Intelligent Data Analysis 20 (2016) 1329–1350
DOI 10.3233/IDA-150207
IOS Press

A novel concept drift detection method in data streams using ensemble classifiers

Mahdie Dehghan, Hamid Beigy* and Poorya ZareMoodi
Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

Abstract. Concept drift, change in the underlying distribution that data points come from, is an inevitable phenomenon in data streams. Due to increase in the number of data streams' applications such as network intrusion detection, weather forecasting, and detection of unconventional behavior in financial transactions; numerous researches have recently been conducted in the area of concept drift detection. An ideal method for concept drift detection should be able to rapidly and correctly identify changes in the underlying distribution of data points and adapt its model as quickly as possible while the memory and processing time is limited. In this paper, we propose a novel explicit method based on ensemble classifiers for detecting concept drift. The method processes samples one by one, and monitors the distribution of ensemble's error in order to detect probable drifts. After detection of a drift, a new classifier will be trained on the new concept in order to keep the model up-to-date. The proposed method has been evaluated on some artificial and real benchmark data sets. The experiments' results show that the proposed method is capable of detecting and adjusting to concept drifts from different types, and it has outperformed well-known state-of-the-art methods. Especially, in the case of high-speed concept drifts.

Keywords: Concept drift, change detection, data stream, online learning, ensemble learning

1. Introduction

According to the challenges that we are faced in data streams, traditional methods of machine learning with assumptions such as the stationary distribution of data and availability of all points for multiple passes through data are not applicable. Data streams are infinite-length, arrive with high-speed, change with evolving patterns, and it is impractical to store them. In applications such as monitoring patients' health, weather forecasting, detecting spam emails, detecting fraud in credit card transactions, detecting unusual behavior in computer networks, telecommunications and financial transactions [1]; we are faced with large amount of data, and an appropriate method is required to handle concept drift rapidly without storing all data.

Concept drift in data streams has a high level of importance especially in the case of classification problems. In this context, the word "concept" means the target class [2] and concept drift refers to changes in the underlying distribution of the target classes. In the real world applications, the underlying distribution of data are not stationary thus previously valid models might lose their efficiency by the passage of time. Concept-evolution is another concept similar to concept drift, and in fact, is the arrival of novel classes, which is an undeniable phenomenon in most real world data streams [3].

*Corresponding author: Hamid Beigy, Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. E-mail: beigy@sharif.edu.

Concept drift has different types and has been categorized in several ways [4–6]. Ref [6] classified concept drift as follows: 1) Sudden drift which is the simplest one. It refers to a sudden replacement of a concept with another one at a specific time. 2) Gradual drift which occurs in a period of time when the data generator is switched from one source to another one. As time goes on, the probability of sampling from the first source decreases gradually, and simultaneously the probability of sampling from the second one increases. At the beginning of occurrence of this drift, the samples from the second source might be considered as random noise. 3) Incremental drift, which is a special case of gradual drift when the concept comes from more than two sources and the probability of choosing each concept, change over time. Therefore, recognizing this type of drift requires considering much more samples than the gradual drift. 4) Reoccurring context, which refers to the reappearance of a concept after a time interval. It should be mentioned that the length of the interval is not known, and it is not necessarily periodic. Recently, there has been an effort to categorize concept drift into mutually exclusive categories based on the number of recurrences, severity, speed, and predictability [7].

An ideal method for detecting concept drift in data streams should be able to: detect all types of drift, have robustness against noise, have sensitivity to real changes, and detect the location of changes. Methods, which have been proposed to handle concept drift, can be categorized into two major groups based on their adoption delay to a new concept [2,8]. The first group contains the methods that explicitly detect changes and adapt the learner to cope with the new concept. These methods are trigger-based that means a signal indicates a need to modify the model. The second group contains the methods that adapt to current concept without explicit drift detection. These methods are called evolving methods that do not maintain a specific connection between the data progress and modifying the model. There are some works based on evolutionary algorithms [9], however most attention has been paid on non-evolutionary methods.

This paper aims to present an explicit drift detection method which is categorized in the first group. It is a robust-to-noise method which can both detect and adapt to concept drift in non-stationary data streams. Generally, explicit drift detection methods can be applied by using categories of classification methods: single classifier, and an ensemble of classifiers. According to the advantages of ensemble learning methods [10,11], we proposed a novel ensemble learning method. In order to detect concept drift, samples are processed one by one, and the error of the ensemble classification method is monitored to detect concept drift. In this method, the number of examples with the possibility of drift is counted, and the distances between these examples are measured. After detection of a drift, a classifier will be trained on the new concept and will be added to the existing classifiers. The experimental results illustrate the efficiency of using drift detection methods to find a concept drift location with much less delay and false detections, and more number of correct drift detections.

The rest of this paper is organized as follows: Section 2 contains related works in concept drift detection in data streams and Section 3 focuses on the proposed drift detection method. In Section 4 we examine our method with synthetic and real datasets and analyze it in the presence of noise. Section 5 evaluates the method by its parameters and Section 6 presents the conclusions and future works.

2. Related works

FLORA and STAGGER were the first systems, which were proposed for handling concept drift [13]. Then, more works were done on detecting changes in data streams [13–17]. Most of these methods are based on estimating the underlying distribution of examples, which require large amount of data, and are often dependent to the attributes of input data. Also, some methods are proposed to detect concept drift using small numbers of examples. These methods monitor the prediction errors of classification [1,18–23]. Whether the error rate decreases, these methods assume stationary distribution of data. A significant increase in the error rate will be considered as a change in the target concept. These methods are dependent on the type of input data and they do not estimate the data distribution. According to the number of classifiers used in these methods, drift detection methods can be divided into two categories [21,22]: 1) concept drift detection methods based on single classifier and 2) concept drift detection methods based on an ensemble of classifiers. The rest of this section describes the methods of these two categories.

2.1. Concept drift detection methods based on single classifier

These methods apply to single classifier learning methods. At any time, only one classifier is kept and after a concept drift is detected, an old classifier is replaced with a newly trained one. These methods include drift detection method (DDM) [1], early drift detection method (EDDM) [18], and detection with a statistical test of equal proportions (STEPD) method [19], to mention a few.

Gama et al. proposed a drift detection method (DDM) [1]. They assumed that if the target concept is stationary, then the error rate of the online classifier decreases as time advances. Therefore, a significant increase in the error rate of the online classifier means that the concept is changing. Two thresholds are specified in DDM, one for warning level and another one for drift level. DDM stores the examples in a short-term memory when the warning level is reached. Then, it rebuilds an online classifier from the stored examples and reinitializes all variables if the drift level is reached. DDM well detects sudden and significant changes, but it's not capable of detecting gradual and small changes. The main shortcomings of this method are short-term memory overflow, and partly misdetection of sudden changes.

Baena-García et al. proposed the early drift detection method (EDDM) [18] to improve the detection of gradual changes in DDM. The key idea is to consider distances (time intervals) between two occurrences of classification errors. EDDM assumes that a significant decrease in the distance suggests that a concept drift is occurring. This method calculates the average distance between two errors. If the warning level is reached, it stores examples in a short-term memory. When it reaches to a pre-specified the drift level, it rebuilds an online classifier from the stored examples and reinitializes all variables of the classifier. This method performs well for gradual changes, because the average distance between two errors is more sensitive to changes than the measure used in DDM (the number of errors). However, the high sensitivity might result in more false alarms in noisy environments.

In order to achieve a more accurate concept drift detection, Nishida and Yamauchi proposed the detection with the statistical test of equal proportions (STEPD) method that monitors the predictive accuracy of a single online classifier [19]. The main idea of this method is to consider two predictive accuracies for the online classifier: the recent accuracy, and the overall accuracy. This method assumes that there is a statistical significance after the occurrence of concept drift between the accuracy for recent examples and the overall accuracy from the beginning of the learning. STEPD compares these two accuracies using a statistical test of equal proportions. STEPD also uses two significance levels, warning level and drift level, similar to DDM and EDDM. Thus, STEPD stores examples in short-term memory while warning condition is satisfied, and then rebuilds a classifier from the stored examples and reinitializes all variables of classifier when drift level is reached. This method performs well for sudden changes. In comparison to EDDM, STEPD performs much better for detection of gradual changes, because EDDM is too sensitive to errors and noise, and has many false alarms. Moreover, STEPD is much faster than DDM in detection of sudden changes. Generally, STEPD performs well for both sudden and gradual changes.

DDM, EDDM, and STEPD store training examples in short-term memory while the warning level is touched and then rebuild an online classifier from the stored examples if the drift level is achieved. Although this memory slightly improves their predictive accuracy immediately after rebuilding the online classifier, their false alarms worsen their predictive accuracies significantly because these methods only use a single online classifier. In the next section we describe the methods that use ensemble of classifiers and their prediction results are more accurate than a single classifier.

Du et al. proposed ADDM [24], which is based on the information entropy of an adaptive sliding window. It keeps a single classifier, and update it incrementally which means after predicting the label of a newly arrived instance, the classifier will be updated with the true label of the instance. It considers correct prediction as 1, and false prediction as 0, and monitors the entropy through the sliding window to detect concept-drift. Detection and adaptation to concept-drift are completely separated in this methods as the classifier updates incrementally regardless of occurrence or non-occurrence of concept-drift.

AADD [25] is based on an anomaly analysis of learner's accuracy between learners' training domain and test data. It first identifies probable conflicts between current concept and newly arrived data. Then, the learner will incrementally update its model using the non-conflict data to extend learned concept. Non-conflict data is one which will not decrease the accuracy of the learner on previous trained data. Otherwise, a new learner will be created based on the new data.

Lu et al. presented a method to detect concept drift via competence models [26]. It requires no prior knowledge about the distribution of data, and measures it through a competence model. The competence-based change detection method can be applied to CBR (Case-Based Reasoning) systems, where new cases are available sequentially one by one.

2.2. *Concept drift detection methods based on ensemble of classifiers*

Concept drift detection methods that are discussed in the previous section use single classifier algorithms. There are other methods, which use an ensemble of classifiers in order to detect changes. The enhanced adaptive classifiers-ensemble (ACE) system [5] is one of these methods. The change detection mechanism of the enhanced ACE calculates the prediction accuracy of each classifier on some recent training examples. This method applies a single online classifier and some batch classifiers. Confidence intervals of the predictive accuracies of batch classifiers are used to detect concept drift. At any time, the best batch classifier is determined, and if the accuracy of it be out of its confidence interval, concept drift is suspected. Checking only the predictive accuracy of the best classifier is more effective for avoiding false alarms than checking the predictive accuracies of all batch classifiers. This method responds to sudden changes quickly and removes redundant classifiers well. Besides, it is able to control reoccurring concepts well. This method, like the methods in a single classifier category, stores samples in a short-term memory to build a new classifier from them when the drift is detected.

