses this history to have an intelligent behavior. The results of conducted experiments showed that the agent is proper for classification of concept shifting data streams.

Keywords: Agent, Classification, Data Stream, Concept Shift.

1. Introduction

The advent of new application areas leads to intensive research on data streams. Due to concept shifts in the underlying data, how to maintain a timely updated model has become one of the most challenging tasks in mining data streams. Many approaches, including both the incrementally updated classifiers [1, 2, 5] and the ensemble classifiers [3, 6] have proved that model update is a very costly process and not depends on rate of disturbance in the data. Data streams originate from the events of the real world and there are associations among the occurrence of these events so we think these associations there are among the concepts in originated data streams too. Recurring events in the real world is another important issue which causes recurring concepts in data streams. Extraction of hidden associations among the recurring concepts can be useful for prediction of following concepts in data stream. If a method can properly behave when a concept shift occurs, it can adapt itself with data stream changes and will be succeed. Such a method should be capable of learning associations, reasoning and having a proper behavior when concept shift occurs.

Considering the capabilities of agents (e.g. intelligence, learning, reasoning,) it seems that using agent for data stream classification task is an interesting approach. In this paper we present a new agent-based method for classification of data streams with concept shift. Section 2 introduces background knowledge. It contains basic term definitions. Section 3 explains the concept shift detection method and in section 4 we propose a mechanism to build a history along the journey of concept changes. Section 5 defines intelligent behavior. Section 6 shows the experiments and the results of the conducted experiments. In section 7 we give concluding remarks.

2. Basic Definitions

[+] Corresponding author. Tel.: +989155044052
E-mail address: Hossein_morshedlou@yahoo.com、

- Data Stream: We consider data stream as a sequence of instances where each instance is a vector of attribute values. Each instance has a class label. If class label of an instance is known, it is a training instance. Otherwise it is an unlabeled instance. Set of training instances form our training data.
- Concept: The term concept is more subjective than objective. Particularly in this paper, a concept is represented by a data model which learned by a classification algorithm.
- Concept Shift: Change of the underlying concept in data stream is called concept shift. After concept shift, the previously learned data model (we call it current data model) is not accurate any more and should be updated or replaced by a new data model.
- Proactive Behavior: Given the current concept, Proactive behavior foresees what the forthcoming concept is. The predicted concept will take effect once a concept shift is detected.
- Reactive Behavior: Reactive behavior does not predict what concept is coming. A new data model is learned upon a concept shift is detected. How to understand the recently learned data model is more similar to which one of the previously learned data models, is described in section 5.
- Monitoring Window (MW): Monitoring Window is used for buffering instances. When a concept shift is detected, agent should wait to receive enough instances for learning a stable data model. When Monitoring Window is full, agent has sufficient instances to induce a stable data model.
- Detection Window (DW): Detection Window is inside the Monitoring Window and used for concept shift detection. MW and DW are the agent's sensors.
- Training Data Feature (TDF): Consider sequence of instances as training data when each instance is a vector of attribute values like (a_1, …, a_m, C). Here C is class label. Assume we have *k* distinct classes. TDF of this training data is defined as (1).

$$((CF_1(1,1), CF_2(1,1), n_1, C_1), \ldots, (CF_1(k,1), CF_2(k,1), n_k, C_k))$$
$$\ldots \quad (1)$$
$$((CF_1(1,m), CF_2(1,m), n_1, C_1), \ldots, (CF_1(k,m), CF_2(k,m), n_k, C_k))$$

In (1), $CF_1(i,j)$ is the sum of the squares of a_j attribute values in instances with C_i class label. $CF_2(i,j)$ is the sum of a_j attribute values in instances with C_i class label. n_i shows the number of instances with C_i class label.

