Early Drift Detection Method *

Manuel Baena-García[1], José del Campo-Ávila[1], Raúl Fidalgo[1], Albert Bifet[2], Ricard Gavaldà[2], and Rafael Morales-Bueno[1]

[1] Departamento de Lenguajes y Ciencias de la Computación
E.T.S. Ingeniería Informática. Universidad de Málaga, Spain
{mbaena, jcampo, rfm, morales}@lcc.uma.es
[2] Universitat Politècnica de Catalunya, Spain
{abifet, gavalda}@lsi.upc.edu

Abstract. An emerging problem in Data Streams is the detection of concept drift. This problem is aggravated when the drift is gradual over time. In this work we define a method for detecting concept drift, even in the case of slow gradual change. It is based on the estimated distribution of the distances between classification errors. The proposed method can be used with any learning algorithm in two ways: using it as a wrapper of a batch learning algorithm or implementing it inside an incremental and online algorithm. The experimentation results compare our method (EDDM) with a similar one (DDM). Latter uses the error-rate instead of distance-error-rate.

1 Introduction

Many approaches in machine learning assume that training data has been generated from a stationary source. This assumption is likely to be false if the data has been collected over a long period of time, as it is often the case: it is likely that the distribution that generates the examples changes over time. In this case, change detection becomes a necessity. Real applications examples are user modelling, monitoring in biomedicine and industrial processes, fault detection and diagnosis, safety of complex systems, etc.

We are interested in drift detection over data streams. Data streams are unbounded sequence of examples received at so high a rate that each one can be read at most once [1]. So, we can not store all the examples in memory, only partially at maximum, and we can not spend so much time processing data, due to the high rate that it arrives.

In this work we present Early Drift Detection Method, a method to detect concept drift, that gets good results with slow gradual changes. Abrupt changes are easier to detect by current methods, but difficulties arise with the slow gradual changes. Our method uses the distance between classification errors (number of examples between two classification errors) to detect change, instead of using

* This work has been partially supported by the FPI program and the MOISES-TA project, number TIN2005-08832-C03, of the MEC, Spain.

errors classifications (as it is done in previous work [2]). It detects change faster, without increasing the rate of false positives, and it is able to detect slow gradual changes.

The paper is organised as follows. The next section presents related work in detecting concept drifting. In section 3 we present our method, Early Drift Detection Method. In Section 4 we evaluate the method and Section 5 concludes the paper and present future work.

2 Related Work

To deal with change over time, most previous work has been classified observing if they use full, partial or not examples memory.

The partial memory methods used variations of the sliding-window idea: at every moment, one window (or more) containing the most recently read examples is kept, and only those are considered relevant for learning. A critical point in any such strategy is the choice of a window size. The easiest strategy is deciding (or asking the user for) a window size W and keeping it fixed through the execution of the algorithm (see e.g. [3–5]). In order to detect change, one can keep a reference window with data from the past, also of some fixed size, and decide that change has occurred if some statistical test indicates that the distributions in the reference and current windows differ.

Another approach, using no examples memory, only aggregates, applies a "decay function" to examples so that they become less important over time [6].

Other approach to detect concept drift monitors the values of three performance indicators [7]: accuracy, recall and precision over time. Then they are compared with a confidence interval of standard sample errors for a moving average value (using the last M batches) of each particular indicator. The key idea is to select the window size so that the estimated generalisation error on new examples is minimised. This approach uses unlabelled data to reduce the need for labelled data, it doesn't require complicated parameterization and it works effectively and efficiently in practise.

A new method to detect changes in the distribution of the training examples monitors the online error-rate of the algorithm [2]. In this method learning takes place in a sequence of trials. When a new training example is available, it is classified using the current model. The method controls the trace of the online error of the algorithm. For the actual context they define a warning level, and a drift level. A new context is declared, if in a sequence of examples, the error increases reaching the warning level at example k_w, and the drift level at example k_d. They take this is as an indication of a change in the distribution of the examples. It uses 6σ ideas, well known in quality theory.

Our method it is based on the latter, but taking into account distances between classification errors as it is presented in the next section.

