PAPER

An Efficient Concept Drift Detection Method for Streaming Data under Limited Labeling

Youngin KIM[†a)], *Nonmember and* Cheong Hee PARK[††b)], *Member*

SUMMARY In data stream analysis, detecting the concept drift accurately is important to maintain the classification performance. Most drift detection methods assume that the class labels become available immediately after a data sample arrives. However, it is unrealistic to attempt to acquire all of the labels when processing the data streams, as labeling costs are high and much time is needed. In this paper, we propose a concept drift detection method under the assumption that there is limited access or no access to class labels. The proposed method detects concept drift on unlabeled data streams based on the class label information which is predicted by a classifier or a virtual classifier. Experimental results on synthetic and real streaming data show that the proposed method is competent to detect the concept drift on unlabeled data stream.

key words: *concept drift detection, limited labeling, probability estimates, streaming data*

1. Introduction

A data stream is a sequence of data samples that is generated continuously over time and therefore it cannot be saved permanently to memory. It is difficult to process data streams using traditional data mining algorithms because the size of the data stream is typically too large to manage. Therefore, there is rising interest with processing data streams efficiently. One of difficulties in dealing with streaming data is that concept drift can happen since the data distribution or the interests of users can vary over time [1]. For example, in credit card payments when a new fraud pattern that has not been learned is found or the existing fraud pattern in deformed, the classifier may not be able to correctly predict a fraud event [2]. Concept drift can be classified into three major categories: the sudden drift which arises abruptly, the gradual drift which occurs steadily with an overlapping period, and the reoccurring drift indicating that the concept disappeared activates again after some interval of time. Regarding various scenarios of concept drift, numerous studies have been conducted so as to adjust the model to the latest concept [3], [4].

Generally, methods to cope with concept drift assume one among three situations according to the availability of data labels. The three assumptions are full access to all class labels of the data streams, no access, and limited access. When we can gain full access to class labels, for the incoming data samples, the labels are predicted through the current classification model. During the comparison process with actual labels, error rates can be used to determine whether to update the current classification model or generate a new model. However, the assumption that labels become available immediately after a data sample arrives is not generally suitable when processing the data stream due to the labeling costs and the time-consuming process to obtain labels. When access to the labels is limited and only small amount of labeled data is available, concept drift can be detected by monitoring the classifier output [5]–[7]. With no access to the labels of data samples, there is no information about the labels of the data stream. In such a case, the distribution of the data samples is typically used to detect concept changes [8], [9].

In this paper, we propose an effective drift detection method which can be applied under the limited or no access to class labels. The proposed method utilizes the probability estimates for class prediction and detects concept drift reliably on unlabeled data streams. This paper extends the conference paper [10] where the concept drift was dealt only for the situation of the limited labeling information. When the access to class labels is not available, the probability estimates for class prediction is generated by a virtual classifier based on the clustering structure.

The remainder of the article is organized as follows. Section 2 gives a brief review of the methods used to detect concept drift in streaming data. In Sect. 3, concept drift detection methods are presented in two types of situations of limited access and no access to class labels. Section 4 covers the experimental results for synthetic and real data. The conclusion is given in Sect. 5.

2. Related Work

Under the assumption that all class labels are available immediately, the drift detection method (DDM) [11] detects changes based on the number of prediction errors yielded by the classification model. Using the confidence interval estimation for the average of the error rates, DDM defines an abrupt increase in the error rate as an occurrence of concept drift. Early Drift Detection Method (EDDM) [12] detects changes by analyzing the distance between two errors. Exponentially Weighted Moving Average (EWMA) [13] method uses exponentially weighted moving average chart

Manuscript received March 15, 2017.
Manuscript revised May 31, 2017.
Manuscript publicized June 26, 2017.

[†]The author is with Information Technology Management Division, Agency for Defense Development, Korea.
[††]The author is with the Department of Computer Science and Engineering, Chungnam National University, Korea.
a) E-mail: yikim0911@add.re.kr
b) E-mail: cheonghee@cnu.ac.kr

DOI: 10.1587/transinf.2017EDP7091

to monitor the misclassification rate of the classifier.

In [14] learning from concept-drifting data streams is performed based on a sliding window of an adaptive size. The window size increases when no change is apparent and it shrinks when data changes. Using an ensemble of classifiers has been shown to be very powerful, especially when a base classifier is weak and unstable. In a concept-drifting data stream, a new member of the ensemble family is built on a chunk of recent data samples and an outdated member is removed. By assigning weights to ensemble members depending on the estimated error rate, concept drift can be dealt with [15]. In [16], two ensemble methods, bagging by ADWIN (ADaptive WINdowing) [14] and bagging by adaptive-size Hoeffding trees, are proposed.

