Online Methods of Learning in Occurrence of Concept Drift

Veena Mittal
PhD Scholar
Deptt.of CSE
FET,MRIU

Indu Kashyap
Associate Professor
Deptt.of CSE
FET,MRIU

ABSTRACT

Due to potentially large number of applications of real-time data stream mining in scientific and business analysis, the real-time data streams mining has drawn attention of many researchers who are working in the area of machine learning and data mining. In many cases, for real-time data stream mining online learning is used. Environments that require online learning are non-stationary and whose underlying distributions may change over time i.e. concept drift, because of which mining of real- time data streams with concept drifts is quite challenging. However, ensemble methods have been suggested for this particular situation. This paper reviews various online methods of drift detection. We also present some results of our experiments that show the comparison of some online drift detection (concept drift) methods.

General Terms

Machine Learning, Data Mining, Online Data Mining, Drift.

Keywords

Concept Drifts, Drift detection algorithms, Online methods of learning.

1. INTRODUCTION

A data stream is an ordered sequence of instances that arrive at a rate that does not allow to store permanently them in memory. Since, data streams are unbounded in size which makes them impossible to process by most data mining approaches. The algorithms required to process the data of data stream model must have a very hard time and space constraints. Following constraints are imposed on data stream model due its characteristics [1].

(i) It is not possible to store all the data from the data stream. Only small summaries of data streams can be computed and stored, and the rest of the information is ignored.

(ii) The arrival speed of data stream tuples forces each particular element to be processed in real time, and then discarded.

(iii) The distribution generating the items can change over time. Thus, data from the past may become irrelevant or even harmful for the current summary.

1.1 Real time data stream

A data source S is real-time data stream source that generates instance vector $\boldsymbol{x}_\mathbf{t} \in \mathfrak{R}^\mathbf{P}$, with P dimensional features where each instance at time $\mathbf{t}$ and having target label $\boldsymbol{y_t}$. Since the values of all features may change drastically with respect to time as compared to the feature values of previously generated instances by the same source S, therefore, classifier learned on previously generated instances may lead high rate of misclassification. Such kind of data sets whose underlying distribution changes with time (i.e. concept drift [6], discussed in next section) makes single classifiers highly inaccurate with time which learned on previously generated instances.

1.2 Concept drift

The concept drift means that properties of the target variable changes over time. It causes problems because the predictions become less accurate as time passes. Often these changes make the data mining model inconsistent for new instances of data. Proper working of such model only dependent on regular updating of the model built on old data with the new data.

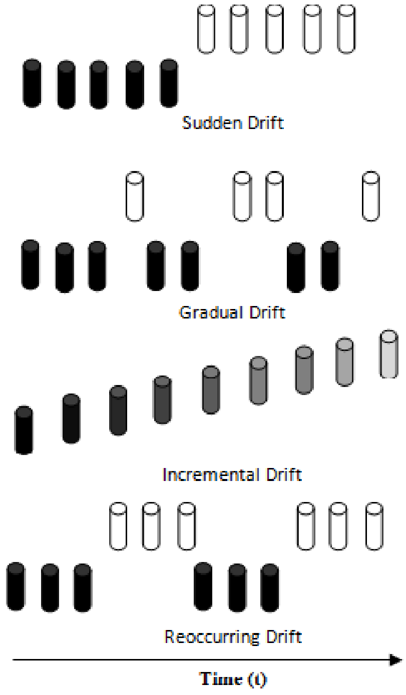


Fig 1: Types of drifts

1.2.1 Types of Drifts

Concepts drifts present in a data stream can be broadly classified into four major categories; (i) Sudden drift (ii) Gradual drift (iii) Incremental drift and (iv) Reoccurring drift.

In a manner to illustrate al these types of drifts in detail, just consider two source of data stream S_a and S_b. The sudden drift occurs when the data stream generated by source S_a is suddenly replaced by S_b. In gradual drift both the data stream sources, S_a and S_b generate data for the non-uniform interval of time alternately. In incremental drift, the underlying distribution of the data stream generated by the source changes in a stepwise fashion as time passes. Whereas in reoccurring concept drift the concepts reapers after some interval of time. Figure 1 depicts all these drifts very clearly.