3 Drift Detection Methods

We consider that the examples arrive one at a time, but it would be easy to assume that the examples arrive in bundles. In online learning approach, the decision model must make a prediction when an example becomes available. Once the prediction has been made, the system can learn from the example (using the attributes and the class) and incorporate it to the learning model. Examples can be represented using pairs $(\overrightarrow{x}, y)$ where $\overrightarrow{x}$ is the vector of values for different attributes and y is the class label. Thus, i-th example will be represented by $(\overrightarrow{x}_i, y_i)$. When the current model makes a prediction (y'_i), it can be correct $(y_i = y'_i)$ or not $(y_i \neq y'_i)$.

3.1 DDM: Drift Detection Method [2]

There are approaches that pay attention to the number of errors produced by the learning model during prediction. The drift detection method (DDM) proposed by Gama et al. [2] uses a binomial distribution. That distribution gives the general form of the probability for the random variable that represents the number of errors in a sample of n examples. For each point i in the sequence that is being sampled, the error rate is the probability of missclassifying (p_i), with standard deviation given by $s_i = \sqrt{p_i(1-p_i)/i}$. They assume (as states the PAC learning model [8]) that the error rate of the learning algorithm (p_i) will decrease while the number of examples increases if the distribution of the examples is stationary. A significant increase in the error of the algorithm, suggests that the class distribution is changing and, hence, the actual decision model is supposed to be inappropriate. Thus, they store the values of p_i and s_i when $p_i + s_i$ reaches its minimum value during the process (obtaining p_{min} and s_{min}). And it checks when the following conditions triggers:

- $p_i + s_i \geq p_{min} + 2 \cdot s_{min}$ for the warning level. Beyond this level, the examples are stored in anticipation of a possible change of context.
- $p_i + s_i \geq p_{min} + 3 \cdot s_{min}$ for the drift level. Beyond this level the concept drift is supposed to be true, the model induced by the learning method is reset and a new model is learnt using the examples stored since the warning level triggered. The values for p_{min} and s_{min} are reset too.

This approach has a good behaviour detecting abrupt changes and gradual changes when the gradual change is not very slow, but it has difficulties when the change is slowly gradual. In that case, the examples will be stored for long time, the drift level can take too much time to trigger and the examples memory can be exceeded.

3.2 EDDM: Early Drift Detection Method

The method that we propose in this paper, called Early Drift Detection Method (EDDM), has been developed to improve the detection in presence of gradual

concept drift. At the same time, it keeps a good performance with abrupt concept drift. The basic idea is to consider the distance between two errors classification instead of considering only the number of errors. While the learning method is learning, it will improve the predictions and the distance between two errors will increase. We can calculate the average distance between two errors (p'_i) and its standard deviation (s'_i). What we store are the values of p'_i and s'_i when $p'_i + 2 \cdot s'_i$ reaches its maximum value (obtaining p'_{max} and s'_{max}). Thus, the value of $p'_{max} + 2 \cdot s'_{max}$ corresponds with the point where the distribution of distances between errors is maximum. This point is reached when the model that it is being induced best approximates the current concepts in the dataset.

Our method defines two thresholds too:

- $(p'_i + 2 \cdot s'_i)/(p'_{max} + 2 \cdot s'_{max}) < \alpha$ for the warning level. Beyond this level, the examples are stored in advance of a possible change of context.
- $(p'_i + 2 \cdot s'_i)/(p'_{max} + 2 \cdot s'_{max}) < \beta$ for the drift level. Beyond this level the concept drift is supposed to be true, the model induced by the learning method is reset and a new model is learnt using the examples stored since the warning level triggered. The values for p'_{max} and s'_{max} are reset too.

The method considers the thresholds and searches for a concept drift when a minimum of 30 errors have happened (note that it could appear a large amount of examples between 30 classification errors). After occurring 30 classification errors, the method uses the thresholds to detect when a concept drift happens. We have selected 30 classification errors because we want to estimate the distribution of the distances between two consecutive errors and compare it with future distributions in order to find differences. Thus, $p'_{max} + 2 \cdot s'_{max}$ represents the 95% of the distribution. For the experimental section, the values used for α and β have been set to 0.95 and 0.90. These values have been determined after some experimentation.