To reduce the bad influence of false alarms on predictive accuracy, the Todi system has been proposed [5]. It uses two online classifiers for learning and detecting concept drift and only one classifier is reinitialized when the concept drift is detected. The predictive accuracies of two classifiers are compared using the same statistical test of equal proportions used by STEPD. In comparison to STEPD, the Todi uses only one significant level that is the slightly modified STEPD method, and the use of short-term memory has been removed. So the trade-off between accurate detection and quick detection in STEPD method has been resolved by adding an online classifier, instead of using two significant levels and storing examples in short-term memory. In addition, the predictive accuracy of Todi system does not seriously deteriorate if false alarms occur.

A classification algorithm for concept drift detection based on an ensemble model of random decision trees (CDRDT) is suggested in [4]. The main goal of this algorithm is to detect concept drift in data streams with noise. This method behaves with prominent characteristics. First, it avoids the oversensitivity to the concept drifts and reducing the noise contamination. Second, it adjusts checking period dynamically for adaptation to concept drifts. Finally, it improves the abilities of space, time and predictive accuracy.

Du et al. proposed a method which has just one incremental classifier [27]. However, it keeps an ensemble, called e-Detector, of base detectors (which are responsible to detect changes in concept) for instance DDM. These base detectors uses the result of the classifier, and if any base detector finds a concept drift, e-Detector declares it.

3. The proposed method

In the related work section we discussed existing methods. As mentioned in detail, most of the exiting methods are focused on a particular type of concept-drift, and are unable or has weak accuracy in detecting other kind of drifts. However in the most of real world application, different kinds of concept-drift can occur in a single data stream. In this section, we introduce a new method for explicitly detecting concept drifts in data streams, which is called NDE (Number and Distance of Errors). The goal of this method is to detect all type of changes in the underlying distribution of data, and distinguish between drift in concept and outliers with minimum delay and a high rate of accuracy. Regardless of application, heightening the level of accuracy in this area points increase the number of true drift detection and decrease in the number of false drift detection. The proposed method is based on an ensemble of classifiers, and processes the output of classification method in order to detect changes. One advantage of the method is that it is not depend on the type of input data attributes. Moreover, it is able to detect concept drift without estimating the distribution of training examples.

During labeling a stationary concept, distance between the occurrences of errors in classification phase is long. Therefore, decrease in this distance and increase in the number of misclassification errors could be a sign of concept drift. Thus, a method to detect concept drift is perennial analyzing classifiers' misclassification rate. The high level idea of the proposed method is to keep a classifier per each observed concept. When a new concept arrives, we build a new classifier, and add it to the ensemble. In order to detect concept-drift occurrence we define a threshold for errors. Errors that are less than the threshold can be ignored as noise. In this method, incoming data are processed one by one, and the potential instances, which can be sign of drift in concept, will be counted. That is to say, if the number of misclassified instances were greater than the threshold, concept-drift is occurred. Also, distance between these instances in stream is calculated for higher accuracy in detection of concept drift. When a drift is detected, a new classifier will be trained and added to the ensemble.

In the NDE method, the Information about recent W instances are stored where W is the window size, and is a parameter of the algorithm. The possibility of drift is checked at any time by doing calculation on this window. With smaller window sizes, changes in the error value will have a great impact on the calculation while larger windows will be less sensitive to changes.

With the arrival of a new data point, the ensemble predicts a label per each newly arrived instance. Then, the error of ensemble's predictions on W recent points will be calculated. The error of classifiers equals to $err = r/W$, where r shows the number of misclassifications, and W is the size of the window. Since the err is a random variable with binomial distribution, by setting the window size with values larger than 30 ($|W| > 30$), this distribution can be approximated by a normal distribution. Hence, we can

```
Parameters: W: window size, E: number of errors, γ: confidence degree, D: distance
between errors
1.  Initialize wariningList = ∅, errList = ∅
2.  While more data point ( $x_t \in X$, $y_t \in Y$) is available do
3.      Calculate error.
4.      $\mu \leftarrow \sum_{i=t-w}^{t-1} err_i/W$
5.      $\sigma^2 \leftarrow \sum_{i=t-w}^{t-1} (err_i - \mu)^2/W$
6.      // Detect concept drift.
7.      // "err" shows error for recent example
8.      if $err > \mu + \gamma * \sigma$ And $|warningList| < E$ then
9.          If (t – last item in warningList) > D then
10.             reinitialize warningList
11.         end if
12.         add t to warningList
13.         if |warningList| ≥ E then
14.             Build new batch classifier from online classifier
15.             reinitialize online classifier, warningList, errList
16.         end if
17.     else
18.         add err to errList
19.         train online classifier with ($x_t$ , $y_t$)
20.     end if
21. end while
```

Fig. 1. Pseudocode of NDE.

use normal distribution to model the behavior of the errors. As a result, the proposed method calculates the mean and variance of stored errors using the following equations.

$$\mu = \sum_{i=t-W}^{t-1} err_i/W, \tag{1}$$

$$\sigma^2 = \sum_{i=t-W}^{t-1} (err_i - \mu)^2/W, \tag{2}$$

where err_i shows the error at time $t = i$.

In the NDE method, a parameter γ is dedicated to determine the confidence level of drift detection. In this method a threshold value, τ, is considered for detecting possibility of drift in the samples, which is calculated by equation $\tau = \mu + \gamma * \sigma$. The high value for the confidence level parameter, γ, indicates that more errors in the range are accepted. According to the value of confidence level specified by the designer, the probability that an error results in a detection of drift can be calculated.

By using μ, γ and σ the threshold value is calculated. At any time, if the calculated error of an instance exceeds this threshold value, it is counted as an instance with the possibility of drift. The usage of τ is because of the approximation of the errors by the normal distribution. The equation $err > \mu + \gamma * \sigma$ is used to compare the error of W recent examples (err) with this threshold. It pays to count the number of examples with the possibility of drift and measures the distance between them.

In the case of sudden drift, the number of errors higher than τ increases and the distance between them decreases. Also, noise in the dataset increases the number of errors, hence the best way to distinguish noise from the sudden drift, is to count the number of errors. The higher values for this number suggest that concept drift is occurring.

Figure 1 illustrates the pseudocode of the drift detection method. In this method, a list, which is called *warninglist*, is used to store examples with errors higher than the threshold value. The parameter E determines the maximum length of this list. There is a trade-off in determining a proper value for E. Large values for it leads to decrease in the number of detected changes. Also, it is possible that some of small changes cannot be detected. On the other side, small values for this parameter lead to increase in false alarms. When the *warninglist* is full, this means that a concept drift is occurred in data stream. Therefore, a new classifier will be built, and it will be added to the ensemble of classifiers.

Considering the number of errors from the beginning of the dataset is not meaningful. Hence, the distance between the occurrences of these errors is measured. If the distance between the most recent error and the previous error is greater than a threshold value, the previous samples are known as noise and will be ignored, and the error counter resets to zero. The parameter D determines the threshold value of distance between errors. The number of detected changes decreases by using small values for this parameter and false alarm increases by using greater values and a trade-off must be established.

It should be mentioned that determining a proper value for parameter D leads to higher accuracy in the detection of gradual changes. If the value of D is set to a small value, it cannot detect gradual changes while large values for it leads to increase in the number of false alarms. Hence, this is a trade-off and based on application, a proper value should be set for this parameter. If the distance between the errors were greater than the D, the algorithm ignores the warninglist buffer.

The NDE maintains an online classifier that is updated after observing each new instance and an ensemble of classifiers. When a concept drift is detected, the online classifier will be added to ensemble and will no longer be updated. A new online classifier will be trained with the instances that will come in future. The ensemble of classifiers is used to process the output of classification method for detecting changes.

The NDE does not update the previously built batch classifiers. In addition, to build the first batch classifier, NDE builds it when the number of batch classifiers is zero and the length of warninglist exceeds the number of errors, parameter E. Using both online and batch classifiers enabled NDE to handle recurring concepts well in the next presentation of each concept.

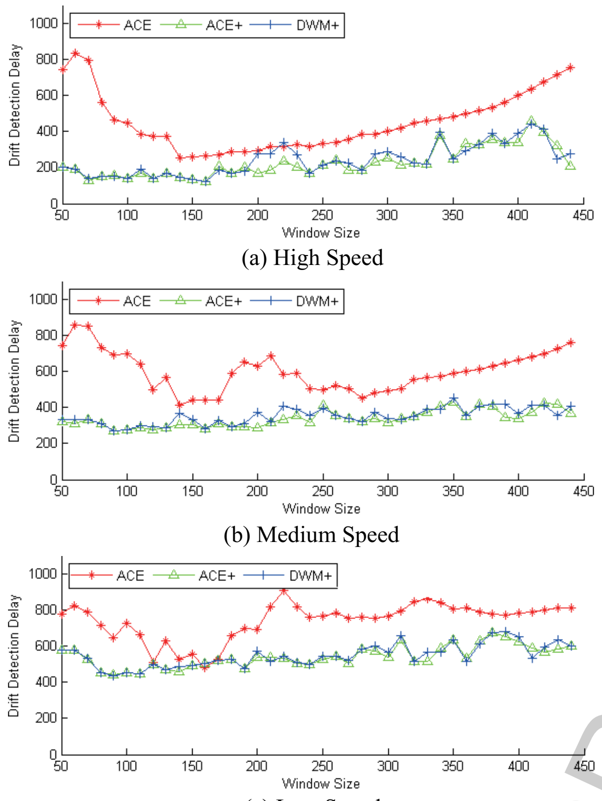
4. Experiments and results

In this section, we evaluate the performance of the proposed drift detection method on benchmark datasets. The NDE method should detect the occurrence of changes quickly. The criteria to evaluate concept drift detection methods are: 1) probability of true detection, 2) probability of false detection, and 3) delay in detecting concept drifts [28]. The DWM [30] and the Enhanced ACE method [5] are two of the well-known methods in this area, which we used to compare their results with our method. These methods process examples one by one. The DWM has an implicit drift detection method and the Enhanced ACE has an explicit method to detect concept drift.

