Modeling Concept Drift from The Perspective of Classifiers

Bai Su, Yi-Dong Shen
Institute of Software, Chinese Academy of Sciences
Beijing, China
subai@ios.ac.cn, ydshen@ios.ac.cn

Wei Xu
NEC Laboratories America, Inc.
Cupertino, CA 95014, USA
xw@sv.nec-labs.com

***Abstract*—The problem of concept drift is of increasing importance to data mining as more and more data is organized in the form of data streams rather than static database, and it is rather unusual that concepts and data distribution stay stable over time. In this paper, we model the concept drift as the changes of the optimal parameters of the discriminative model. By employing the extended Kalman filter to track the optimal parameters in this model, we get a dynamical classifier with adaptability to the dynamics of concept drift. The empirical results in both synthetic and real data sets indicate that the proposed algorithm is an effective and efficient solution to classification for evolving data streams.**

***Index Terms*—Data Streams, Classification, Concept Drift.**

I. Introduction

As there is a growing number of emerging applications of data streams, such as sensor data, network data, and web click streams, mining data streams is becoming increasingly important. As a major mining task, classification on data stream has been extensively studied in recent years.

The classification tasks in data streams are different from traditional ones at least in two aspects. First, data streams often grow without limit. Since it is impossible to cache all historical data, classifiers running in an online and one-pass style are often preferred. Second, in a system where time is a variant, the underlying mechanism that generates the data may change over time, which is referred to as concept drift. If we only build a static model in the presence of concept drift, it would cause a high error rate in classification.

So far, most of efforts [6], [7], [8], [9] were put on devising a suitable strategy or heuristic method to explore the capabilities of existing static classifiers for mining concept-drifting data streams. However very few efforts [5] concern a dynamical classifier suitable for the concept-drifting scenario and still few [7] concern modeling concept drift. In this paper, by modeling concept drift explicitly, we devise a classifier which can dynamically adjust its parameters to adapt to concept-drifting data stream.

Generally, we can identify three distinct approaches to the classification problem. The simplest involves constructing a discriminant function that directly assign each input x to a specific class. Most existing data streaming mining works belong to this category. In some situations, however, it is more desirable to model the conditional probability distribution $p(C_k|x)$ and then subsequently to use this distribution to make optimal classifications. There are two different approaches to determining the conditional probabilities $p(C_k|x)$. One is to model the class-conditional densities given by $p(x|C_k)$, together with the prior probabilities $p(C_k)$ from the classes and then compute the required posterior probabilities using Bayes' theorem. Alternatively, we can model them directly for example by representing them as parametric models and then optimizing the parameters using a training set. Approaches that model the posterior probabilities directly are called discriminative models. In this paper, we will focus on the third approach, i.e., the discriminative model.

In order to minimize the influence of concept drift on the performance of classifier, first we need to investigate the reasons for concept drift from the perspective of data distribution. In [4], all four possible cases of concept drift have been summarized based on changes in probability distribution. However, since a discriminative model only concerns the conditional probability, we argue that from the perspective of discriminative models, we only need to consider one specific case of concept drift, i.e., the changes in the posterior distribution $p(C_k|x)$.

The rest of the paper is organized as follows. We briefly describe the problem of streaming data classification in the next section. Related work is given in Section III. We introduce an dynamical probabilistic model for concept drift in the section IV. Section V discusses how to estimate some parameters in this model. We give an algorithm for training the dynamical classifier in Section VI. Experimental results are given in Section VII, followed by conclusion in Section VIII.

II. Problem Definition and Analysis

The streaming data classification problem is generally defined as follows. A set of N training example of the form (x_t, y_t) is given with a unique time stamp at each example, where N is an arbitrarily large integer, y_t is a discrete target variable which represents class label and x_t is a vector of D attributes. Our goal is to produce from these examples a discriminative model $P(y = c_i|x) = f(x, w_t)$ that will predict the probability that an arbitrary x belongs to c_i at time t.

In this paper, we focus on the problem that the underlying process behind the streaming data may not be static, i.e., it may change over time. We call such changes concept drifts.