Under the limited access to class labels due to high labeling cost and time consumption, the classification model is constructed using the small amount of labeled data and concept drift on unlabeled data is detected using the confidence value derived from the classifier. For the unlabeled data, the reference window and the detection window's confidence distributions are used to detect the change. The method in [5] analyzes a sequence of the posterior estimates derived from the classifier by using the univariate statistical tests such as Two sample t-test and Wilcoxon Rank sum test. CDBD (Confidence Distribution Batch Detection) [6] uses the confidence estimated by the classifier. It uses Kullback-Leibler Divergence to compare the distribution of the confidence values. Recently, uMD (using Margin Density) [7] method with the SVM classifier was proposed. The term Margin Density refers to the proportion of data samples that fall within the region of uncertainty by the SVM model. When the difference between the minimum and maximum margin density levels exceeds a predefined threshold, it is determined that a change has occurred. After drift happens, the labeling information of recent data samples is used to retrain the classifier. However, those methods in [5]–[7] can only be applied for binary class problems.

Semi-supervised learning methods utilize a large number of unlabeled data with the limited amount of labeled data. In [17], an incremental decision tree is built on data streams with unlabeled data. A clustering algorithm is used to detect concept drift at leaves and the detection of concept drift triggers splitting of the leaf node. In [18], a classification model is built on the data chunk of a fixed size by combining relational k-means clustering and semi-supervised learning, and it is used to predict class labels for the unlabeled data samples of the next data chunk. The method can be more useful for streaming data with gradual drift. An ensemble method of semi-supervised learning models was proposed in [19]. A classification model is constructed by k-nearest neighbor algorithm based on the clusters which are obtained by semi-supervised clustering on the data chunk of a fixed-sized window. The ensemble model is updated by choosing the best L models from the previous L models and the new model.

In the situation of no access to data labels, the drift detection methods capture the change using the distribution of the data features, as the labeling of the data is not available. General drift detection methods compare the data distribution in two windows and determine whether they come from the same distribution. Statistical tests such as The CNF Density Estimation Test [21] and Kolmogorov-Smirnov Test [8] are used to determine the differences in the distribution. When the null hypothesis that data in two windows are derived from the same distribution is rejected, the occurrence of concept drift is declared. The drift detection method in [9] uses Multivariate Wald-Wolfowitz test to detect the concept drift. However, it requires costly computations for the minimum spanning tree on a complete graph. Hido et al. [22] suggested the drift detection method by the virtual classifier(VC) which is constructed by assigning the label $+1$, -1 to adjacent two windows. This method assumes that if there is not certain difference in the two windows' distribution, the data samples in two windows will be classified by the almost 50% of prediction accuracy. The concept drift detection method in [21] also uses the VC. By using the SVM as a base model, it utilizes the average of margin and the error rate obtained by the VC. In [20], learning on a data stream of unlabeled data samples after initially labeled data samples is performed. Given the initially labeled data samples in a data chunk, unlabeled data samples within the same data chunk are classified with semi-supervised learning algorithm. Then, a boundary object for each class is compactly constructed and data samples called core supports are extracted which play a role of labeled data samples with predicted labels in the next step [20]. This process is repeated for the next unlabeled data chunk together with core supports. This method is intended for facing gradual drift rather than abrupt drift.

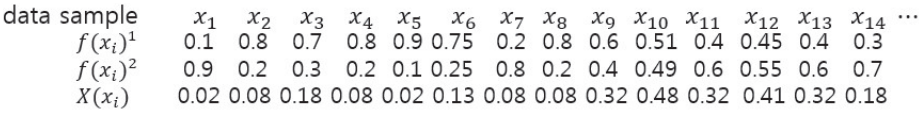
3. The Proposed Drift Detection Methods

3.1 Drift Detection under the Limited Access to Class Labels

The initial classification model is constructed through a certain amount of data samples with their labels. Assume that the data sample sequence $x_1, \cdots, x_{i-1}, x_i, \cdots$ comes after constructing the initial model and their labels are predicted by the model. Because the proposed method operates under the condition of limited access to labels, it is not available to use the true class label for data samples that arrives continuously. Hence, we define the random variable X by using the confidence vector (or posterior probability) $f(x_i)$ estimated by the classifier. For the data sample x_i, let us denote the confidence belonging to the class j as $f(x_i)^j$. The value $X(x_i)$ is defined through the distance between $f(x_i) = [f(x_i)^1, \cdots, f(x_i)^c]$ and the predicted label vector $\bar{y}_i$ such as