If the similarity between the actual value of $p'_i + 2 \cdot s'_i$ and the maximum value ($p'_{max} + 2 \cdot s'_{max}$) increase over the warning threshold, the stored examples are removed and the method returns to normality.

4 Experiments And Results

In this section we describe the evaluation of the proposed method: EDDM. The evaluation is similar to the one proposed in [2]. We have used three distinct learning algorithms with the drift detection methods: a decision tree and two nearest-neighbourhood learning algorithms. These learning algorithms use different representations to generalise examples. The decision tree uses DNF to represent generalisation of the examples. The nearest-neighbourhood learning algorithms uses examples to describe the induced knowledge. We use the weka implementation [9] of these learning algorithms: J48[10] (C4.5, decision tree), IB1[11] (nearest-neighbourhood, it is not able to deal with noise) and NNge[12] (nearest-neighbourhood with generalisation). We have used four artificial datasets previously used in concept drift detection [13], a new data set with

very slow gradual change and a real-world problem [14]. As we want to know how our algorithms work in different conditions, we have chosen those artificial datasets that have several different characteristics - abrupt and gradual drift, presence and absence of noise, presence of irrelevant and symbolic attributes, numerical and mixed data descriptions.

4.1 Artificial Datasets

The five artificial datasets used later (Subsections 4.2 and 4.3) are briefly described. All the problems have two classes and each class is represented by 50% of the examples in each context. To ensure a stable learning environment within each context, the positive and negative examples in the training set are alternated. The number of examples in each concept is 1000, except in Sine1g that have 2000 examples in each concept and 1000 examples to transit from one concept to another.

- SINE1. Abrupt concept drift, noise-free examples. The dataset has two relevant attributes. Each attribute has values uniformly distributed in $[0, 1]$. In the first concept, points that lie below the curve $y = sin(x)$ are classified as positive, otherwise they are labelled as negative. After the concept change the classification is reversed.
- CIRCLES. Gradual concept drift, noise-free examples. The examples are labelled according to a circular function: if an example is inside the circle, then its label is positive, otherwise is negative. The gradual change is achieved by displacing the centre of the circle and growing its size. This dataset has four contexts defined by four circles:

Center	(0.2,0.5)	(0.4,0.5)	(0.6,0.5)	(0.8,0.5)
Radius	0.15	0.2	0.25	0.3

- GAUSS. Abrupt concept drift, noisy examples. The examples are labelled according to two different but overlapped gaussian density functions ($N([0, 0], 1)$ and $N([2, 0], 4)$). The overlapping can be considered as noise. After each context change, the classification is reversed.
- MIXED. Abrupt concept drift, boolean noise-free examples. Two boolean attributes (v, w) and two numeric attributes (x, y). The examples are classified positive if at least two of the three following conditions are satisfied: v, w, $y < 0.5 + 0.3 \sin(3\pi x)$. After each concept change the classification is reversed.
- SINE1G. Very slow gradual drift, noise-free examples. This dataset remains the same as Sine1, but the concept drift is made by gradually choosing of examples from the old and the new concept. So, there is a transition time between concepts. The probability of selecting an example from the old concept becomes lower gradually and the probability of selecting an example from the new concept becomes higher when the transition time is ended.