One popular option to meet both the online processing

978–1–4244–1674–5/08/$25.00 © 2008 IEEE

requirement and the requirement of keeping the model up-to-date is the sliding window scheme. However, it suffers from two major drawbacks. First, the optimal size of sliding windows is hard to determined. One fixed window size L will not be appropriate for every type of concept drift. It may be beneficial to dynamically change L during a run. For example, it may make sense to shrink L in response to a rapid change of concept to exclude the outdated data. Similarly, some applications may benefit from an increase in L when the concept is stable - a good time to learn a more detailed model. Second, a model learned from a sliding window is vulnerable to over-fitting due to the limited data in a sliding window.

Another approach to dealing with the problem of concept drift is based on detecting methods. In this approach, we only need to change the model when the detecting system indicates the occurrence of concept drift. An obvious idea of detecting concept drift is to monitor the error rate in classification. If it exceed a user-defined threshold, we are convinced that a concept drift is really happening. But this method requires that a part of training examples are allocated to determine the error rate, which could aggravate the over-fitting problem. In addition, this approach is questionable when facing smooth changes of concept which generally tend to incurs low error rate. The third flaw naturally related to this approach is that when detecting the occurrence of concept drift, a new model has to be learned from scratch, which always incurs high error rate at the beginning of the learning phase.

To avoid these difficulties, we propose a different approach which dynamically adjusts model parameters based on a feed-back mechanism. In this approach, the model is updated every time a new labeled example arrives. In essence, we adjust parameters of the discriminative model according to the label of each incoming example. If the label of incoming example at time t agrees with the prediction that the model made, we tend to believe that the model is right; if it is against the prediction, we get the opposite information. In both cases, the parameters of the model can be adjusted by incorporating new information. One may doubt that there is no "free lunch" and this model could suffer from the outdated history data because it does not explicitly divide the outdated data from the new data as the sliding window scheme does. But since the adjustments of the model only involve parameters and the model does not include complex structures like the nodes in decision trees, the model could easily be adapted to the new concept as shown by the experiment results. Furthermore, the learning rate of the model could also be adjusted according to the data distribution.

III. Related Work

There is much work in the area of data stream classification. The CVFDT [5] tries to eliminate the influence of concept drift based on a sliding window scheme and adaptively change the decision tree by growing alternative subtrees in questionable portions of the old tree. When the accuracy of the alternative subtree outperforms the old subtree, the old one is replaced. The On-Demand-Stream classifier [8] focuses on the issue of how to choose the best window size by storing a series of micro-clusters in a geometric time frame and then choosing the best time window according to the best performance. The weighted classifier ensemble [6] represents the effort of boosting the classifiers based on their accuracy.

The following work is closely related to ours. The K-ADWIN [11] algorithm is also a sliding window scheme in which a Kalman filter is employed to estimate the average of data. When two sub-windows exhibit distinct averages, one can deduce the existence of concept drift and the older sub-window is dropped. The RePro algorithm [9] models concept drift as a first-order Markov chain and maintains a concept history to predict the next concept. Wang, et. al. [7] proposed to model concept drift as a hidden Markov model and assume the dynamics of concept drift is a Poisson process. We model concept drift as changes of the optimal parameters in a discriminative model and get different results.

IV. Our Approach

To investigate the nature of the problem, first we build a dynamical probabilistic model as follows

$$w_t = g(w_{t-1}) + s \tag{1}$$

$$p(C_k|x_t) = f(w_t) + v \tag{2}$$

where $f(\cdot)$ is the optimal discriminative model and by optimal we mean it has a optimal parameter vector w; v is a random variable which represents the uncertainty in the posterior distribution $p(C_k|x_t)$; $g(\cdot)$ denotes the relationship between the value of parameter vector w at time $t-1$ and the one at time t; s is a random variable which represents the uncertainty in this relationship. Since the evidence of concept drift exists only in the changes of its parameter vector from the perspective of the optimal discriminative model, $g(\cdot)$ and s together define the dynamics of concept drift. The distributions of s and v can be varied over time, but for simplicity we just assume they are stable. Note that the parameter vector w_t is a hidden variable of the model and $p(C_k|x_t)$ can be viewed as an indirect measurement of w_t.