$$X(x_i) = \|f(x_i) - \bar{y}_i\|_F^2 \,. \qquad (1)$$

The class j with the highest value from the confidence vector $f(x_i)$ is predicted as the class label of x_i. In this case, the prediction label vector $\bar{y}_i$ refers to the vector on which



data sample	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	$\cdots$
$f(x_i)^1$	0.1	0.8	0.7	0.8	0.9	0.75	0.2	0.8	0.6	0.51	0.4	0.45	0.4	0.3	
$f(x_i)^2$	0.9	0.2	0.3	0.2	0.1	0.25	0.8	0.2	0.4	0.49	0.6	0.55	0.6	0.7	
$X(x_i)$	0.02	0.08	0.18	0.08	0.02	0.13	0.08	0.08	0.32	0.48	0.32	0.41	0.32	0.18	

Fig. 1 An example of the distribution of X

only the j-th component is 1 while the others are 0. Figure 1 illustrates $X(x_i)$ along with the predicted confidence values $f(x_i)^1$ and $f(x_i)^2$ in a two-class problem. From the data sample x_9, it can be observed that $X(x_i)$ is increased. The increment of $X(x_i)$ is due to the decrement of the largest confidence value in $f(x_i)^1$ and $f(x_i)^2$, which means the increase of the uncertainty in the prediction of a class label. Changes in the distribution of $X(x_i)$ implies that the model's behavior pattern has changed.

When $\bar{X}$ is the sample mean of n data samples from the distribution with a mean μ and standard deviation σ, the value $\bar{X}$ gives the point estimation for the mean μ. On the other hand, the confidence interval estimation for the mean μ is known to be $[\bar{X} - z_\alpha(\sigma/\sqrt{n}), \bar{X} + z_\alpha(\sigma/\sqrt{n})]$ when n is large ($n \geq 30$) and the samples are randomly drawn from a population [23]. When the standard deviation σ is unavailable, the standard deviation can be replaced with the sample standard deviation s. z_α is defined by the significance level. The proposed method detects a change in the distribution of X using two windows: the reference window W_{ref} and the detection window W_{det}. Data samples in the reference window come from the current concept. On the other hand, it is conjectured that data samples in the detection window might be generated from a new concept. When the variables m_{ref}, s_{ref} are set as the mean and standard deviation of $X(x_i)$ in the reference window W_{ref}, the upper bound of the confidence interval for the mean μ is $m_{ref} + z_\alpha \times s_{ref}/\sqrt{n}$. When the point estimation by the mean m_{det} in the detection window W_{det} satisfies the condition (2), the decision is made that the change has occurred.

$$m_{det} \geq m_{ref} + z_\alpha \times s_{ref}/\sqrt{n} \tag{2}$$

We set the upper bound as $z_\alpha = 3$ with the corresponding significance level of 99% in the experiments of Sect. 4. After the change is detected, the classification model is retrained using the data in W_{det} with the true class labels acquired from the experts. It should be noted that large-sample confidence interval estimation for a population mean requires that samples are drawn randomly from a population [23]. However, the condition of random sampling may not hold for confidence interval estimation in the reference window.

In setting the reference and detection window, we propose three approaches: using Fixed Reference Windows (FRW), Moving Reference Windows (MRW), and Ensemble of Reference Windows (ERW).

Fixed Reference Windows (FRW)

When using a Fixed Reference Window, changes are detected while keeping W_{ref} steady and only moving W_{det}. After training the initial model, in an incoming data sequence $x_1, x_2, \cdots$, the reference window W_{ref} is constructed by the values $X(x_i)$ from x_1 to x_n, and the detection window W_{det} is built by the data from x_{n+1} to x_{2n}. If the drift detection by Eq. (2) does not occur, W_{det} moves forward. That is, the oldest element $X(x_{n+1})$ in W_{det} is deleted, and from a newly arriving data sample $x_{2n+1}, X(x_{2n+1})$ is calculated and added to W_{det}. Consequently, W_{det} now becomes $\{X(x_{n+2}), \cdots, X(x_{2n+1})\}$. Figure 2 (a) depicts the method FRW.