2. ONLINE METHODS OF LEARNING IN OCCURRENCE OF CONCEPT DRIFT

Algorithms that handles concept drift can be divided in two broader categories: (i) Methods that use approaches to detect drifts [7][8][9] and (ii) Methods that are not explicitly used in any drift detection mechanism [10][11][12]. Both methods, directly or indirectly handle drifts on the basis of accuracy of the current classifiers.

The first approach has a quick response to drifts by discarding the current system and creating a new system on the basis of some measures related to the accuracy of drift detection. However, they may suffer from non-accurate drift detection. The second methods usually assign some weights to each base learner based on its accuracy, which possibly allow pruning and addition of some new classifiers. As it is not very promising to define that when a drift occurred, ensemble members are likely to maintain, which do reflect the new concept very well with weights that are influenced by the old concept, as a result these approaches may take longer time to recover from drifts.

Using the old concept in a manner to aid the learning of new concept is a matter of investigation as none of the above approaches do that. However, there are approaches that use ensembles of classifiers to deal with drifts without having any proper study and justification on the success of these approaches.

The remaining sub-sections of this section present some existing approaches and an interesting discussion on their strengths and weaknesses. Some methods of drift detection that are discussed are: Drift Detection Method (DDM) [8], Early Drift Detection Methods (EDDM) [7], Dynamic Weight Majority (DWM) [11], Adaptive expert ensembles [12], The CUSUM Test [15], Exponential Weighted Moving Average Chart (EWMA) [13], and ADaptive Sliding WINdow Algorithm (ADWIN)[14],

2.1 DDM and EDDM

DDM and EDDM both reset the system when drift is detected. The basic idea behind DDM is that as when the underlying distribution of example is stationary the error rate of a learning example decreases as the number of examples increases. So the sufficient amount of increase in error rate indicates the change of underlying distribution and hence to the presence of concept drift. The ensembles methods that do not detect drifts handle them by pruning members of an ensemble with high training error are also using this concept.

In DDM, binomial distribution is used to model the error in the sample of n examples. The error rate is the number of occurences of misclassification *(p_t)* for each point t in the sequence that is sampled, for which standard deviation is given by

$$s_t = \sqrt{p_t\,(1 - p_t)/t} \qquad (1)$$

The minimum error rate p_{min} and standard deviation s_{min} obtained so far are stored in DDM. DDM then checks for following two conditions to occur:

(i) $if\ p_t + s_t \geq p_{min} + \ 2s_{min}$
$the\ warning\ level\ is\ triggred.$ (2)

(ii) $if p_t + s_t \geq p_{min} + 3s_{min}$
$concept\ drift\ is\ supposed\ to\ be\ true$ (3)

Under case (ii) the model that is derived by the learning method is reset and a new model is learned using the examples stored since the warning level triggered. The values for p_{min} and s_{min} are reset.

The EDDM is an improvement of DDM; It is based on the estimated distribution of the distances between classification errors. In EDDM the average distance among two errors p'_t and its standard deviation s'_t is calculated. There maximum values ($p'_{max}\ and\ s'_{max}$) so far are stored. EDDM checks for two conditions:

(i) $if\ (p'_t +\ 2s'_t)/(p'_{max} +\ 2s'_{max}\ ) < \alpha,$
$the\ warning\ level\ is\ triggered.$ (4)

(ii) $if\ (p'_t +\ 2s'_t)/(p'_{max} +\ 2s'_{max}\ ) < \beta,\ \alpha > \beta,$
$there\ is\ concept\ drift$ (5)

Both DDM and EDDM react spontaneously to drifts whenever they detected. As the model gets reset whenever drift is detected, hence, these methods cannot adopt any previously learned knowledge.

DDM works well for detecting abrupt changes and reasonably fast changes, but it has difficulties detecting slow,gradual changes. In the latter case, examples will be stored for long periods of time, the drift level can take very much time to trigger and the examples in memory may overflow [16].