3. Concept shift detection

The Sliding-window methodology is used here to detect the underlying concept shift in data stream. Two important parameters are DW size and error threshold. The agent uses the current data model to predict the class of coming instances one by one. The beginning of the DW is always a misclassified instance. Whenever the DW is full and its error rate exceeds the error threshold, the beginning instance is taken as a trigger; otherwise, the beginning of the DW is slid to the next misclassified instance (if there is any) and the previous instances are dropped. When a concept shift occurs, the current data model is no longer appropriate for classification task and must be replaced.

4. Building Histories

The mechanism which the agent uses to build concept histories along the journey of concept change is described here. For each concept and concept shift, the agent maintains a history as shown in table 1. Consider this history contains information about concept A. This history contains information about when the concept A occurred for the first and last times, How many times concept A has happened since the first time and when is the previous n^{th} time that concept A happened for n = 5, 10, …, 1280. A new occurrence of concept A changes this information as below:

- Last Time: will be updated to time of the new occurrence
- Counter: will be increased by one.
- Previous 5^{th} time: 512000+ ((621000-512000)/5) // Assume 621000 is time of the new occurrence
- Previous 10^{th} time: 445000+ ((621000-445000)/10)

[1] For example consider A and B are concepts. The agent maintains two histories for A and B and two histories for AB and BA (if there are any concept shift from A to B or from B to A). These histories hold the information of the past concepts and concept shifts.

Because of the prohibitive volume, it is very expensive to keep all the streaming data. In comparison, a data model is compact such as a bunch of abstract rules. Keeping data models of concepts is much more tractable, so the agent stores the data models. They can be reused in the future instead of learning from scratch. Having the concept shifts, this helps to reduce the prediction time. Furthermore the associations among different concepts can be extracted using the histories. In addition of a data model, agent stores a TDF value too. The agent calculates the TDF value using the training data used to learn the data model. In section 5 we will explain the usage of TDF.

TABLE I HISTORY STRUCTURE FOR CONCEPTS AND CONCEPT SHIFTS

Counter	1	5	10	20	40	80	160	320	640	1280	First
147	620000	512000	445000	300000	200000	98000	-1	-1	-1	-1	800

TABLE II HISTORY STRUCTURE FOR CONCEPT A

Counter	1	5	10	20	40	80	160	320	640	1280	First
89	320000	290000	260000	150000	52000	8000	-1	-1	-1	-1	800

4.1. Analysis of Histories

Consider the history of table 2. We want to estimate how many times A occurred in [230000, 330000]. Table 2 shows that the occurrence time of concept A in 10^{th} and 20^{th} previous times had been in 260000 and 150000 respectively. 10 times occurrence of A in [150000, 260000] means that this concept approximately occurred three times in [230000, 260000]. So A occurred near 13 times in [230000, 330000].

5. Intelligent Behavior

Having a concept shift, the correct decision making based on knowledge (here historical information) to behave proactively or reactively is an intelligent behavior. Agent architecture has good potentials for implementing intelligent behaviours. Discussion about the architecture of agents is not in the scope of this paper. In following sections first we will describe proactive and reactive behaviors in more detail and then we will explain how agent decides to use either proactive or reactive behavior to have an intelligent behavior.

5.1. Proactive Behavior

Consider the current concept in data stream is A. The next concept may be one of B, C or … concepts. These are beliefs of agent in its knowledgebase and agent assigns a probability to each belief. The belief with the higher probability shows the concept which is more probable as the next concept. So agent arranges beliefs based on their probability in descending order. Without losing the order of beliefs, the agent inserts beliefs which their probability is more than a certain threshold to a buffer. When concept changes, if agent decides to behave proactively, this buffer will be used to predict the next concept. First belief in the buffer shows the most probable concept as the next one. If the prediction is correct, the agent knows which previously learned data model is appropriate and should record this shift in the related histories. Being a wrong prediction, the agent should decide to continue proactive behavior or not. If the agent wants to continue proactive behavior, the next belief in the buffer will be checked as the next one like before. When resuming the proactive behavior is not rational or buffer is empty, the agent will behave reactively. Here one question remains which should be answered.