The location of concept drift is not known in real datasets; hence we cannot utilize them to evaluate the method. Therefore, we employ the synthetic datasets and simulated datasets to experiment the detection methods.

We evaluated the NDE method in the presence of different types of drifts by using the artificial datasets described in [29], and the simulated dataset is given in [31]. Also two real datasets, KDDCUP-99 [32] and Usenet [31] are selected for comparison. Drifts have different severity and speed [29]. Severity describes the extent of the changes offered by a new concept and speed is the inverse of the time taken for a new concept to replace the old one. For more information, please refer to [29,33].

All methods are implemented in the MOA framework [34] and the classifiers are trained using a

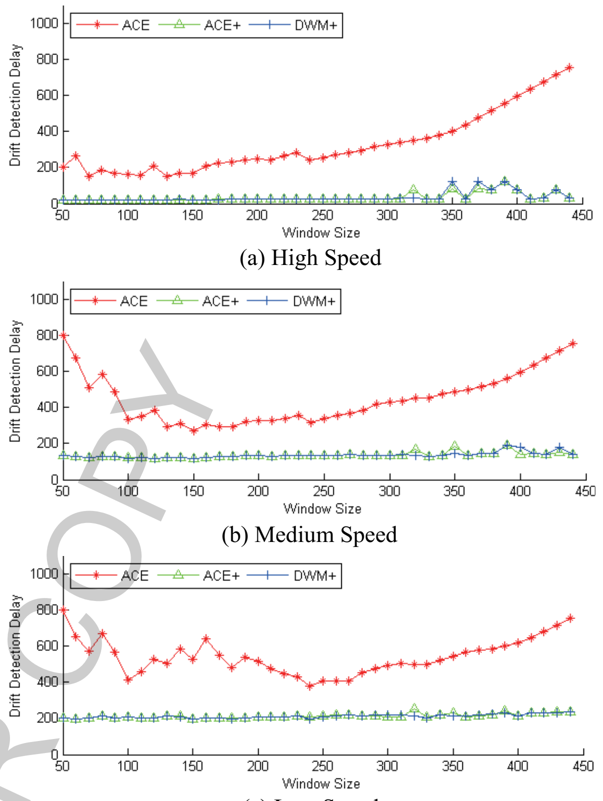


(a) High Speed

(b) Medium Speed

(c) Low Speed

Fig. 2. Results of drift detection delay on line problem with low severity and different levels of speed.



(a) High Speed

(b) Medium Speed

(c) Low Speed

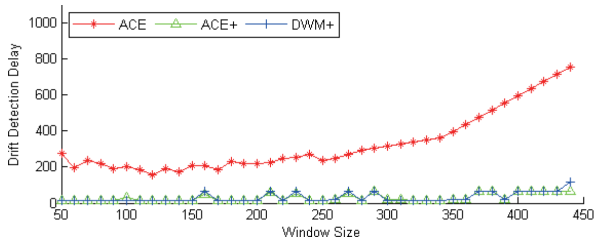
Fig. 3. Results of drift detection delay on line problem with medium severity and different levels of speed.

Naïve Bayes algorithm. For examining the drift detection method, we substitute the proposed detection method in the enhanced ACE and DWM and call them ACE+ and DWM+ respectively.

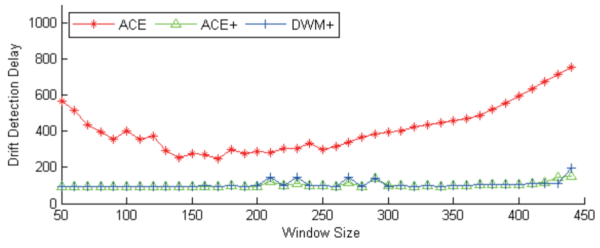
First, we evaluate the performance of the proposed concept drift detection method using artificial datasets. The most important feature of these datasets is the existence of concept drifts with different severities and speeds and also the clarity of the location of drifts. This artificial dataset includes circle, line, sine moving vertically (SineV) and sine moving horizontally (sineH) problems [29]. In each problem nine datasets are simulated by varying among three measures of severity and three speeds, each dataset contains one concept drift and 2000 instances, so that the first thousand examples are generated according to the first concept and then the drift is occuring. For the test on these datasets, 20 datasets of each type with similar characteristics, but with different data are produced and the results of experiments are averaged over 20 datasets of the same type.

The important criteria that are used for evaluating the performance of change detection methods include:

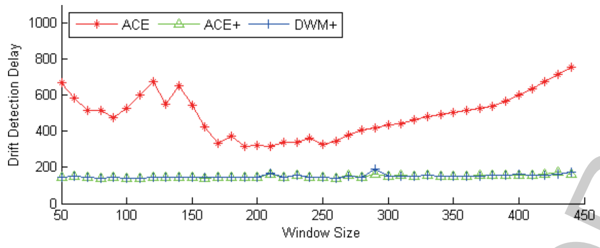
1. Probability of true change detection: This criterion indicates the percentage of correct changes that are detected in the dataset.
2. Probability of false alarm detection: This measure means that among all detected drifts, what percentage has been detected incorrectly (drift did not occurred).



(a) High Speed

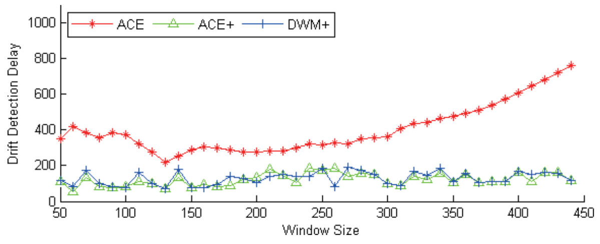


(b) Medium Speed

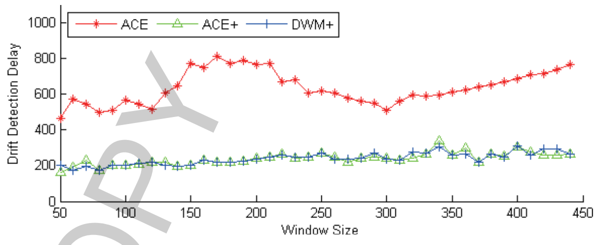


(c) Low Speed

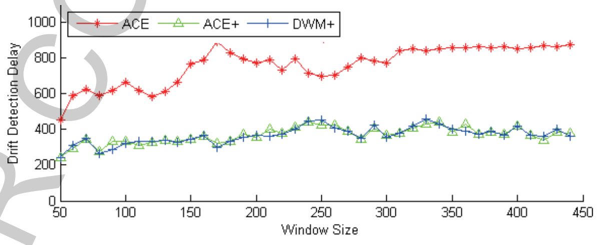
Fig. 4. Results of drift detection delay on line problem with high severity and different levels of speed.



(a) High Speed



(b) Medium Speed



(c) Low Speed

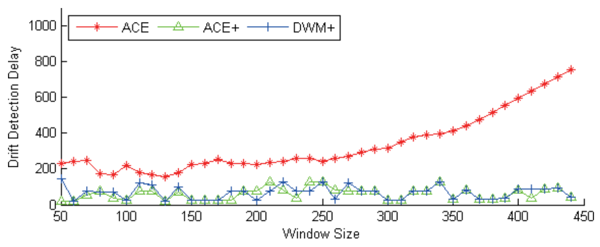
Fig. 5. Results of drift detection delay on circle problem with low severity and different levels of speed.

3. Delay in detection of concept drift: This criterion is used to compare the speed of change detection methods. Small delay in the detection of a concept drift means that the method detects changes more rapidly.

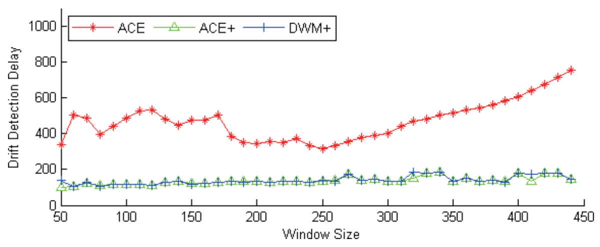
In Figs 2 through 13, the delay in detection of concept drifts for different datasets are presented. Figures 2 through 4 illustrate the delay of detecting different types of concept drift based on window size in line problem. It is clear that by using the NDE method, the delay decreases and the concept drifts were detected faster. The proposed method expresses that it can decrease the delay in detection of concept drift, and if a classification method uses it, it will detect changes faster than before as the comparison between ACE and ACE+ shows. This superiority is due to consider not only a threshold for counting the errors, but also both the number of errors and the distance between them.

Figures 5 through 7 show the drift detection delay in the circle problem. It is expected to have an increase in delay of detection by increasing the window size value. Also, it is obvious that concept drift was detected faster in high speed and severity drifts because the existence of concept drift is clearer and the drift occurred faster. So the NDE method is faster in detecting sudden drifts than gradual drifts.

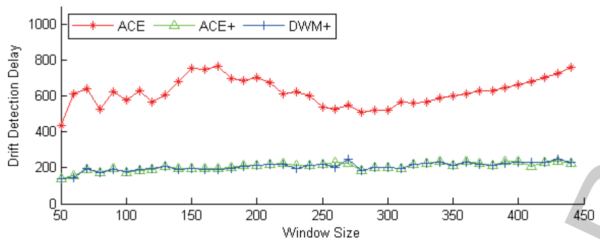
Figures 8 through 13 depict the drift detection delay in sineH and sineV problems. In these figures, it can be seen that the performance of the NDE drift detection method is better than the other methods. Moreover, it can be seen that increase in window size results in an increase in detection's delay. Besides,



(a) High Speed

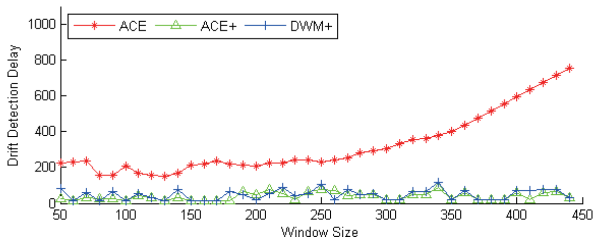


(b) Medium Speed

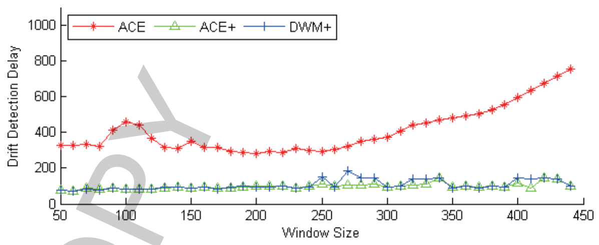


(c) Low Speed

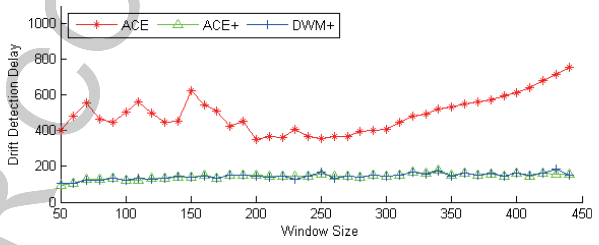
Fig. 6. Results of drift detection delay on circle problem with medium severity and different levels of speed.