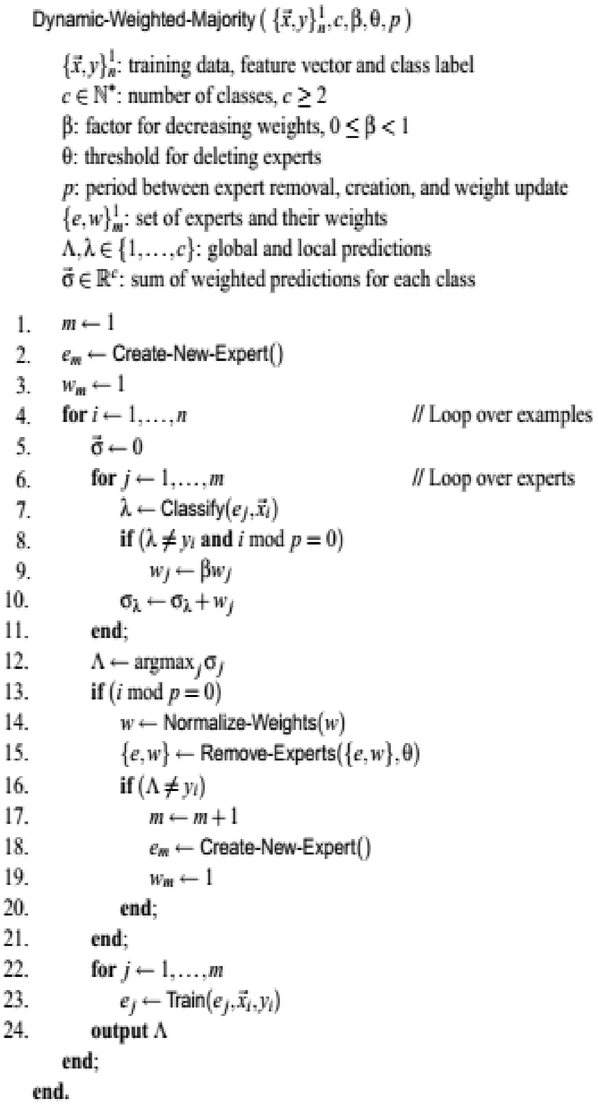
2.2 DWM and AddExp

DWM [11] is one of the best algorithms used for concept drift detection. In DWM each ensemble member has assigned a weight which starts with a value of 1 and when wrong prediction is made by the member it gets reduced by a multiplicative constant $\beta\ (0 \leq \beta < 1)$ as in Weighted Majority of [17]. The updating of the weight of a member of an ensemble is done only in time steps multiple of p *(P is a pre-defined value).*

In DWM, if last training example gets misclassified at every p training example by the ensemble, after the weight update a new classifier is added. All those members of the ensemble are removed whose weight is less than a predefined value. Hence, when drift occurs, new concepts are learned by creating new ensemble members without forgetting the old one. All those members who take long time to forget the previous concept can be dropped from the set. And all remaining members are now trained with the new concept. Before adding any new member to the ensemble after every p training examples, the weight of all members are normalized.

AddExp [12] method is quite similar to DWM. In AddExp, when the collective prediction of the ensemble is wrong a new classifier is added. The summation of the weights of all members of the ensembles is multiplied by a constant $\gamma\ (0 \leq \gamma \leq 1)$ to represent the collective weight of a classifier. Whenever, an ensemble member makes a wrong prediction, then at every time step, the weight of ensemble member is multiplied by a constant $\beta\ (0 \leq \beta < 1)$. Pruning is required in AddExp, as without pruning the ensemble size become excessively large enough and impractical to use. For this, once the maximum size is achieved, a new member is added only after deleting the lowest weight member or the oldest member.