In classification tasks, our goal is to infer the value of the optimal parameter w_t. To achieve this goal in an tractable way, we continue to make some assumptions about this model. With no prior knowledge of concept drift, which usually happens in practice, we can safely assume that the expectation of parameter w at time $t-1$ is just the same as the one at time t. For simplicity we can assume that s and v follow zero mean normal distributions with isotropic covariance. Then we have

$$w_t = w_{t-1} + s \tag{3}$$

$$s \sim N(0, aI) \tag{4}$$

$$v \sim N(0, r) \tag{5}$$

where I is the identity matrix and a is a single value which controls the variance of the parameter vector w. Note that all the information we have in this data stream classification task is the previous measurements and the current measurement. So

the problem of inferring the value of w is also called filtering problem in machine learning literatures [2][10].

Having been widely used in the area of control and navigation, the Kalman filter [1] is an optimal recursive data processing algorithm in that it minimize the posterior estimate error covariance. One of its assumption is linear model. But the optimal discriminative model for classification is usually nonlinear in practice. So, in order to apply the kalman filter scheme, in this paper, we use one of its nonlinear forms: the extended Kalman filter [1] to infer the value of the parameter vector w_t. The main idea of the extended Kalman filter is to approximate the nonlinear model using its Taylor expansion. As such, we expand $f(w_t)$ by Taylor series around w_{t-1} and truncate it and use terms up to the first order, then (2) changes to

$$p(C_k|x_t) = f(w_{t-1}) + H_t(w_t - w_{t-1}) + v \tag{6}$$

$$H_t = \left.\frac{\partial f}{\partial w_t}\right|_{w_{t-1}} \tag{7}$$

The extended Kalman filter addresses the problem of estimating the hidden variable w_t by using a form of feedback control: the filter first estimates its value at some time and then obtains feedback in the form of measurement $p(C_k|x_t)$ with certain measurement noise v. As such, it is common to split the computation steps into two groups: "predict" equations and "correct" equations. The predict equations are responsible for projecting forward the current value of the hidden variable and error covariance of estimates to obtain a priori estimates for the next time step. The correction equations are responsible for incorporating a new measurement into a priori estimate to obtain an improved posterior estimate. By applying the extended Kalman filter to solving the filter problem in our model, we can have the following equations

$$w_t^- = w_{t-1}^+ \tag{8}$$

$$P_t^- = P_{t-1}^+ \tag{9}$$

$$K_t = P_t^- H_t (H_t^T P_t^- H_t + r)^{-1} \tag{10}$$

$$w_t^+ = w_t^- + K_t \left(p(C_k|x_t) - f(w_t^-)\right) \tag{11}$$

$$P_t^+ = (I - K_t H_t^T) P_t^- \tag{12}$$

where w_t^- is the priori estimate for w_t; P_t^- is the priori estimate for the covariance of w_t; w_t^+ is the posterior estimate for w_t; P_t^+ is the posterior estimate for the covariance of w_t; K_t is called the Kalman gain matrix. (8) and (9) are the "predict" equations while (10), (11) and (12) are the "correct" equations.

Theoretically, any type of discriminative model with continuous parameters is the candidate of $f(\cdot)$. In this paper, we choose the well-known logistic regression as the specific model, although the same idea applies to other discriminative model as well. In the case of two-class classification, the logistic regression model is given by

$$f(w) = \sigma(w^T x) \tag{13}$$

where $\sigma(\cdot)$ is the logistic sigmoid function[2]. In the case of multiclass classification, it takes the form

$$f_k(w) = \frac{exp(w_k^T x)}{\sum_k exp(w_k^T x)} \tag{14}$$

where k is the index of class.

For the simplicity in illustration, we consider two-class classification problem. Note that $H_t = \sigma(w_{t-1})(1 - \sigma(w_{t-1}))x_t$. Thus the "correct" equations are updated to

$$K_t = \frac{P_t^- \sigma(w_{t-1}) \left(1 - \sigma(w_{t-1})\right) x_t}{\left(\sigma(w_{t-1}) \left(1 - \sigma(w_{t-1})\right)\right)^2 x_t^T P_t^- x_t + r} \tag{15}$$

$$w_t^+ = w_t^- + K_t \left(p(C_k|x_t) - \sigma(w_t^-)\right) \tag{16}$$

$$P_t^+ = (I - \sigma(w_{t-1})(1 - \sigma(w_{t-1}))K_t x_t^T) P_t^-. \tag{17}$$

In the case of two-class problems, we use binary representation in which there is a single target variable $y \in \{0, 1\}$ such that $y = 1$ represents class C_1 and $y = 0$ represents class C_2. We can interpret the value of y as the probability that the class is C_1, with the values of probability taking only the extreme values of 0 and 1. As a result, we have $p(C_k|x_t) = y_t$ in (16). For the problem of multiclass, the similar scheme can be adopted.