(a) High Speed



(b) Medium Speed



(c) Low Speed

Fig. 7. Results of drift detection delay on circle problem with high severity and different levels of speed.

by increasing the speed and the severity of drift, the proposed method works better and detects changes faster.

We studied the effect of using the NDE method on delay of drift detection in the previous figures. Now, we evaluate the effect of the proposed method on the number of correct and incorrect detections. The percentage of true concept drift detections and false alarms are displayed in Table 1. The percentage of true drift detection means that what percentage of true concept drifts are detected. Therefore, 100% means that all of the true changes are detected. The percentage of false drift detection means that what percentage of changes that are false alarms. Hence, numbers greater than 100 percent means that we have false detections more than number of true changes and more of the change detections are. In this table, we see that the proposed drift detection method approximately detects all changes and we have no false alarm and it is better than ACE results. Also, as the speed and severity of changes increases, the proposed method works better.

4.1. Results on real datasets

In real datasets the occurrence of concept drift is not known. Hence, some of these datasets are simulated and location of drift and its type was determined. In the rest of this section we examine the proposed method on some of these datasets.

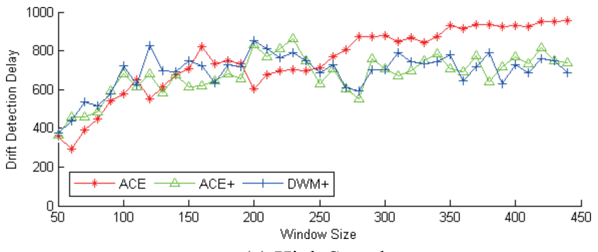
Table 1
Results of the percentage of true and false drift detection with different levels of severity and speed

Dataset	Properties	The percentage of true drift detection (%)			The percentage of false drift detection (%)		
		ACE	ACE+	DWM+	ACE	ACE+	DWM+
Circle	Low severity, high speed	100	100	100	125	50	30
Circle	Low severity, medium speed	35	100	100	50	40	10
Circle	Low severity, low speed	35	100	100	55	40	15
Circle	Medium severity, high speed	100	100	100	115	15	10
Circle	Medium severity, medium speed	80	100	100	100	80	15
Circle	Medium severity, low speed	100	100	100	75	95	35
Circle	High severity, high speed	100	100	100	120	165	30
Circle	High severity, medium speed	100	100	100	125	170	60
Circle	High severity, low speed	75	100	100	100	165	45
Line	Low severity, high speed	100	100	100	135	10	5
Line	Low severity, medium speed	85	100	95	100	20	5
Line	Low severity, low speed	65	95	95	95	10	5
Line	Medium severity, high speed	100	100	100	125	25	25
Line	Medium severity, medium speed	95	100	100	125	65	10
Line	Medium severity, low speed	60	100	100	75	95	15
Line	High severity, high speed	100	100	100	135	15	15
Line	High severity, medium speed	100	100	100	125	90	0
Line	High severity, low speed	70	100	100	105	130	5
SineV	Low severity, high speed	90	90	90	100	05	20
SineV	Low severity, medium speed	65	100	100	90	10	10
SineV	Low severity, low speed	50	90	90	75	15	5
SineV	Medium severity, high speed	100	100	100	140	20	20
SineV	Medium severity, medium speed	95	100	100	155	80	10
SineV	Medium severity, low speed	75	100	100	120	85	10
SineV	High severity, high speed	100	100	100	140	15	10
SineV	High severity, medium speed	100	100	100	150	90	0
SineV	High severity, low speed	95	100	100	145	140	10
SineH	Low severity, high speed	40	60	40	45	60	75
SineH	Low severity, medium speed	50	65	55	60	70	70
SineH	Low severity, low speed	45	55	65	60	60	65
SineH	Medium severity, high speed	50	70	50	50	60	65
SineH	Medium severity, medium speed	35	75	75	40	90	80
SineH	Medium severity, low speed	40	70	60	50	55	65
SineH	High severity, high speed	75	90	70	60	90	85
SineH	High severity, medium speed	40	85	90	40	95	85
SineH	High severity, low speed	50	90	75	55	65	75

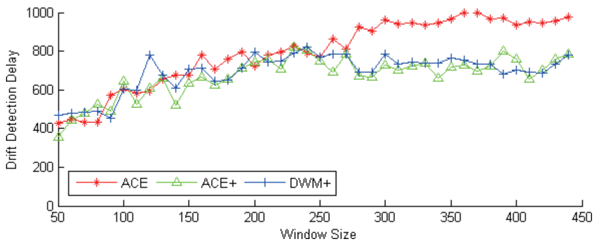
4.1.1. Results on usenet

The Usenet dataset is a simulation of news filtering with a concept drift related to the alternation of interest of a user over time. There are six topics chosen and the topics of interest are repeated to simulate recurring concepts. The set has 659 attributes and 5,931 examples. For more information, please refer to [31]. Figure 14 shows that the delay in our drift detection method has significant improvement over the other methods. This figure shows that DWM+ also has very good performance with small delay for detecting concept drifts. One of the reasons for the good performance of our method is that it only stores the W recent examples; hence it enhances the detection rate.

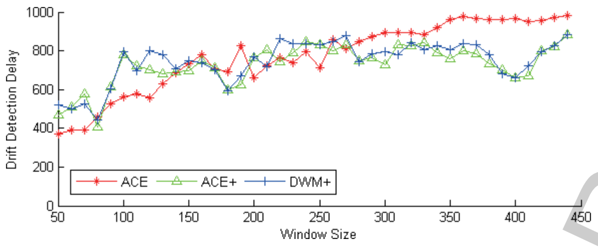
In Table 2, the percentage of true detection and false detection is presented. We note that by increasing the window size, there isn't a large variety in the number of detection the true concept drifts and the proposed detection method detects at least 90% of the true changes and the ACE method detects at least 60% of them. The number of false alarms tends to zero by increasing the window size in all methods. So we can conclude that our method performs better in detection the sudden drifts and reoccurring concepts.



(a) High Speed

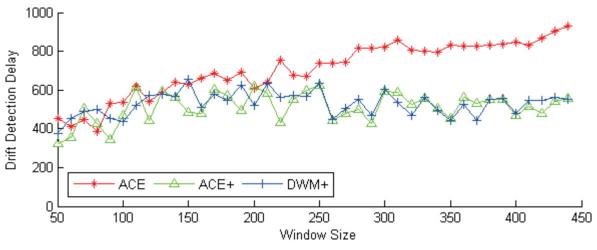


(b) Medium Speed

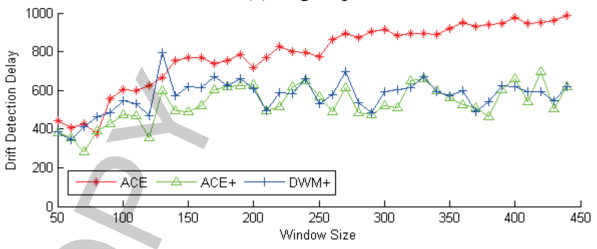


(c) Low Speed

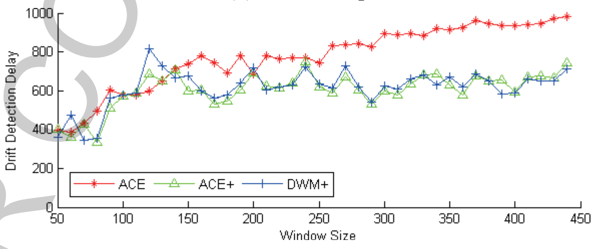
Fig. 8. Results of drift detection delay on sineH problem with low severity and different levels of speed.



(a) High Speed



(b) Medium Speed



(c) Low Speed

Fig. 9. Results of drift detection delay on sineH problem with medium severity and different levels of speed.

4.1.2. Results on KDDCUP-99

This real dataset was used in KDD Cup 1999 Competition [35] and is a set with only 10% of its size [32]. The location of drifts is not determined in this dataset, so only we can compare the accuracy of the methods. For this purpose, we calculate the accuracy in time steps with 1000 examples and the results is displayed in Figs 15 and 16. We have a small reduction in the accuracy of classification methods after the occurrence of concept drift by using our drift detection method. The reason lies behind considering both the number of errors and the distance between them. Besides, removing the need to have a short-term memory to store examples.