In both of the methods, the diversity is not needed to built -in prior as the base learners are trained with different sequences of data and ensemble members are created in a different moment. In DWM, the deletion or addition of classifiers is possible at only p training examples. The pruning mechanism of DWM does not sure about the number of experts created. Both DWM and AddExp do not provide any mechanism to deal with recurrent drifts.



```
Dynamic-Weighted-Majority ( $\{\vec{x},y\}_n^1, c, \beta, \theta, p$ )
    $\{\vec{x},y\}_n^1$: training data, feature vector and class label
    $c \in \mathbb{N}^*$: number of classes, $c \geq 2$
    $\beta$: factor for decreasing weights, $0 \leq \beta < 1$
    $\theta$: threshold for deleting experts
    $p$: period between expert removal, creation, and weight update
    $\{e,w\}_m^1$: set of experts and their weights
    $\Lambda, \lambda \in \{1,\ldots,c\}$: global and local predictions
    $\vec{\sigma} \in \mathbb{R}^c$: sum of weighted predictions for each class

    $m \leftarrow 1$
    $e_m \leftarrow$ Create-New-Expert()
    $w_m \leftarrow 1$
    for $i \leftarrow 1,\ldots,n$                          // Loop over examples
        $\vec{\sigma} \leftarrow 0$
        for $j \leftarrow 1,\ldots,m$                      // Loop over experts
            $\lambda \leftarrow$ Classify($e_j, \vec{x}_i$)
            if ($\lambda \neq y_i$ and $i$ mod $p = 0$)
                $w_j \leftarrow \beta w_j$
            $\sigma_\lambda \leftarrow \sigma_\lambda + w_j$
        end;
        $\Lambda \leftarrow \text{argmax}_j \sigma_j$
        if ($i$ mod $p = 0$)
            $w \leftarrow$ Normalize-Weights($w$)
            $\{e,w\} \leftarrow$ Remove-Experts($\{e,w\}, \theta$)
            if ($\Lambda \neq y_i$)
                $m \leftarrow m + 1$
                $e_m \leftarrow$ Create-New-Expert()
                $w_m \leftarrow 1$
            end;
        end;
        for $j \leftarrow 1,\ldots,m$
            $e_j \leftarrow$ Train($e_j, \vec{x}_i, y_i$)
        output $\Lambda$
    end;
end.
```

Fig 2: DWM Algorithm

2.3 EWMA Chart

EWMA is similar to DDM; It is a new method for drift detection that uses an exponentially weighted moving chart to update the estimate of error faster. EWMA is a single pass method with a computational complexity of O(1). It was originally proposed by [18]. EWMA charts are used for detecting change or increase in the mean of a sequence of random variables X_1, X_2,…..X_n. Let μ_0 and μ_1 represent the common mean of the random variables before and after the change respectively. It is assumed that both μ_0 and σ_X (standard deviation of the stream) are known. Let μ_t is used to represent mean at time t whose estimated value is represented by Z_t. Then the estimator of μ_t i.e. Z_t is defined as:

$$Z_0 = \mu_0 \qquad (6)$$

$$Z_t = (1-\lambda)Z_{t-1} + \lambda X_t, t > 0 \qquad (7)$$

The EWMA estimator provides a way to estimate the recent estimate of μ_t while down weighing the older data. In [18], it is mentioned that the mean and the standard deviation of Z_t are:

$$\mu_{Z_t} = \mu_t \qquad (9)$$

$$\sigma_{Z_t} = \sqrt{\frac{\lambda}{2-\lambda}(1-(1-\lambda)^{2t})\sigma_X} \qquad (10).$$

The value of Z_t fluctuates between μ_0 and μ_1. The change occurred when:

$$Z_t > \mu_0 + L\sigma_{Z_t} \qquad (11)$$

Where L, is called ***control parameter*** that determines how far Z_t diverse from μ_0 before a change occurs. **ARL_0** (Average Run Length) is performance measure is the expected time between false positive detections. L is chosen so that the expected time between false positives is equal to some desired value for ARL_0 [18].

2.3.1 EWMA for Concept Drift Detection

Let p_t, is a Bernoulli parameter representing the probability of misclassifying a point at time *t*. Then p_t may attain only two values p_0 or p_1, which are the probabilities before and after the drift respectively. Considering Bernoulli distribution σ_X is now dependent on p_t and both of them are assumed to known in advance. Now the EWMA estimator Z_t already defined in (11) can be redefined according to Bernoulli distribution that gives the pre-change standard deviation of the EWMA estimator as:

$$\sigma_{Z_t}^2 = \sqrt{p_0(1-p_0)\frac{\lambda}{2-\lambda}(1-(1-\lambda)^{2t})} \qquad (12)$$

The value of λ = 0.2 is suggested by the author [18]. In addition to EWMA estimator Z_t, a new estimator of p_0 which we denote by $\hat{p}_{0,t}$, defined as:

$$\hat{p}_{0,t} = \frac{1}{t}\sum_{i-1}^{t} X_i = \frac{t-1}{t}\hat{p}_{0,t-1} + \frac{1}{t}X_t \qquad (13)$$