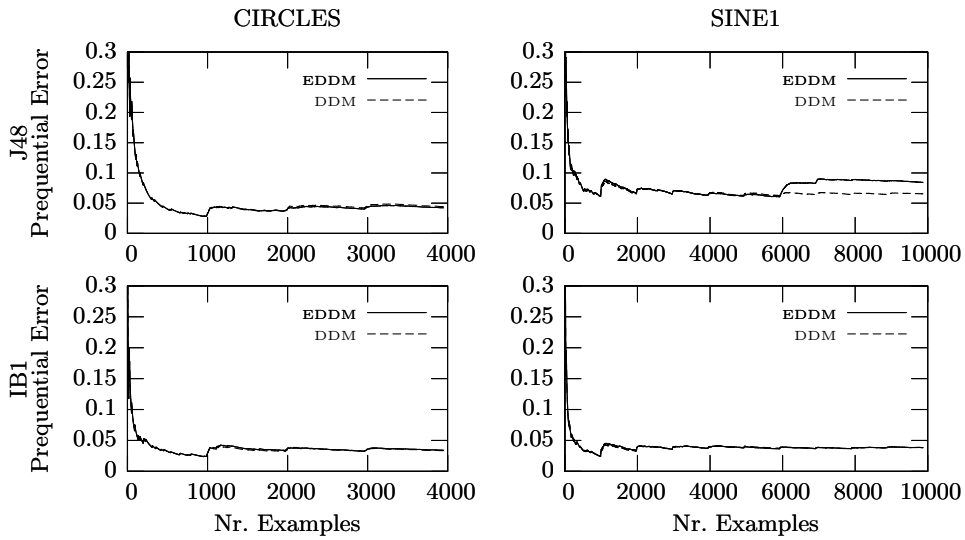

Fig. 1. Prequential error of J48 (up) and IB1 (down) on Circles (left) and Sine1 (right) datasets

4.2 Results On Artificial Domains: Abrupt And Gradual Drifts

The purpose of these experiments is to analyse how the proposed drift detection method works with different learning algorithms, and compare it to the one proposed by Gama et al. [2] (we refer to it as DDM, Drift Detection Method) in terms of prequential [15] error (prequential error is the mean of the classification errors obtained with the examples to be learnt). It is not the aim of this article to compare the results of the different learning algorithms used. Figure 1 shows the prequential error results when Circles and Sine1 datasets are faced with two learning algorithms with the proposed drift detection method (EDDM) and with DDM. Note that changes occur every 1000 examples.

We can observe that prequential error curves obtained by EDDM and DDM on Sine1 dataset (an abrupt changing dataset) are almost the same. When they deal with a gradual dataset (Circles) the results of both methods are very similar, independently of the learning method used.

Table 1 presents the prequential error results and the total number of changes detected by the learning algorithms with both drift detection methods at the end of each dataset. On datasets with abrupt concept change, both algorithms react quickly and reach low error rates. When noise is present, EDDM is more sensitive than DDM, detecting changes and improving the performance when the base algorithm does not support noise (i.e. IB1). This is so because after some time (some number of examples) the base algorithms overfit and the frequency of classification errors increases.

Table 1. Prequential error and total number of changes detected by methods

		EDDM		DDM	
		Prequential	Drifts	Prequential	Drifts
Circle	IB1	0.0340	3	0.0343	3
	J48	0.0421	3	0.0449	3
	NNge	0.0504	4	0.0431	4
Gauss	IB1	0.1927	32	0.1888	9
	J48	0.1736	22	0.1530	9
	NNge	0.1763	31	0.1685	12
Mixed	IB1	0.0322	9	0.0330	10
	J48	0.0425	9	0.0449	11
	NNge	0.0639	10	0.0562	9
Sine1	IB1	0.0376	9	0.0377	9
	J48	0.0847	11	0.0637	10
	NNge	0.0819	14	0.0767	11

Table 2. Prequential error and total number of changes detected by methods in Sine1g dataset

		EDDM		DDM	
		Prequential	Drifts	Prequential	Drifts
Sine1g	IB1	0.1462	50	0.2107	12
	J48	0.1350	34	0.1516	12
	NNge	0.2104	28	0.2327	12

4.3 Results On Artificial Domains: Slow Gradual Drift

There are many real-world problems that have slow gradual drift. In this section we use the Sine1g dataset to illustrate how EDDM works with this kind of change. Figure 2 shows the prequential error curves obtained when EDDM and DDM deal with this dataset. The plots on the left are prequential error calculated with the whole dataset, and the plots on the right are prequential error calculated from scratch after every concept change detected by both methods.