4.2. Performance of NDE in the presence of noise

It is important to consider noise in datasets when designing a method for detecting concept drifts in data streams. Noise makes it difficult to distinguish between the concept drift and noise, and some noisy examples may be identified as concept drift. In this section, we evaluate the performance of the proposed method in comparison with the other methods of change detection in the presence of noise. We perform some experiments on a modified SEA dataset. The dataset consists of sudden changes, gradual changes and reoccurring concepts, and is created by modifying the SEA dataset [5]. This dataset has 30 attributes, each with a value in the interval [0, 10], and only the first six attributes are related. We create a dataset with 10% noise and investigate the accuracy of the classification methods in the presence of

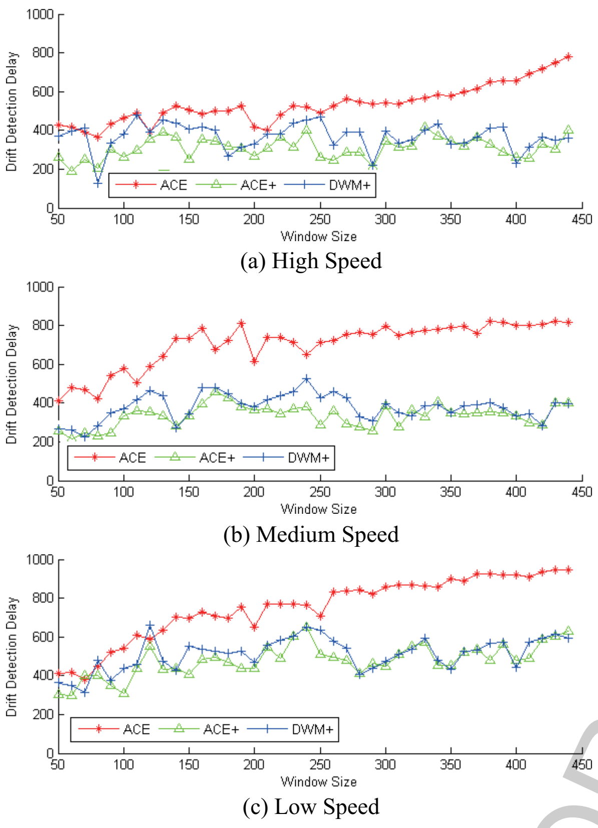


Fig. 10. Results of drift detection delay on sineH problem with high severity and different levels of speed.

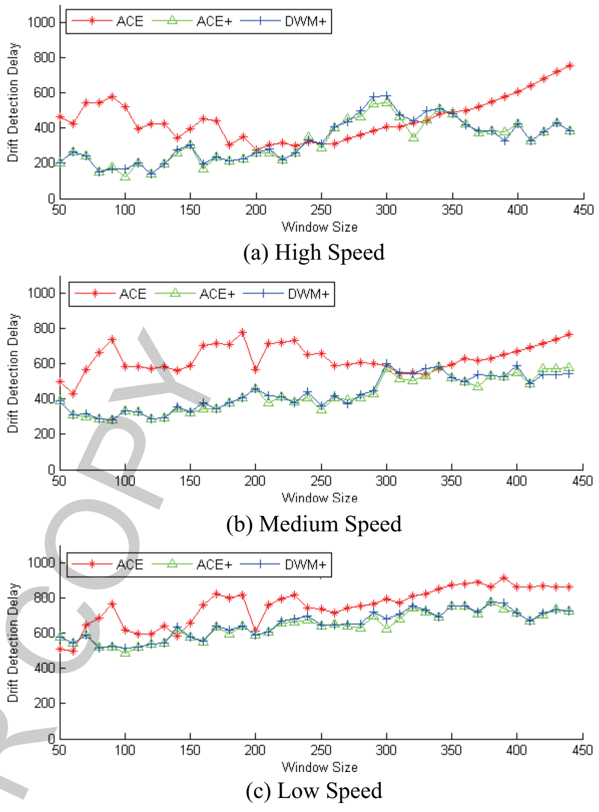


Fig. 11. Results of drift detection delay on sineV problem with low severity and different levels of speed.

noise. In Fig. 17 the accuracy of methods is measured by the value of window size equal to 200. In these experiments the accuracy of the classification method is computed at each time step, for example in time step 200 the accuracy is calculated on 200 previous examples and in time step 5500, it is calculated on 5500 examples. Figures 17(a) and (b) indicate that the accuracy of methods that use NDE method is almost similar to other methods and both of them have the same behavior in noisy environments.

In Fig. 18, the accuracy of classification methods in the intervals of 100 time steps on the 100 previous test data is noted. Diagrams indicate the NDE method in comparison with the DWM and DWM+ methods as well as ACE and ACE+ methods, we suffers a smaller accuracy drop, especially in the time steps in the interval of [11500, 13800], that there is a sudden drift, furthermore NDE adapts to new concept faster. It can be concluded that the proposed detection method works like other methods of detecting changes, but with little loss in accuracy when encountering the concept drift.

5. Analysis of NDE parameters

In this section, we examine the issue of changing the NDE parameters on its performance. Parameters that are considered in this section include: window size (W), number of errors (E), distance between the errors (D) and confidence degree (γ) parameters. We examine the outcome of these parameters on

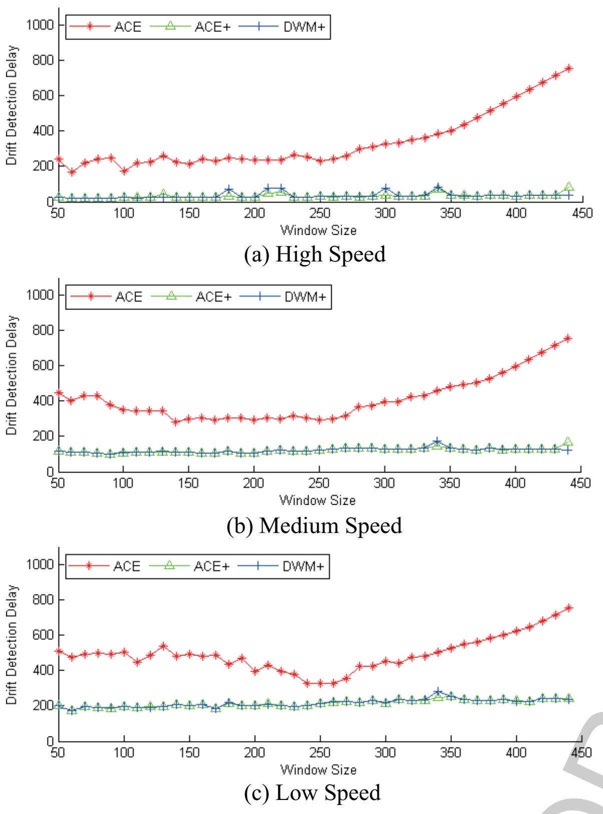


Fig. 12. Results of drift detection delay on sineV problem with medium severity and different levels of speed.

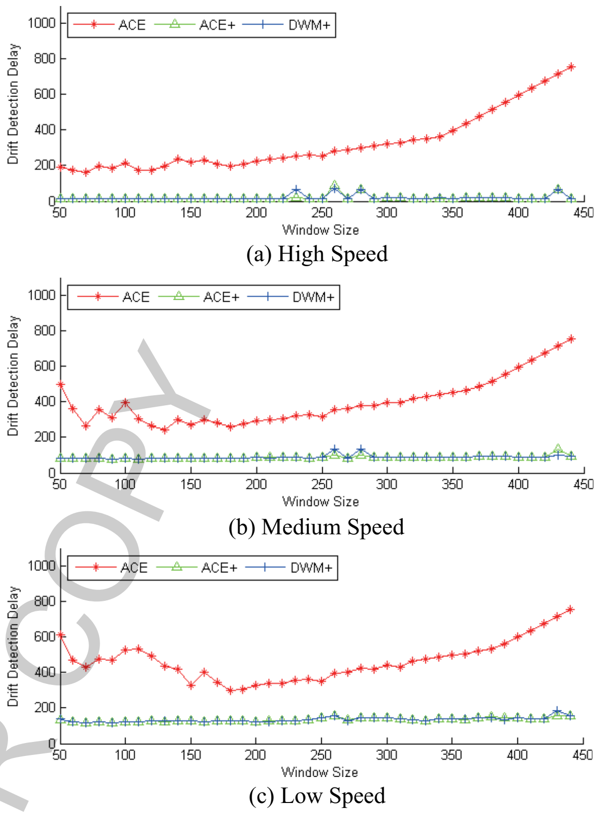


Fig. 13. Results of drift detection delay on sineV problem with high severity and different levels of speed.

the performance of NDE method. Measures that are evaluated here are delay in change detection, the number of true and false detections and finally the classification accuracy.

In order to test the NDE method in the presence of different types of drift, we use artificial datasets described in [7] and the previous section. In these datasets, drifts are categorized in 9 groups according to their severity and speed of changes. For each of line, circle, sineH and sineV problems, we assemble these 9 groups of concept drift in one dataset. Therefore we have a dataset with 9 concept drifts for each of these problems that drifts are occurring in the interval of 1000 time steps. We use these datasets since of including different types of concept drift according to their severity and speed. Then we study all of the NDE parameters in the presence of different types of concept drifts. The ACE+ classification method is used to evaluate NDE method.

5.1. Analysis of the window size parameter

Setting the window size in the presence of concept drift is a difficult challenge. Smaller windows may have not adequate data for precise decision about detection of concept drifts. In contrast, larger windows may contain data of different concepts and it makes the new concept drift detection slow. To obtain a suitable window size, we can experiment the method with different window sizes and select the best window size, or we can set the window size by intuition and according to the size of the dataset. In our experiments given in this paper, the W parameter is determined by trial and error.

Table 2
Percentage of true and false drift detections on usenet dataset

Window size	Percentage of true drift detection (%)			Percentage of false drift detection (%)		
	ACE	ACE+	DWM+	ACE	ACE+	DWM+
50	60	100	80	40	20	20
100	60	100	80	120	120	40
150	60	80	80	0	60	40
200	60	80	100	20	40	40
250	80	80	80	20	40	20
300	60	80	80	20	20	20
350	60	100	100	0	0	0
400	60	100	100	0	0	0

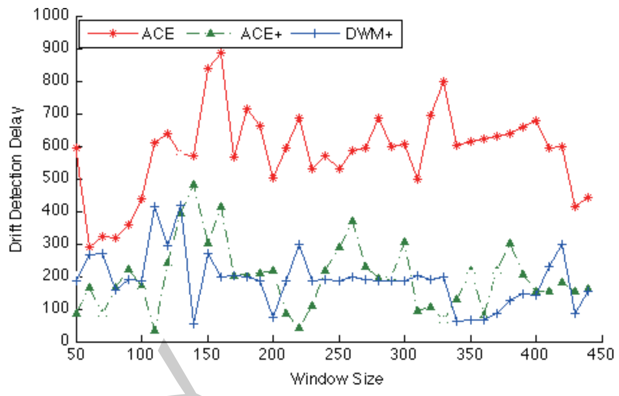


Fig. 14. Results of drift detection delay on usenet dataset.