EWMA procedure flags for a change whenever the distance between these two estimators exceeds a certain threshold, i.e. when

$$Z_t > \hat{p}_{0,t} + L\sigma_{Z_t} \qquad (14)$$

2.4 ADWIN: ADaptive SlidingWINdow Algorithm

ADWIN maintain a variable length window of recently seen items. The maximum length of the window is decided by the assumption that the average value inside the window does not change throughout the length. In other words an older fragment of the window is dropped only when there is an enough evidence for the change of average value. It has two consequences: one, that change can reliably be declared

Whenever the window shrinks; and two, that at any time the average over the existing window can be reliably taken as an estimate of the current average in the stream (barring a very small or very recent change that is still not statistically visible). These two points appears in [14] in a formal theorem.

ADWIN is data parameter- and assumption-free in the sense that it automatically detects and adapts to the current rate of change. Its only parameter is a confidence bound δ, indicating how confident we want to be in the algorithm's output, inherent to all algorithms dealing with random processes

3. EXPERIMENTS AND RESULTS

All experiments are carried out using MOA (Massive Online Analysis) [5][19]. We have considered two variations of concept drifts (i) Gradual drift and (ii) Abrupt drift on four drift detection algorithms i.e. DDM, EDDM, **EWMA** Chart, and ADWIN. All of these methods are analyzed for mean perdition error and number of drifts detected. Finally, the graphs are plotted using MATLAB.

Figure 3(a)-3(d), represents the average of the prediction error of all four drift detection algorithms on data streams generated by gradual drift generator. All experiments are carried out of 1000 instances.

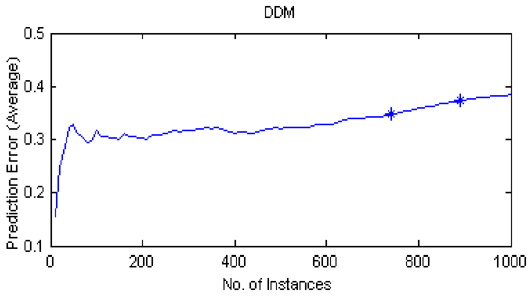


Fig 3(a): Prediction error average of DDM Algorithm on gradual data stream

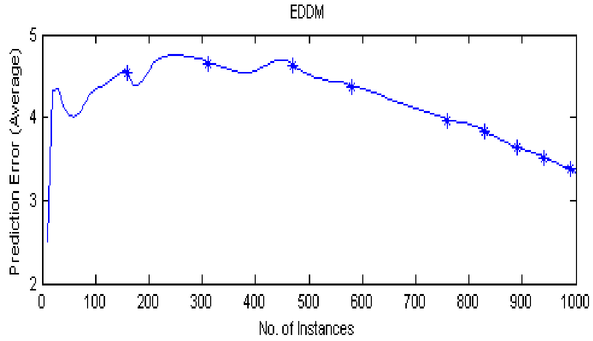


Fig 3(b): Prediction error average of EDDM algorithm on gradual data stream

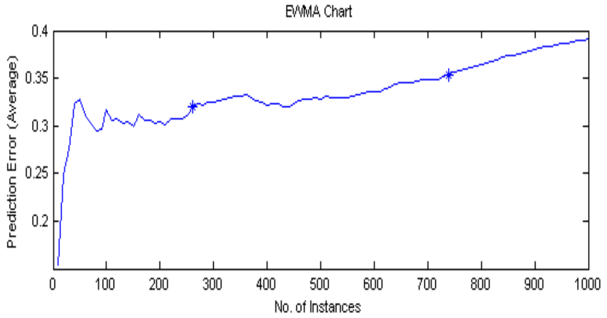


Fig 3(c): Prediction error average of EWMA Chart algorithm on gradual data stream

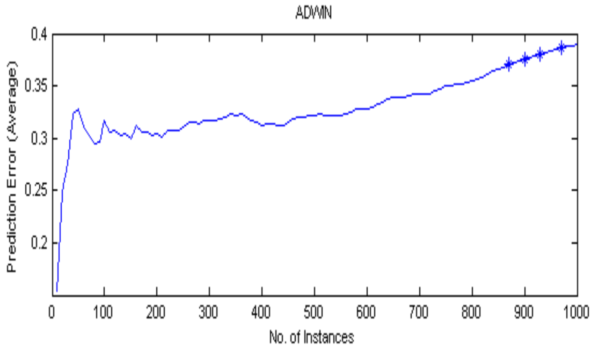


Fig 3(d): Prediction error average of ADWIN algorithm on gradual data stream

Figure 4(a)-4(d), represents the average of the prediction error of all four drift detection algorithms on data streams generated by abrupt drift generator. All experiments are carried on 1000 instances.