How much is the probability of occurrence of a certain concept as a next concept? Consider the agent want to calculate the probability of B as a next concept (the current concept is A). As mentioned in section 4, the agent maintains the concepts and concept shifts histories. Using the histories, agent determines how many times concept A and shift from A to B are occurred in (t4,t3), (t3,t2) and (t2,t1) when $t4<t3<t2<t1$ and t1 is the current time (In section 4.1 we described how the agent determines how many times a concept or concept shift is occurred in a specific interval). The agent uses (2) to approximate the occurrence probability of B after A. $n(A \cap B)$ and $n(A)$ in (2) mean how many times AB and A are occurred in that interval

respectively. After estimation of the probabilities, the agent forms (t4, p4), (t3, p3), (t2, p2) pairs while p4, p3 and p2 are the probabilities in (t4, t3), (t3, t2) and (t2, t1) respectively. Now the agent uses an extrapolation method to estimate the value of p1 in (t1, p1). p1 shows that how much is probability of being the next concept for B.

$$P(B|A) = P(A \cap B)/P(A) = n(A \cap B)/n(A) \quad (2)$$

5.2. Reactive Behavior

When a concept shift is detected, agent tries to replace the current data model with a proper one. If agent can to predict the next concept, it replaces the current data model quickly; otherwise agent should wait to receive enough instances for training a stable data model. Please note that the DW size numbers of instances are sufficient to indicate that the current data model is not proper any more, but insufficient to induce what the proper model should be. After accumulation of enough instances, the agent learns a new data model from scratch. If there is not a data model similar to the new learned one in memory, the agent should create two new history (one to hold the information of new concept and the other for concept shift from previous concept to the new one) and store them with the new data model in knowledgebase unless storing the information of shift in the related histories is sufficient. To check the existence of a data model similar to the new learned one, the agent calculates the TDF feature of accumulated data. Having TDF we can calculate CF_3 and CF_4 using (3) and (4).

$$CF_3(i,j) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 = \frac{1}{n}[\sum x_i^2 - \frac{1}{n}(\sum x_i)^2] = \frac{1}{n}[CF_1(i,j) - \frac{1}{n}(CF_2(i,j))^2] \quad (3)$$

$$CF_4(i,j) = \frac{1}{n}(\sum x_i) = \frac{1}{n}(CF_2(i,j)) \quad (4)$$

So we define $CTDF_{current}$ as (5).

$$\begin{matrix} ((CF_3(1,1),\ CF_4(1,1),C_1),\ldots,\ (CF_3(k,1),\ CF_4(k,1),C_k)) \\ \ldots \\ ((CF_3(1,m),\ CF_4(1,m),C_1),\ldots,\ (CF_3(k,m),\ CF_4(k,m),C_k)) \end{matrix} \quad (5)$$

The agent calculates the CTDF of other data models in memory using their TDFes and then chooses the *alpha* nearest ones to $CTDF_{current}$ based on Euclidean distance. Starting from the data model of nearest one, data models are checked by classification of accumulated data instances and comparing the result with true class labels. If number of misclassified instances is less than a certain threshold, the data model is similar to new learned one else the next data model will be checked.

5.3. Proactive vs. Reactive

In this section we want to answer to this question that when the agent should behave proactively and when reactively? We propose that the agent should behave proactively until it is rational. When resuming the proactive behavior is not rational, the agent should switch to reactive behavior. Now how can agent distinguish the proactive behavior is not rational? As we said in section 5.1, the agent inserts beliefs with the probability more than threshold to a buffer in descending order. When a concept shift is detected, at the most, the agent checks no more than the first *K* beliefs in the buffer. *K* is determined by pseudo code of (6).