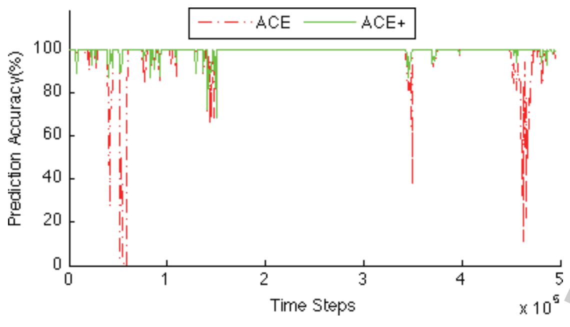


Fig. 15. The result of comparison between ACE and ACE+ on KDDCUP-99 dataset.

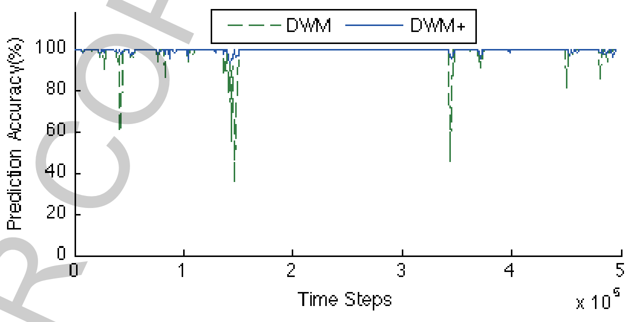
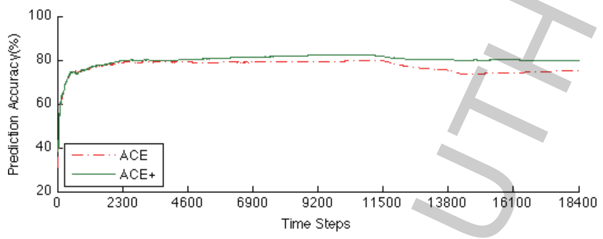
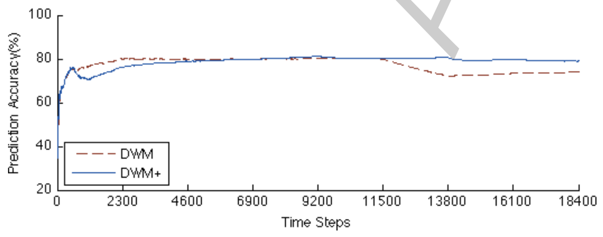


Fig. 16. The result of comparison between DWM and DWM+ on KDDCUP-99 dataset.

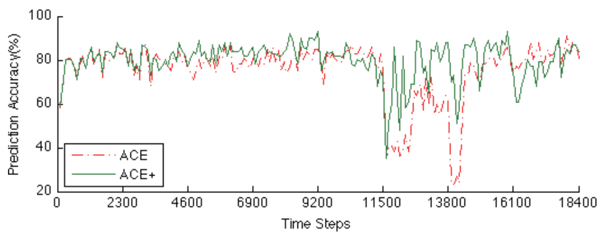


(a) The result of comparison between ACE and ACE+

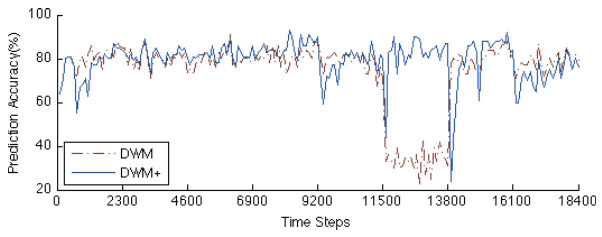


(b) The result of comparison between DWM and DWM+

Fig. 17. Comparison in the presence of noise.

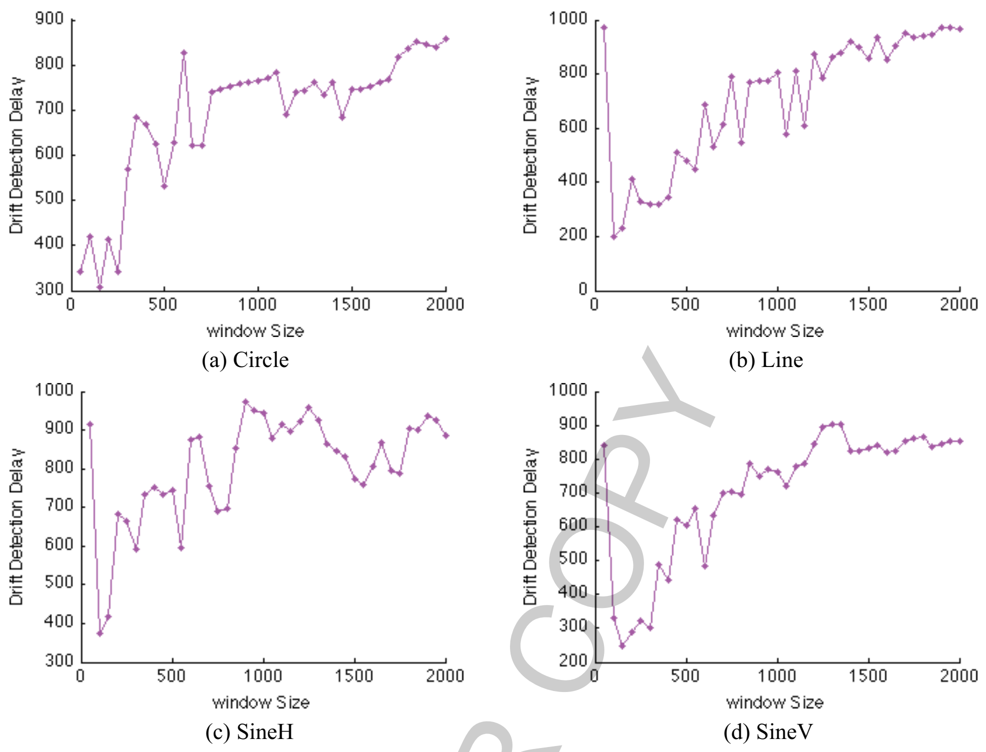


(a) The result of comparison between ACE and ACE+



(b) The result of comparison between DWM and DWM+

Fig. 18. Comparison in the presence of noise in time steps of 100.



(a) Circle (b) Line

(c) SineH (d) SineV

Fig. 19. Drift detection delay per window size.

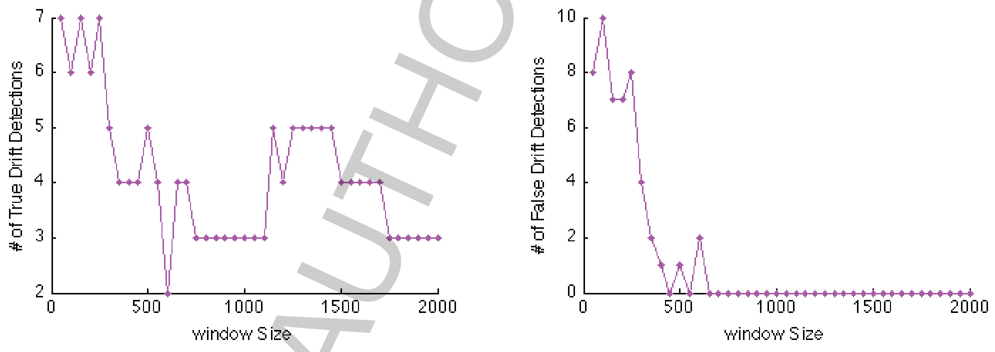


Fig. 20. Number of correct and incorrect detections per window size on circle problem.

In this section, the effect of different window sizes on line, circle, sineH and sineV problems are examined. The criteria that are examined to study the effect of this parameter include change detection delay, the number of true and false drift detections, besides the accuracy of the ACE+ classification method. The first value of W is set to 50 examples and in each trial the window size is increased by 50 examples.

Figure 19 shows the delay of concept drift detection per window size in line, circle, sineH and sineV problems. We take the average of the delay value in detection of 9 available concept drifts in each dataset to calculate the overall delay. Whether a concept drift is not detected, we record the maximum delay (1000) for it. There is a direct relationship between the window size and delay in change detection.

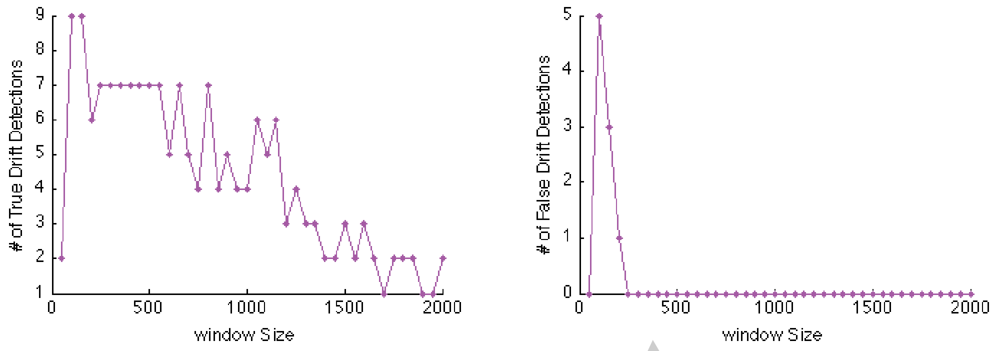


Fig. 21. Number of correct and incorrect detections per window size on line problem.

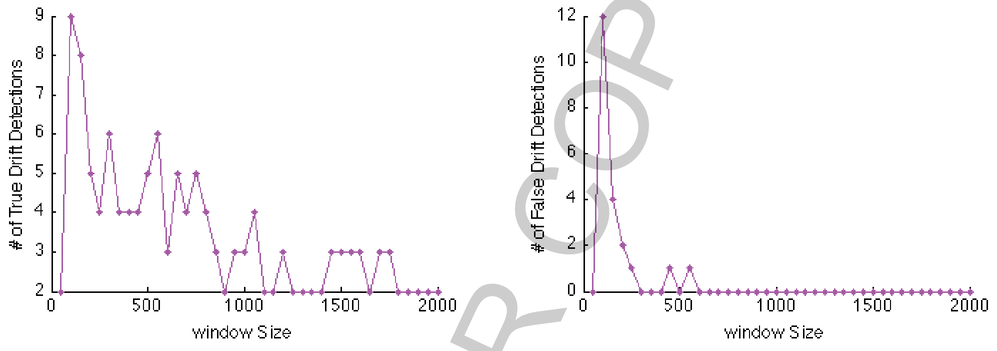


Fig. 22. Number of correct and incorrect detections per window size on sineH problem.