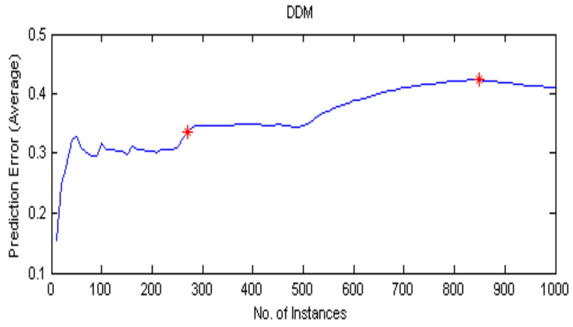


Fig 4(a): Prediction error average of DDM algorithm on abrupt data stream

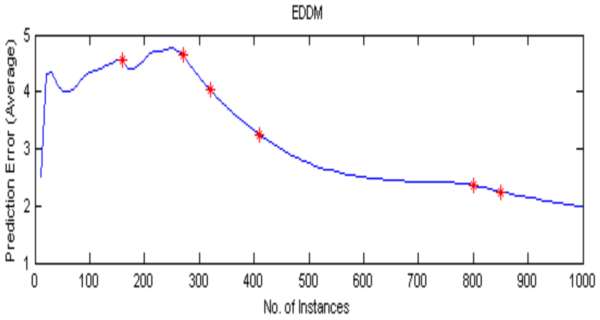


Fig 4(b): Prediction error average of EDDM algorithm on abrupt data stream

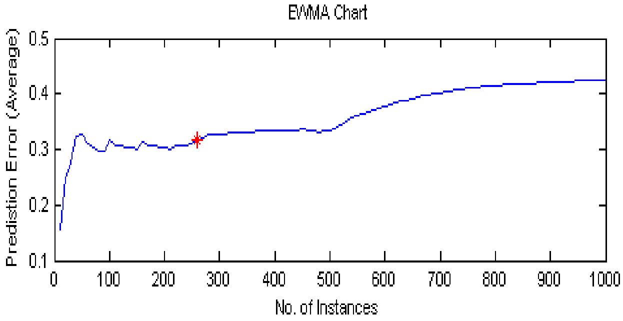


Fig 4(c): Prediction error average of EWMA Chart algorithm on abrupt data stream

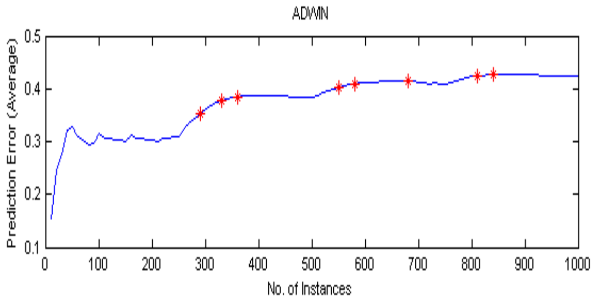


Fig 4(d): Prediction error average of ADWIN algorithm on abrupt data stream

Table 1 summarizes the analysis of all experiments carried out for DDM, EDDM, EWMA Chart and ADWIN, under two variation of data stream generators i.e. gradual drift generators and abrupt drift generator. In the table mean of prediction error for all 1000 instances are mentioned with the total number of drift detected in the data stream for both gradual drift generators and abrupt drift stream generator source.

Table 1: Comparison of DDM, EDDM, EWMA Chart and ADWIN for prediction error

Type of Concept Drift	Drift Detection Method	Prediction Error Mean	Nos. of Drifts Detected
Gradual Drift	DDM	0.3297	2
	EDDM	4.2261	9
	EWMA Chart	0.3351	2
	ADWIN	0.3298	4
Abrupt Drift	DDM	0.3631	2
	EDDM	3.1526	6
	EWMA Chart	0.3575	1
	ADWIN	0.3762	8

4. CONCLUSIONS

Change detection is a significant element of systems that need to adapt to changes in their input data. Both DDM and EDDM react spontaneously to drifts whenever they detected. DDM works well for detecting abrupt changes and reasonably fast changes, but it has difficulties detecting slow, gradual changes. EDDM has a high number of false positives and performs worse than DDM in our experiments. However, if the data size increased, it has been observed that the performance of EDDM gets increased with data instances. ADWIN is the methods with fewer false positives. EWMA Chart seems that this is a very low value compared with other change detectors. ADWIN seems to be the algorithm with the best results.