$\forall\ A_i \in \{A_1,\ldots,A_m\} \rightarrow P(A_i \mid A_0) > Treshold$,

T_{train} = The required time to learn a data model from scratch, T_{test} = The required time to test a data model on current data

D= T_{train} ;

For (r=0; r < m ; r++){

$$D_{temp} = \sum_{i=1}^{r}(P(A_i \mid A_0)\times i \times T_{test}) + (1 - \sum_{i=1}^{r} P(A_i \mid A_0))\times(T_{train} + r \times T_{test}) \quad (6)$$

if ($D > D_{temp}$) { $D = D_{temp}$; K=i; }

}

In the above pseudo code, A_0 is the current concept and A_1 to A_m are concepts that their probability are more than threshold and $P(A_1) > P(A_2) > \ldots > P(A_m)$. T_{train} and T_{test} show the required time to learn a data model from scratch and to test a previously learned data model on current data respectively. Consider e.g. the current concept in data stream is D and probabilities of being the next concept for C, F, B and A are 0.35, 0.28, 0.17 and 0.03. We show D, C, F, B and A with A0, A1, A2, A3 and A4 respectively. Also T_{train} and T_{test} are 30 and 5. So we have:

- *r=0: required-time = 30*
- *r=1: required-time = 0.35 × 1 × 5 + (1 - 0.35) × (30 + 1× 5) = 24.5*
- *r=2: required-time = 0.35× 1× 5 + 0.28 × 2× 5 + (1- 0.35 - 0.28) × (30 + 2× 5) = 19.35*
- *r=3:required-time = 0.35× 1× 5+0.28× 2× 5+0.17× 3 × 5+(1-0.35-0.28-0.17)× (30 + 3× 5) = 16.1*
- *r=4: required-time = 0.35× 1× 5 +... + (1 - 0.35 - 0.28 - 0.17 - 0.03) × (30 + 4× 5) = 16.2*

According to above expressions, the required time has minimum value when r is 3, so when a concept shift is detected, at the most, the agent will check no more than first three beliefs (C, F and B) and switch to reactive behaviour when the data models of these three beliefs, all are inappropriate for new concept.

6. Experiments

6.1. Data

This paper employs benchmark data sets. The experiments involve both artificial data and real world data. By using artificial data, one can access information such as what type of concept change is taking place and when. Our artificial data sets are Stagger and HyperPlane. You can see the details of these datasets and how to generate a data stream with concept shifts using these datasets in [4]. We also used real life data set "nursery" in the University of California, Irvine (UCI) Machine Learning Repository. We randomly sample instances to generate a stream. To simulate concept shift we randomly select some attributes of the data set and change their values in a consistent way. One method is to shuffle the values; for example we can change (a0, a1, …, an) values as (a1, a2, …, an, a0) for all instances and keep the class label intact.

6.2. Results

We selected WCE [3] and RePro [4] as rival methods and C4.5 decision tree algorithm as the learning method for all the three methods. Our evaluation measures are accuracy and the needed time to classify one million instances from data stream. We assume in starting point (time = 0) the agent and RePro had been learned appropriate data models for commonly recurring concepts in data stream and afterward they will learn new data models for the concepts which never had been seen before. The monitoring window and detection window sizes are 500 and 200 respectively and the error threshold of detection window is 25%. After every 2500 instances, we change the concept of data to simulate concept shift.

Figure 1 shows the performance of RePro and Agent on one million data from HyperPlane in comparison with optimal method. In optimal mode we know what the next concept will be. As shown in this figure, the agent performs closer to the optimal mode than RePro. Classification time of WCE is very longer than classification time of both agent and RePro, therefore it is not included here. Figure 2 and 3 show the results on Stagger and Nursery data streams. Table 3 shows the accuracy of the methods on data streams generated from HyperPlane, Stagger and Nursery data. WCE learns data models frequently, so it suit with gradual concept drifts and drop its accuracy when we have concept shift in data streams. In stopping mode when a concept shift is detected, agent pauses the classification task and fits its data model to the new concept in data stream, but in non-stopping mode, agent continue the classification of data and fits its data model to new concept in parallel with classification task.