Moving Reference Windows (MRW)

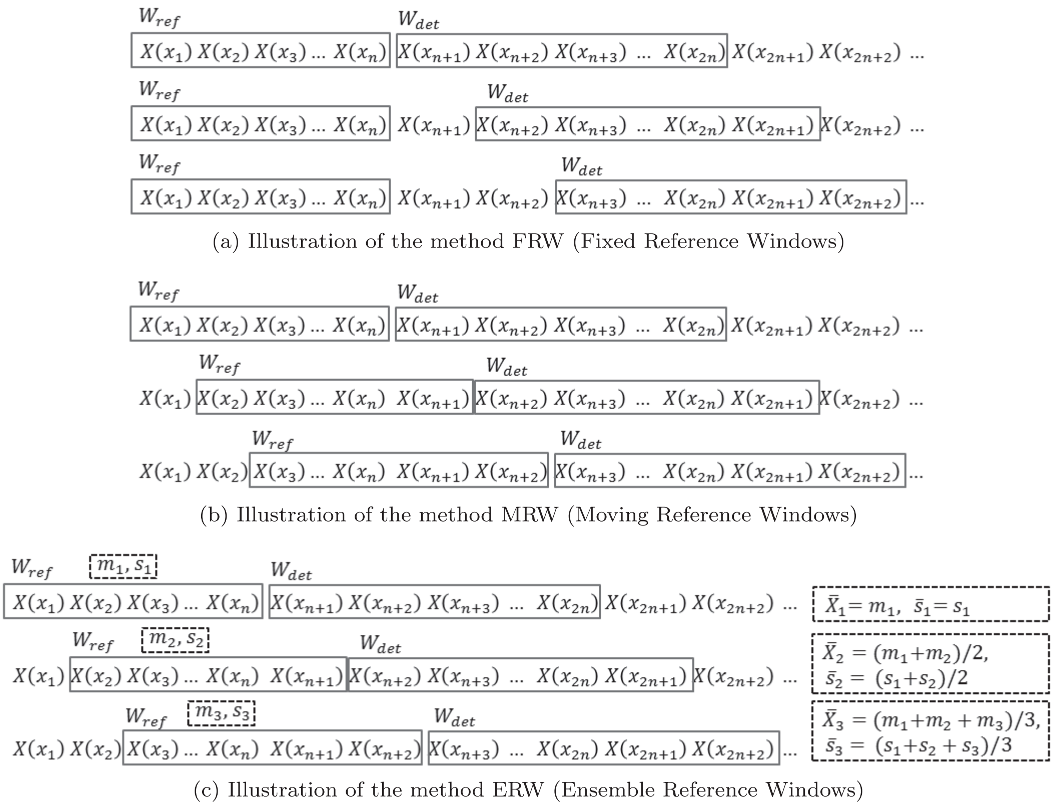
When using a Moving Reference Window, concept drift is detected by moving both windows, $W_{ref} = \{X(x_1), \cdots, X(x_n)\}$ with $W_{det} = \{X(x_{n+1}), \cdots, X(x_{2n})\}$. If no changes is signaled, W_{ref} and W_{det} move step by step and the revised windows are made such as $W_{ref} = \{X(x_i), \cdots, X(x_{n+i-1})\}$ and $W_{det} = \{X(x_{n+i}), \cdots, X(x_{2n+i-1})\}$. The MRW method is shown in Fig. 2 (b).

Ensemble of Reference Windows (ERW)

In the method ERW, drift is detected by using $\bar{X}$ *and* s *chart* [24]. Control chart is generally used in industries to control the statistical quality of data. $\bar{X}$ and s chart, which is one of various control charts, observes changes in distribution using the mean $\bar{X}_w$ of windows' means $\{m_1, m_2, \cdots, m_w\}$ and the mean $\bar{s}_w$ of windows' standard deviations $\{s_1, s_2, \cdots, s_w\}$. When the condition (3) is satisfied, it is determined that certain problem has occurred [24].

$$\bar{X}_w + 3 * \bar{s}_w/(\sqrt{n} * c_4) \leq m_{w+1} \tag{3}$$

The constant $c_4 = \sqrt{2/(n-1)} * (n/2-1)!/((n-1)/2-1)!$ varies with the size n of the window. It is used to minimize the difference in the confidence interval by the size of the window. The goal in control charts is to prevent the occurrence of possible problems beforehand. While in the methods FRW and MRW a single reference window is used, in ERW the mean and standard deviation from previous reference windows are kept until concept drift is detected. That is, the mean $\{m_1, \cdots, m_w\}$ and the standard deviation $\{s_1, \cdots, s_w\}$ of the previous reference windows $W_{ref_1}, \cdots, W_{ref_w}$ are continuously generated until drift occurs. When the mean m_{det} of W_{det} exceeds the upper bound of the confidence interval estimation in (3), it is considered that concept shift occurs. Figure 2 (c) depicts the ERW method.