V. Estimate Filter Parameters

In the actual implementation of our model, the measurement noise variance r and aI which is the initial value of P_t^- need to be estimated from data. To obtain an on-line algorithm, we take an approximate approach. We treat y_t as a binary sample drawn from a Bernoulli distribution with occurrence probability $\sigma(w_t^T x)$, then we have $r \simeq \sigma(1 - \sigma)$.

The random variable s in our model expresses the uncertainty of the relationship, which is caused by concept drift, between the optimal parameter vectors at two neighborhood time points, whose covariance aI indicates the degree of concept drift. One way to think about it is that when P_t^- approaches zero, which indicates that the concept is stable, K weights the residual less heavily in (16). Therefore, by estimating the value of a we can adjust the learning rate of the model. Let's consider the following equation

$$p(y_t = 1) = \int p(y_t = 1|w_t) p(w_t)\ dw_t \tag{18}$$

where $p(y_t = 1|w_t) = \sigma(w_t^T x_t)$ and $p(w_t) = N(w_t, aI)$. (18) represents the convolution of a Gaussian with a logistic sigmoid function. According to [2], we have

$$p(y_t = 1) \simeq \sigma\left(\kappa\left(a x_t^T x_t\right) w_t^T x_t\right) \equiv h(a) \tag{19}$$

$$\kappa(x_t) = (1 + \pi x_t/8)^{-1/2} \tag{20}$$

Because $y_t \in \{0, 1\}$, we have

$$p(y_t) = h(a)^{y_t} \left(1 - h\left(a\right)\right)^{1 - y_t} \tag{21}$$

$$p(Y) = \prod_{t=1}^{N} p(y_t) \tag{22}$$

so $p(Y)$ is a function of a. By maximize the evidence $p(Y)$ we can find the desired a, which is an nonlinear optimization problem with the constraint condition $a > 0$. We propose to solve a using the method of preconditioned conjugate gradients [3]. Finally, we have

$$a = \underset{a}{argmax}\ ln\left(p(Y)\right) \tag{23}$$

VI. Algorithm

In this section, based on the proposed methods, we describe an algorithm for training the dynamical classifier.

At the initial stage, the algorithm uses the *iterative re-weighted least squares* (IRLS) method [2] to train a logistic regression classifier based on a off-line data set. Then we use the parameter vector of the initial classifier as the initial value of w_t. Based on the same data set, the initial value of covariance of the measurement noise and parameter vector can also be estimated. With these parameters, we can use the methods described in Section IV and V to adjust the parameter vector of the classifier online. Because the covariance of w_t which denotes the degree of concept drift could change over time, we estimate its value based on a length-fixed buffer when it is filled with new examples and then update the related equations periodically. Notice that although we use a time-related buffer here, the purpose is totally different from previous work which use the data in sliding window to train a new classifier. In addition, the performance of our classifier is quite stable when the size of the buffer is varied as shown in experiments.

Input: S: a dataset from the incoming stream
C: a off-line dataset for evaluating the initial value of parameters
K: a size-fixed buffer for estimating a
Output: w_t: a series of parameter vector of classifier for each time stamp

learn the initial parameter vector w_t from C using the IRLS method;
estimate the value of a from C using (23);
while *S not empty* **do**
  get an instance x_t from S;
  compute the prior estimate for w_t and P_t using (8).(9);
  compute the posterior estimate for w_t and P_t using (15), (16), (17);
  output the posterior estimate of w_t;
  if *the buffer K is full* **then**
    estimate the value of a from K using (23);
    $P_t^- = aI$;
    empty K;
end

Algorithm 1: Training the Dynamical Discriminative Model

TABLE I
Parameters for the Data Set.