5. REFERENCES

[1] Albert Bifet. *Adaptive learning and mining for data streams and frequent patterns*. PhD thesis, UniversitatPolit´ecnica de Catalunya, 2009.

[2] Albert Bifet and Richard Kirkby, *DATA STREAM MINING: A Practical Approach*, August 2009.

[3] Alexey Tsymbal, *"The problem of concept drift: Definitions and related work"*, Technical report, Department of Computer Science, Trinity College, 2004.

[4] Indre Zliobaite. Learning under Concept Drift: an Overview. Tech. Report, Vilnius University, Faculty of Mathematics and Informatic, 2010

[5] MOA: Massive Online Analysis, a Framework for Stream Classification and Clustering Albert Bifet, Geoff Holmes, Bernhard Pfahringer, Philipp Kranen, Hardy Kremer, Timm Jansen, Thomas Seidl. Journal of Machine Learning Research (JMLR) Workshop and Conference Proceedings. Volume 11: Workshop on Applications of Pattern Analysis (2010).

[6] Minku, Leandro Lei, Online ensemble learning in the presence of concept drift, (2011) Ph.D. thesis, University of Birmingham

[7] Baena-Garc´ıa, M., Del Campo-Avila, J., Fidalgo, R. and Bifet, A. (2006). Early drift detection method, Proceedings of the Forth ECML PKDD International Workshop on Knowledge Discovery From Data Streams (IWKDDS'06), Berlin, Germany, pp. 77–86.

[8] Gama, J., Medas, P., Castillo, G. and Rodrigues, P. (2004). Learning with drift detection, Proceedings of the Seventh Brazilian Symposium on Artificial Intelligence (SBIA'04) – Lecture Notes in Computer Science, Vol. 3171, Springer, S˜ao Luiz do Maranh˜ao, Brazil, pp. 286–295.

[9] Nishida, K. (2008). Learning and Detecting Concept Drift, PhD thesis, Hokkaido University, Japan.

[10] Stanley, K. O. (2003). Learning concept drift with a committee of decision trees, Technical Report UT-AI-TR-03-302, Department of Computer Sciences, University of Texas at Austin, Austin, USA.

[11] Kolter, J. Z. and Maloof, M. A. (2003). Dynamic weighted majority: A new ensemble method for tracking concept drift, Proceedings of the Third International IEEE Conference on Data Mining (ICDM'03), IEEE Press, Los Alamitos, CA, pp. 123–130.

[12] Kolter, J. Z. and Maloof, M. A. (2005). Using additive expert ensembles to cope with concept drift, Proceedings of the Twenty Second ACM International Conference on Machine Learning(ICML'05), Bonn, Germany, pp. 449–456.

[13] Ross, G.J., Adams, N.M., Tasoulis, D.K., Hand, D.J.: Exponentially weighted moving average charts for detecting concept drift. Pattern Recognition Letters 33(2), 191–198 (2012)

[14] Bifet, A., Gavald`a, R.: Learning from time-changing data with adaptive windowing. In: SIAM International Conference on Data Mining (2007)

[15] Page, E.S.: Continuous inspection schemes. Biometrika 41(1/2), 100–115 (1954)

[16] Albert Bifet, Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Indr˙e Zliobait˙ ˇ e CD-MOA: Change Detection Framework for Massive Online Analysis

[17] Littlestone, N. and Warmuth, M. K. (1994). The weighted majority algorithm, Information and Computation 108: 212–261.

[18] Roberts, S. W., 1959. Control chart tests based on geometric moving averages. Technometrics 42 (1), 97–101.

[19] Albert Bifet, Geoff Holmes, Richard Kirkby and Bernhard Pfahringer, MOA: Massive Online Analysis, Journal of Machine Learning Research 11, 1601-1604. , 2010