Although the global curves are similar, the local prequential curves show that EDDM reacts before and more times than DDM when a slow concept drift is present. During the transition from the previous concept to the next, EDDM detects repeatedly concept drifts. This is so because the frequency of classification errors continuously increase until the next concept is stable. Meanwhile, DDM shows less sensitivity to this kind of problems, reacting later and less than EDDM. Table 2 presents the final prequential errors and number of drifts obtained by these two methods on Sine1g dataset.

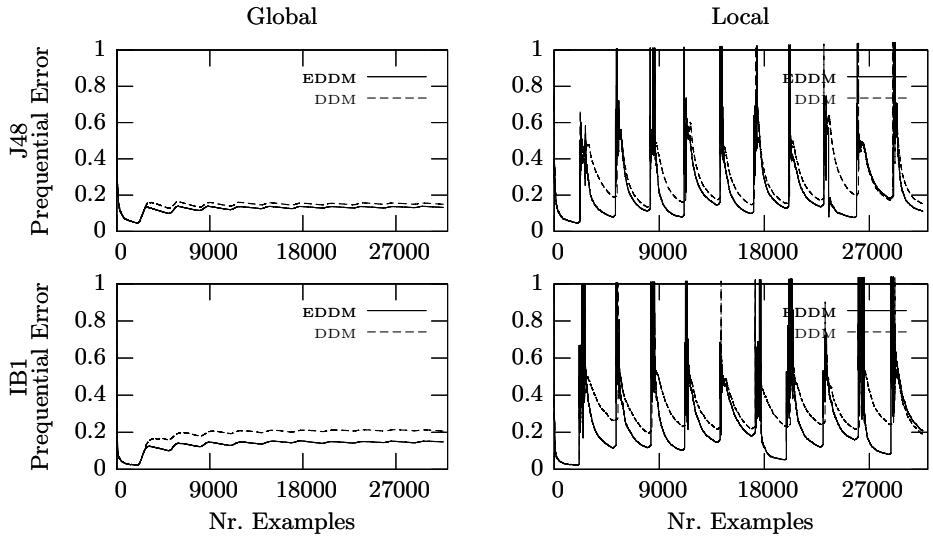

Fig. 2. Prequential global error (left) and local error (right) for EDDM and DDM in Sine1g dataset

4.4 The Electricity Market Dataset

The data used in this experiment was first described by M. Harries [14]. The data was collected from the Australian New South Wales Electricity Market. In this market, the prices are not fixed and they are affected by demand and supply of the market. A factor for the price evolution is the time evolution of the electricity market. During the time period described in the data the electricity market was expanded with the inclusion of adjacent areas. This produced a more elaborated management of the supply. The production surplus of one region could be sold in the adjacent region. A consequence of this expansion was a dampener of the extreme prices. The ELEC2 dataset contains 45312 instances dated from May 1996 to December 1998. Each example of the dataset refers to a period of 30 minutes. Each example on the dataset has 5 fields, the day of week, the time stamp, the NSW electricity demand, the Vic electricity demand, the scheduled electricity transfer between states and the class label.

The class label identifies the change of the price related to a moving average of the last 24 hours. The class level only reflect deviations of the price on a one day average and removes the impact of longer term price trends. The interest of this dataset is that it is a real-world dataset. We do not know when drift occurs or if there is drift. We have considered this problem as a short term prediction: predict the changes in the prices relative to the next 30 minutes.

In Figure 3 we present the traces of the prequential error rate of EDDM and DDM, with the base learning algorithms, through the full ELEC2 dataset. As