Symbol	Meaning
f_c	Concept drift frequency (per records)
d	The intensity of concept drift
D	The number of dimension of data sets
p_{noise}	The level of noise

The algorithm for classification is straightforward, and it is omitted here. Basically, given a test case x_t, the logistic regression classifier $\sigma(w_t)$ is applied on it.

The main computational load of our algorithm involves (15), (16), and (17) whose complexity is $O(D)$ where D is the number of dimension. So the computational complexity of our algorithm is O(nD) which makes it suitable for fast and high-dimensional streaming data classification.

VII. Experiments

We conducted an empirical study to examine the performance of our proposed algorithm. The experiments were conducted on an Intel Pentium 4 PC (1.66GHz) with 1G main memory, running Windows XP Professional.

A. Data sets

There are two data sources in our experiments. The first is a synthetic data generator using a rotating hyperplane, which is a popular concept drifting data streams generation method having be used by [4], [5], [6], [7], [8], [9]. A D-dimensional hyperplane can be viewed as a set of points which satisfy

$$\sum_{i=1}^{D} w_i a_i = w_0 \tag{24}$$

where a_i is the coordinate of the ith dimension. We can treat the vector $< a_1, a_2, \ldots, a_D >$ as a data record, where a_i is the value of attribute A_i. The class label v of the record can be determined by the following rule: if $\sum_{i=1}^{D} w_i a_i > w_0$, it is assigned the positive label; otherwise, it is assigned the negative label. By randomly assigning the value of a_i in a record, an infinite number of data records can be generated in this way. One can regard w_i as the weight of A_i. The larger w_i is, the more dominant is the attribute A_i. Therefore, through rotating the hyperplane to some degree by changing the magnitude of w_i, the underlying data distribution changes, which is equal to concept drift. In our experiments, we set w_0 to 0 and restrict the value of w_i in $[-1, 1]$. We increase the value of w_i with $+0.1d$ or $-0.1d$ once every f_c records. After it reaches either -1 or $+1$, it then changes in the opposite direction. While generating the synthetic data, we also inject noise into the data. With the probability p_{noise}, the data is arbitrarily assigned to the class labels. In Table I, we collect the parameters used in the synthetic data sets and our experiments.

The second data source is the sensor data[12] collected from a drilling process in a petroleum well with 20000 records arranged according to the depth. Each example includes 18 attributes with a class label as either bit balling or strong

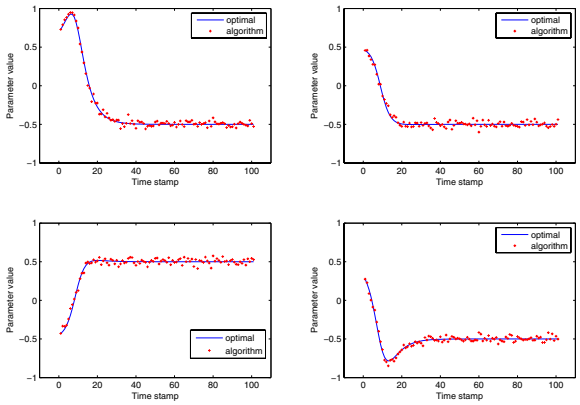

Fig. 1. The Changes of Parameters in the First Four Dimensions

rock, which indicates the current status of the bit body. As the drilling pipe goes deeper into the ground, the geological conditions around will change a lot, which causes the changes in data distribution of sensor data. We ran principal components analysis on that data set, saving only the top ten principal components as the attributes of the data record.

In all experiments, we allocated 50% data to be the test data.

B. Accuracy

The first experiment show how closely our algorithm can approximate to the optimal classifier by monitoring the deviation between the parameters of optimal classifier and the parameters of our classifier.

In this experiment, we set $f_c = 200$, $p_{noise} = 0.1$, $D = 10$, and $d = 1$. The optimal parameters are defined by the parameter vector of the hyperplane as $< w_0, w_1, \ldots, w_D >$. For the convenience of illustration, we normalize this vector each time after we change it.

To illustrate the result of our algorithm, Figure 1 depicts the evolutions of parameters in the first four dimensions, where the blue solid lines represent the changes of optimal parameters and the red cross markers represent the estimated values of the parameter in the corresponding dimension just before each time concept drift happens. The concept drift happened for 100 times in the experiment. Note that the trend of parameters appears to be curve instead of straight line because they are normalized after we change their values each time. The curves show our algorithm can track the optimal parameters in each dimension very closely.