7. Conclusion

Both foreseeing into the future and retrospection into the history have been proven important in many aspects of life. This paper incorporates those useful practices in mining data streams by using an agent with proactive-reactive behaviour and has proposed a mechanism to organize into a concept history. As a result, the problem of the intractable amount of streaming data is solved since data models are much more compact than raw data while still retain the essential information. Besides, associations among concept shifts can be

extracted from the histories, which help to foresee the future. Using the histories and incorporating reactive and proactive behaviour, the agent will be capable to achieve both effective and efficient predictions in various scenarios of concept change, including concept shift.

8. Acknowledgements

We would like to thank ITRC for their guidance and support in this research.

Figure 1: classification time on HyperPlane

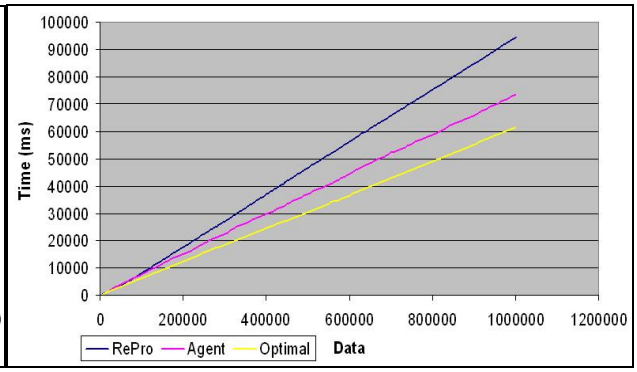

Figure 2: classification time on Stagger

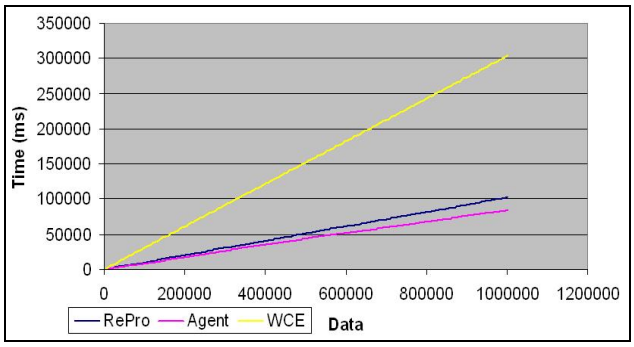

Figure 3: classification time on Nursery

Dataset	Mode	Agent	RePro	W CE
HyperPlan	Non-Stopping	80.6	70.9	76.3
	Stopping	92.7	91.6	
Stagger	Non-Stopping	97.2	90.2	82.1
	Stopping	98.6	98.8	
Nursery	Non-Stopping	83.4	75.8	73.9
	Stopping	93.9	92.7	

TABLE III CLASSIFICATION ACCURACY

9. References

[1] P. Wang, H. Wang, X. Wu, W. Wang, and B. Shi, "A Low-Granularity Classifier for Data Streams with Concept Drifts and Biased Class Distribution", IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 9, Sept., 2007, pp. 1202-1213.

[2] P. Domingos, and G. Hulten, "Mining high-speed data streams", ACM Press, Boston, MA, 2000, pp. 71–80.

[3] H. Wang, W. Fan, P.S. Yu, and J. Han, "Mining concept-drifting data streams using ensemble classifiers". In Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2003, pp. 226–235

[4] Y. Yang, X. Wu, and X. Zhu, "Mining in Anticipation for Concept Change: Proactive-Reactive Prediction in Data Streams", Journal of Data Mining and Knowledge Discovery, Springer, ISSN: 1384-5810, Volume 13, Number 3, November, 2006, pp. 261-289.

[5] G. Hulten, L. Spencer, and P. Domingos. "Mining time changing data streams". *In SIGKDD*, ACM Press, CA, 2001, pp. 97–106.

[6] W.N. Street and Y.S. Kim. "A streaming ensemble algorithm (SEA) for large-scale classification". In *SIGKDD*, 2001.