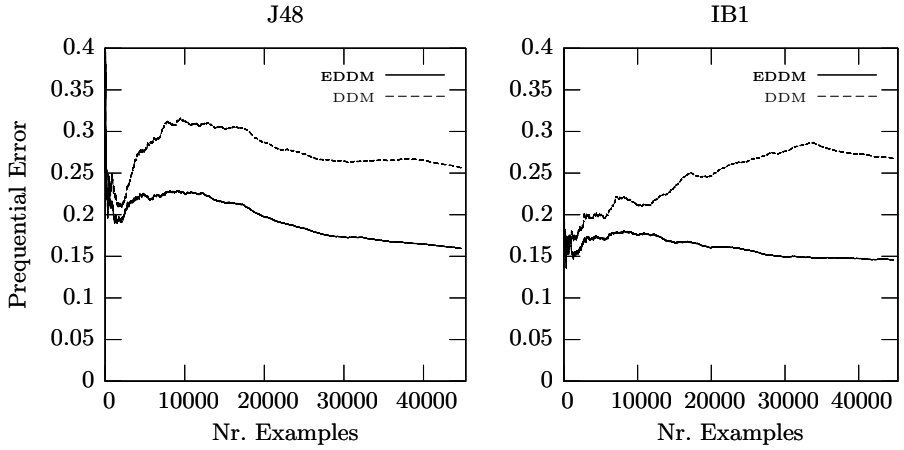

Fig. 3. Prequential error for EDDM and DDM in ELEC2 dataset

Table 3. Prequential error and total number of changes detected by methods in ELEC2 dataset

		EDDM		DDM	
		Prequential	Drifts	Prequential	Drifts
Elect2	IB1	0.1426	171	0.2764	44
	J48	0.1564	187	0.2123	10
	NNge	0.1594	193	0.2110	130

can be seen, EDDM outperforms DDM, detecting concept changes earlier and with a better sensitivity. Table 3 shows the final prequential errors and number of drifts obtained by these two methods on the electricity market dataset.

5 Conclusion

This paper introduces a new method for detecting concept drifts based on the distances between classification errors. This method achieves an early detection in presence of gradual changes, even when that change is very slow. The results that are presented have been obtained after using the method as a wrapper for different learning algorithms, but it would be easy to implement it in a local way inside those algorithms. The experimental evaluation of EDDM illustrates the advantages of using this detection method. As well as obtaining good results detecting concept drifts, this method shows itself as a way to deal with noisy datasets even when the base algorithm is not designed with that aim. When the base algorithm begins to overfit, the frequency of classification errors begins to increase and that is detected by the proposed method.

Out aim of improving EDDM involves some issues where the most important is finding a way to determine the values for the parameters of the method (α and β) in an automatic way.

References

1. Muthukrishnan, S.: Data streams: algorithms and applications. In: Proc. of the 4th annual ACM-SIAM symposium on discrete algorithms. (2003)
2. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. Lecture Notes in Computer Science **3171** (2004)
3. Dong, G., Han, J., Lakshmanan, L.V.S., Pei, J., Wang, H., Yu, P.S.: Online mining of changes from data streams: research problems and preliminary results. In: Proc. of the 2003 ACM SIGMOD Workshop on Management and Processing of Data Streams. (2003)
4. Fan, W.: Streamminer: A classifier ensemble-based engine to mine concept-drifting data streams. In: Proc. of the 30th VLDB Conference. (2004)
5. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proc. 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, ACM Press (2003) 226– 235
6. Cohen, E., Strauss, M.: Maintaining time-decaying stream aggregates. In: Proc. of the 21nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems. (2003)
7. Klinkenberg, R., Joachims, T.: Detecting concept drift with support vector machines. In: Proc. of the 17th Int. Conf. on Machine Learning. (2000) 487 – 494
8. Mitchell, T.: Machine Learning. McGraw Hill (1997)
9. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. 2 edn. Morgan Kaufmann, San Francisco (2005)
10. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
11. Aha, D., Kibler, D.: Instance-based learning algorithms. Machine Learning **6** (1991) 37–66
12. Martin, B.: Instance-based learning : Nearest neighbor with generalization. Master's thesis, University of Waikato, Hamilton, New Zealand (1995)
13. Kubat, M., Widmer, G.: Adapting to drift in continuous domain. In: Proc. of the 8th European Conference on Machine Learning, Springer Verlag (1995) 307–310
14. Harries, M.: Splice-2 comparative evaluation: Electricity pricing. Technical report, The University of South Wales (1999)
15. Dawid, A., Vovk, V.: Prequential probability: Principles and properties (1999)