Figure 2 depicts more detailed local changes of parameters in one dimension over 1000 time stamps, where blue line represents the optimal parameter and red cross markers represent the output of our algorithm in each time stamp. As we can see, the occurrence of concept drift each time can cause the vibration of the approximate value in output but it eventually approximated to the optimal parameter closely after being adjusted at each time stamp.

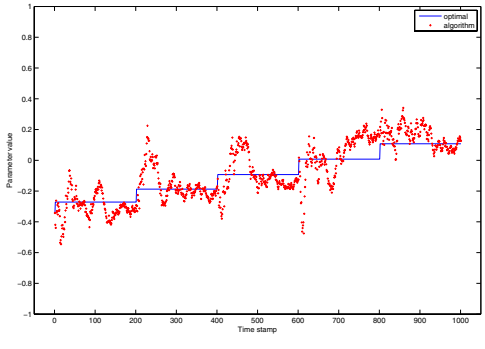

Fig. 2. The local changes of parameters in one dimension

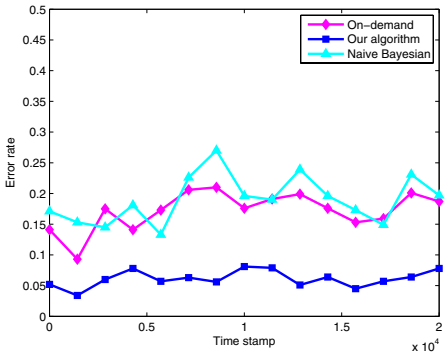

Fig. 3. The error rate in classification

We also compare our algorithm with the On-Demand-Stream classifier [8], which is based on the rule of the nearest neighborhood and use an adaptive sliding window scheme, and a naive Bayesian classifier with a size-fixed sliding window. Table II records the average accuracy of both algorithm in seven synthetic data sets with the same parameter setting. The error rate in classification for the sensor data set is illustrated in Figure 3. As we can see, our algorithm achieves higher classification accuracy in all data sets.

C. Sensitivity

In this part, we show the influences of the frequency of concept drift and the intensity of concept drift on the performance of our algorithm.

Seven data sets with increasing intensity of concept drift and other seven data sets with increasing frequency of concept drift are generated with the same setting. Table III and Table IV give the average classification accuracy on each data set. Both results show no significant decline in classification accuracy with different setting of concept drift, which means our classifier can be adapted to concept drift very well.

To show the influence of the buffer size on the performance, we set different buffer size to compare the classification accuracy in the above 14 synthetic data sets. Table V shows that the performance is stable when the size of buffer is varied.

TABLE II
CLASSIFICATION ACCURACY

Classifier	$Dataset_1$	$Dataset_2$	$Dataset_3$	$Dataset_4$	$Dataset_5$	$Dataset_6$	$Dataset_7$
Ours	0.9341	0.9436	0.9430	0.9536	0.9441	0.9366	0.9511
On-Demand	0.8707	0.8808	0.8844	0.8783	0.8809	0.8614	0.8774
Naive Bayesian	0.8241	0.8201	0.8457	0.8433	0.8503	0.8277	0.8394

TABLE III
SENSITIVITY TO THE INTENSITY OF CONCEPT DRIFT

	$Dataset_1$	$Dataset_2$	$Dataset_3$	$Dataset_4$	$Dataset_5$	$Dataset_6$	$Dataset_7$
intensity	1.0000	1.6667	2.3333	3.0000	3.6667	4.3333	5.0000
accuracy	0.9727	0.9712	0.9606	0.9648	0.9688	0.9614	0.9560

TABLE IV
SENSITIVITY TO THE FREQUENCY OF CONCEPT DRIFT

	$Dataset_1$	$Dataset_2$	$Dataset_3$	$Dataset_4$	$Dataset_5$	$Dataset_6$	$Dataset_7$
frequency	350	300	250	200	150	100	50
accuracy	0.9679	0.9387	0.9545	0.9294	0.9126	0.9505	0.9476