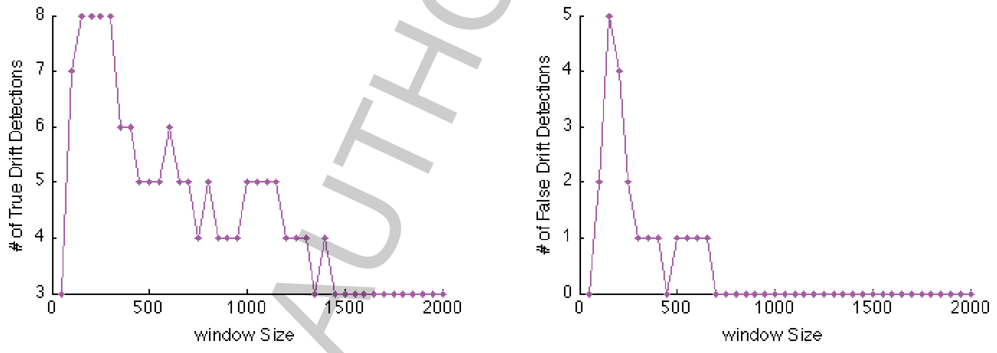


Fig. 23. Number of correct and incorrect detections per window size on sineV problem.

Whether the window size increases, the delay will go up too. In Figs 19(a) through (d) the minimum delay is achieved for windows with 100 or 150 examples. It is important to note that obtaining the best window size depends on the difficulty of the problem (dataset), location of concept drifts and the value of other parameters of the algorithm. The best method for determining the efficient window size is testing different values of this parameter and paying attention to characteristics of problem like the speed of the data stream.

In the rest of this section, the effect of changing the window size value on the number of correct and incorrect detections is investigated. Figures 20 through 23 show the experimental results on each

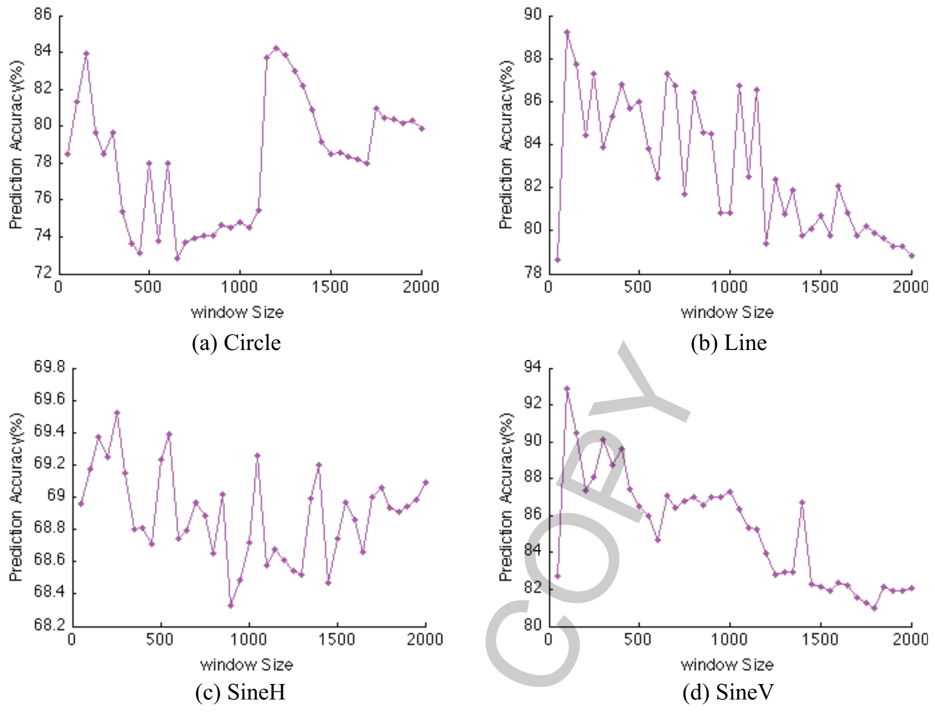


Fig. 24. The results of changing windows size on prediction accuracy of each problem.

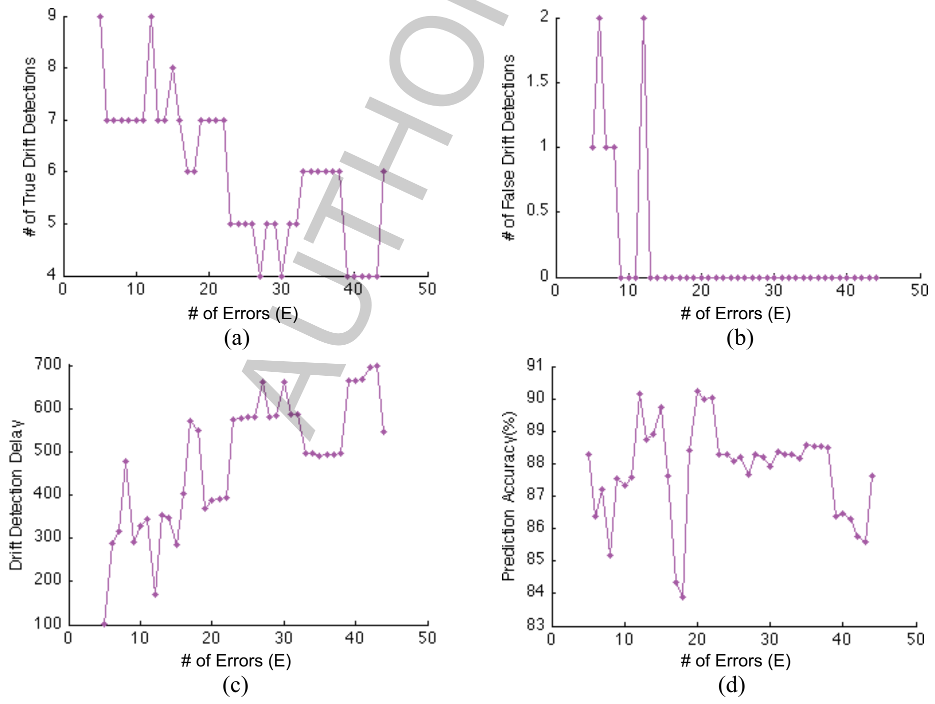


Fig. 25. The results of changing E parameter on line problem.

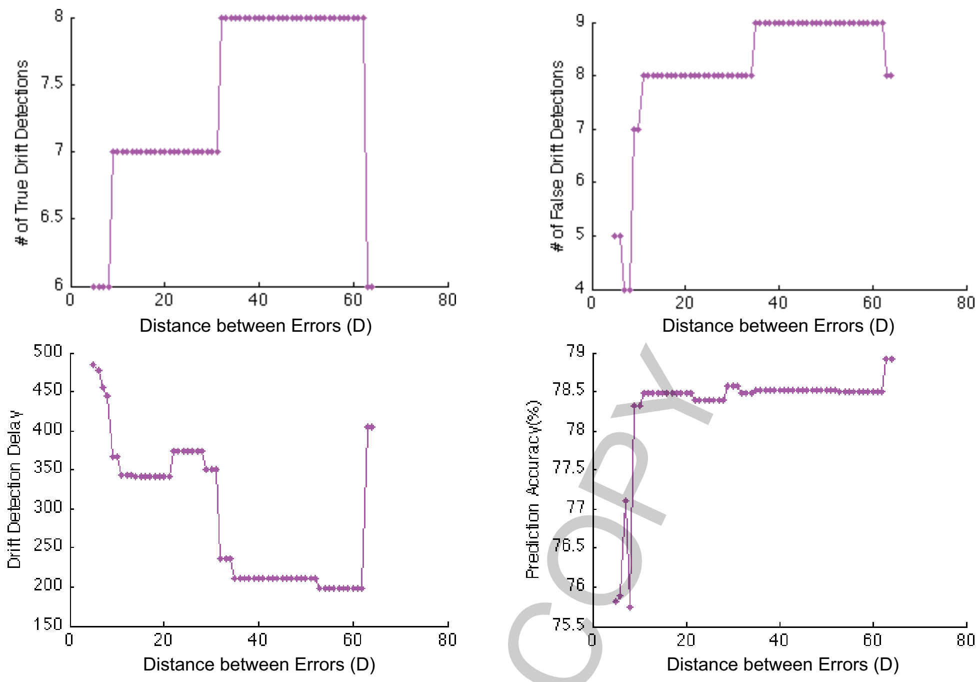


Fig. 26. The results of changing D parameter on circle problem.

problem. As we expected, by increasing the window size, the number of false alarms tends to zero. As it was said the detection of concept drifts becomes difficult by large window sizes and fewer changes are detected.

Finally, Fig. 24 shows the accuracy of the ACE+ method as window size increases. The accuracy of classification method is calculated along the entire dataset. As it is noticed the accuracy decreases by increasing the window size.

It is concluded that increasing the window size results to diminish in the accuracy, reduction in the number of false alarms and true detections and increasing the delay in change detection. This point should also be noticed that the dataset and types of its changes and the classification method effect on the value of this parameter.

5.2. Analysis of the number of errors parameter

As discussed, the NDE method sets a threshold for the error value, then calculates the error and counts the errors upper than the threshold value. This counter helps the method to detect concept drift. In this section, while the other parameters are fixed, parameter E changes and its outcome will be investigated. According to studies conducted until here, the behavior of the method is alike in all circle, line, sineV and sineH problems, so in the following we just study the effect of parameters on line problem. Figure 25 shows the results of increasing the value of the E parameter on drift detection criteria. It is observed as E increases, the speed of change detection decreases. In the other words, it detects changes later. Figure 25(d) illustrates that increasing E value causes to low accuracy in classification of data streams and it is due to a reduction in speed of change detection.

It is clear in Figs 25(a) and (b) that by increasing the number of errors, the number of either false or true detections decreases.

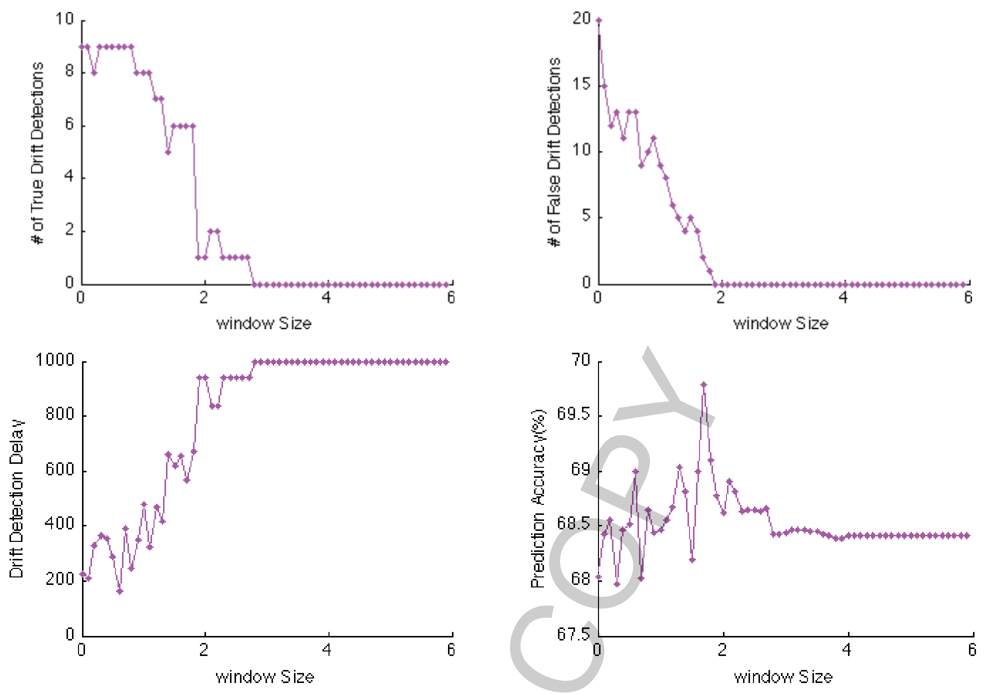


Fig. 27. The results of changing γ parameter on sineH problem.

5.3. *Analysis of the distance of errors parameter*

As reported earlier, considering the distance between the errors is an important parameter in the proposed change detection method. In this section, we examine the effect of changing D parameter in circle problem. In Fig. 26, the effect of this parameter on the circle dataset is shown. It is observed that increasing D parameter causes to 1) faster detection of changes, and 2) Increasing the accuracy of classification method in general, and 3) increasing the number of detections.

5.4. *Analysis of the confidence degree parameter*

By increasing confidence degree parameter, γ, from 0 to 2.4, confidence percent changes from 50% to 99.99%. Whether the γ value goes upper than 2.4, confidence percent tends to 100%.

It can be concluded that as the γ increases, the change detection delay increases and drifts are detected later than before. Figure 27 illustrates the results of changing γ in SineH problem. It is clear that for interval [0, 2.4], fluctuation in the criteria is observed and after 2.4 everything going to be fixed. Increasing this parameter from 0 to 2.4 results in a reduction in the number of true and false detections and it decreases the accuracy of ACE+ method. Since of reduction in the number of true detections, rate of change detection moves down and delay increases.

6. Conclusion

This paper introduced a new method, called NDE, for detecting concept drifts based on processing data one by one and measuring the error of classification methods. The idea behind the NDE method is considering a threshold value for errors, according to a normal distribution. Then we consider the number of errors and measure the distance between these errors based on this threshold. Errors are calculated for each example and are compared with determining a confidence interval for the most recent values for errors. The advantage of this approach, compared to the other methods, is that it can detect different types of drifts with different speed and severity. In all experiments on artificial and actual datasets, it is clear that by utilizing the proposed drift detection method, finding the location of drifts becomes faster and the number of true detections is improved, while the number of false detections is reduced. Also, there is no need for short-term memory for storing examples and this matter improves the detection rate. In future works, we intend to adjust the parameters of our drift detection method automatically and determine a relation between them. Also to reduce the bad influence of false alarms on predictive accuracy, we work on developing a novel system that uses two online classifiers.

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful comments that improved the quality of paper.

References

[1] J. Gama, P. Medas, G. Castillo and P. Rodrigues, Learning with Drift Detection, in: *Proc of the 17th Brazilian Symposium on Artificial Intelligence (SBIA'04)* (2004), 286–295.

[2] M.J. Hosseini, Z. Ahmadi and H. Beigy, Using a classifier pool in accuracy based tracking of recurring concepts in data stream classification, *Evolving Systems* **4**(1) (2013), 43–60.

[3] P. ZareMoodi, H. Beigy and S.k. Siahroudi, Novel class detection in data streams using local patterns and neighborhood graph, *Neurocomputing* **158** (2015), 234–245.

[4] P. Li, X. Hu, Q. Liang and Y. Gao, Concept drifting detection on noisy streaming data in random ensemble decision trees, *Machine Learning and Data Mining in Pattern Recognition* **5632** (2009), 236–250.

[5] K. Nishida, Learning and Detecting Concept Drift, Ph. D. dissertation, *School of Information Science and Technology*, Hokkaido University, Japan, 2008.

[6] I. Zliobaite, Learning under concept drift: An overview, technical report, *Dept of Mathematics and Informatics*, Vilnius University, Lithuania, 2009.

[7] K. Nishida and K. Yamauchi, Learning, Detecting, Understanding and Predicting Concept Changes, *International Joint Conference on Neural Networks* (Jun 2009), 2280–2287.

[8] I. Zliobaite, Adaptive Training Set Formation, *Ph D Thesis*, Vilnius University, Lithuania, 2010.

[9] Z. Karimi, H. Abolhassani and H. Beigy, A new method of mining data streams using harmony search, *Journal of Intelligent Information Systems* **39**(2) (2012), 491–511.

[10] H. Wang, W. Fan, P. Yu and J. Han, Mining Concept-Drifting Data Streams Using Ensemble Classifiers, in: *Proc of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 2003, 226–235.

[11] T. Dietterich, Ensemble Methods in Machine Learning, in: *Proc of the First International Workshop on Multiple Classifier Systems* (2000), 1–15.

[12] D. Brzeziński, Mining data streams with concept drift, *MS Thesis*, Dept. of Computing Science and Management, Poznan University of Technology, Poznan, Poland, 2010.

[13] C.M. Bishop, Novelty detection and neural network validation, in: *IEE Proc of Vision, Image and Signal Processing* **141**(4) (1994), 217–222.

[14] D. Kifer, S. Ben-David and J. Gehrke, Detecting change in data streams, in: *Proc of the 30th Very Large Data Base Conference* (2004), 180–191.

[15] M. Markou and S. Singh, Novelty detection: A review – part 1: Statistical approaches, *Signal Processing* **83**(12) (2003), 2481–2497.

[16] M. Markou and S. Singh, Novelty detection: A review – part 2: Neural network based approaches, *Signal Processing* **83**(12) (2003), 2499–2521.

[17] Z. Ouyang, M. Zhou, T. Wang and Q. Wu, Mining concept-drifting and noisy data streams using ensemble classifiers, in: *Proc of the International Conference on Artificial Intelligence and Computational Intelligence* **4** (2009), 360–364.

[18] M. Baena-Garcıa, J. Del Campo-Avila, R. Fidalgo and A. Bifet, Early drift detection method, in: *Proc of the 4th ECML PKDD International Workshop on Knowledge Discovery From Data Streams* (2006), 77–86.

[19] K. Nishida and K. Yamauchi, Detecting concept drift using statistical testing, in: *Proc of the 10th International Conference on Discovery Science* **4755** (2007), 264–269.

[20] K. Nishida, K. Yamauchi and T. Omori, ACE: Adaptive classifiers-ensemble system for concept-drifting environments, in: *Proc of the 6th International Workshop on Multiple Classifier Systems* **3541** (2005), 176–185.

[21] P. Sobhani and H. Beigy, New drift detection method for data streams, in: *Proc of 2nd International Conference on Adaptive and Intelligent Systems* **6943** (2011), 88–97.

[22] M. Dehghan, Concept drift detection in data streams using ensemble classifiers, *MS Thesis*, Dept. of Computer Engineering, Sharif university of Technology, Tehran, Iran, Jan, 2012.

[23] Z. Ahmadi and H. Beigy, Semi-supervised ensemble learning of data streams in the presence of concept drift, in: *Proc of 7th International Conference on Hybrid Artificial Intelligent Systems* **7209** (2012), 526–537.

[24] L. Du, Q. Song and X. Jia, Detecting concept drift: An information entropy based method using an adaptive sliding window, *Intelligent Data Analysis* **18**(3) (2014), 337–364.

[25] A. Liu, G. Zhang and J. Lu, Concept drift detection based on anomaly analysis, in: *Neural Information Processing*, Springer International Publishing (2014), 263–270.

[26] N. Lu, G. Zhang and J. Lu, Concept drift detection via competence models, *Artificial Intelligence* **209** (2014), 11–28.

[27] L. Du, Q. Song, L. Zhu and X. Zhu, A selective detector ensemble for concept drift detection, *The Computer Journal* **58** (2015), 457–471.

[28] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby and R. Gavaldà, New ensemble methods for evolving data streams, in: *Proc of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2009), 139–148.

[29] L.L. Minku, A.P. White and X. Yao, The impact of diversity on online ensemble learning in the presence of concept drift, *IEEE Transactions on Knowledge and Data Engineering* **22**(5) (May 2010), 730–742.

[30] J.Z. Kolter and M.A. Maloof, Dynamic weighted majority: AnEnsemble method for drifting concepts, *Journal of Machine Learning Research* **8** (2007), 2755–2790.

[31] Datasets for Concept Drift, (20 Nov 2010). http://wwwliaad.up.pt/∼kdus/kdus_5.html.

[32] C.C. Aggarwal, J. Han, J. Wang and P.S. Yu, A framework for on-demand classification of evolving data streams, *IEEE Trans on Knowledge and Data Engineering* **18**(5) (2006), 577–589.

[33] L.L. Minku and X. Yao, DDD: A new ensemble approach for dealing with concept drift, *IEEE Trans on Knowledge and Data Engineering* **24**(4) (2012), 619–633.

[34] A. Bifet, G. Holmes, R. Kirkby and B. Pfahringer, MOA: Massive online analysis, *Journal of Machine Learning Research*, **11** (2010), 1601–1604.

[35] A. Frank and A. Asuncion, UCI machine learning repository, (12 Sep 2010).