TABLE V
SENSITIVITY TO THE SIZE OF BUFFER

intensity	$Dataset_1$ 1.0000	$Dataset_2$ 1.6667	$Dataset_3$ 2.3333	$Dataset_4$ 3.0000	$Dataset_5$ 3.6667	$Dataset_6$ 4.3333	$Dataset_7$ 5.0000
200	0.9727	0.9712	0.9606	0.9648	0.9688	0.9614	0.9560
400	0.9729	0.9737	0.9645	0.9644	0.9681	0.9621	0.9476
600	0.9679	0.9787	0.9745	0.9694	0.9726	0.9555	0.9576
800	0.9679	0.9745	0.9695	0.9695	0.9696	0.9705	0.9621
1000	0.9739	0.9784	0.9545	0.9794	0.9586	0.9505	0.9676
buffer size							
frequency	$Dataset_1$ 350	$Dataset_2$ 300	$Dataset_3$ 250	$Dataset_4$ 200	$Dataset_5$ 150	$Dataset_6$ 100	$Dataset_7$ 50
200	0.9679	0.9387	0.9545	0.9294	0.9126	0.9505	0.9476
400	0.9678	0.9387	0.9561	0.9360	0.9244	0.9612	0.9431
600	0.9682	0.9325	0.9578	0.9263	0.9124	0.9435	0.9596
800	0.9519	0.9355	0.9574	0.9216	0.9282	0.9568	0.9419
1000	0.9636	0.9362	0.9498	0.9272	0.9309	0.9544	0.9436
buffer size							

VIII. CONCLUSION

The main contribution of this paper is having proposed a framework of modeling concept drift explicitly as the changes of the optimal parameters in a dynamical probabilistic model. With this probabilistic model, we transform the training problem into a filtering problem. We then approximate the optimal parameters by using the extended Kalman filter method. Under this framework, we can use different discriminative models and give them adaptability to concept drift by combining Kalman filter scheme. As evidenced by the empirical results, the proposed method is able to track the optimal parameters very closely and achieves high accuracy with low time complexity. In addition, it is robust to the intensity and frequency of concept drift.

IX. ACKNOWLEDGMENTS

This work is supported in part by NSFC grants 60673103 and 60721061.

REFERENCES

[1] Welch, G., Bishop, G.: An Introduction to the Kalman Filter. Technical report, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, (1995)

[2] Bishop, C.M.: Pattern Recognition and Machine Learning. Springer-Verlag, Berlin Heidelberg New York (2006)

[3] Coleman, T.F., Li, Y.: An Interior, Trust Region Approach for Nonlinear Minimization Subject to Bounds. SIAM Journal on Optimization, Vol. 6. (1996) 418-445

[4] Gao, J., Fan, W., Han, J., Philip, S.Yu.: A General Framework for Mining Concept-Drifting Data Streams with Skewed Distributions. SIAM International Conference on Data Mining. (2007)

[5] Hulten, G., Spencer, L., Domingos, P.: Mining Time-Changing Data Streams. ACM SIGKDD. (2001)

[6] Wang, H., Fan, W., Philip, S.Yu., Han, J.: Mining Concept-Drifting Data Streams Using Ensemble Classifiers. ACM SIGKDD. (2003)

[7] Wang, H., Yin, J., Pei, J., Philip, S.Yu., Jeffrey, X.Yu.: Supressing Model Overfitting in Mining Concept-Drifting Data Streams. ACM SIGKDD. (2006)

[8] Aggarwal, C.C., Han, J., Philip, S.Yu.: On Demand Classification of Data Streams. ACM SIGKDD. (2004)

[9] Yang, Y., Wu, X., Zhu, X.: Combining Proactive and Reactive Predictions for Data Streams. ACM SIGKDD. (2005)

[10] Jordan, M. (ed.): Learning in Graphical Models. MIT Press, Cambridge, MA (1999)

[11] Albert Bifet, Ricard Gavald: Kalman Filters and Adaptive Windows for Learning in Data Streams. 29-40, Proc. Discovery Science. (2006)

[12] Arash Aghassi: Investigation of Qualitative Methods for Diagnosis of Poor Bit Performance Using Surface Drilling Parameters. Thesis, Louisiana State University. (2003)