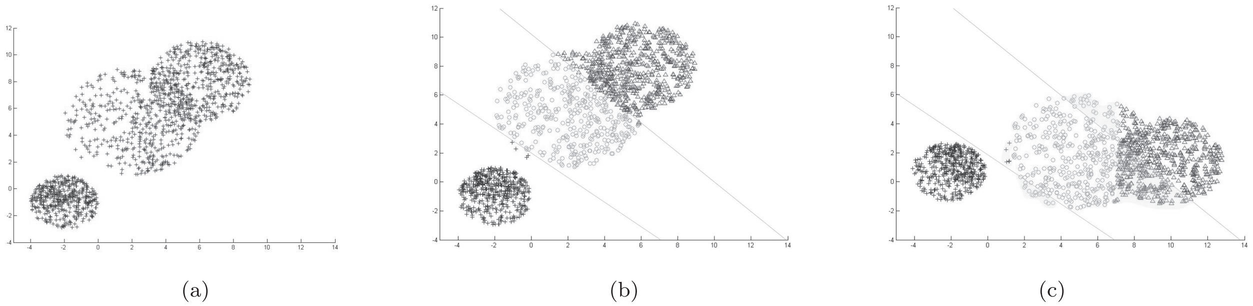

W_{ref}
W_{det}
$X(x_1)\ X(x_2)\ X(x_3) \dots X(x_n)$
$X(x_{n+1})\ X(x_{n+2})\ X(x_{n+3})\ \dots\ X(x_{2n})$
$X(x_{2n+1})\ X(x_{2n+2})\ \dots$
m_1, s_1
m_2, s_2
m_3, s_3
$\bar{X}_1 = m_1,\ \bar{s}_1 = s_1$
$\bar{X}_2 = (m_1+m_2)/2,\ \bar{s}_2 = (s_1+s_2)/2$
$\bar{X}_3 = (m_1+m_2+m_3)/3,\ \bar{s}_3 = (s_1+s_2+s_3)/3$

(a) Illustration of the method FRW (Fixed Reference Windows)

(b) Illustration of the method MRW (Moving Reference Windows)

(c) Illustration of the method ERW (Ensemble Reference Windows)

Fig. 2 Illustrations of FRW, MRW, and ERW



(a)

(b)

(c)

Fig. 3 (a) A chunk of unlabeled data samples (b) clustering structure and a virtual classifier (c) data samples after concept drift

3.2 Drift Detection under No Access to Class Labels

The detection of drift at appropriate time can help effective analysis about the change in data generation mechanism. We intend to apply the drift detection method proposed in Sect. 3.1 for drift detection under no access to class labels. However, since no class labels are available at all, it is impossible to model the initial classifier. In order to circumvent the problem, we resort to the construction of grouping structure by clustering. By applying a clustering method such as k-means clustering for the unlabeled data chunk, a clustering structure is obtained. Now a virtual classifier is built based on the class label information which is formed by clustering. For incoming data samples, the class labels are predicted by the virtual classifier and drift detection based on posterior probabilities by the virtual classifier is performed as in Sect. 3.1. Figure 3 (b) shows the result of k-means clustering with $k = 3$ for unlabeled data chunk of Fig. 3 (a). A virtual classifier described by two hyperplanes is also shown in Fig. 3 (b). Figure 3 (c) depicts data samples after the change in data distribution by the 45 degree's rotation.

Table 1 summarizes the algorithm for drift detection under the limited access to class labels which is presented in Sect. 3.1. A little modification is needed for the application in the situation of no access to class labels. The initial classifier f is replaced by the virtual classifier constructed on the grouping structure by a clustering method. Also the lines 11∼12, 18∼19, and 28∼30 need to be substituted by

Table 1 The algorithm for drift detection under the limited access to class labels

```
Input : $f$ : current classifier; $n$ : window size;
        $type$ : three approaches of the proposed method;
        $m_{ref}$,$s_{ref}$ : the mean and the standard deviation of $W_{ref}$;
        $m_{det}$ : the mean of $W_{det}$; $w = 0$;

while (an incoming data sample $x$ is available)
  if length($W_{ref}$)$< n$
    Compute $X(x)$ and attach it to the end of $W_{ref}$;
  elseif length($W_{det}$)$< n$
    Compute $X(x)$ and attach it to the end of $W_{det}$;
  else
    Compute $X(x)$ and attach it to the end of $W_{det}$;
    if $type$ == FRW
      Delete the oldest instance in the $W_{det}$;
      if $m_{det} \geq m_{ref} + 3 * s_{ref} / \sqrt{n}$
        Learn a new classifier $f$ using data instances of $W_{det}$;
        Initialize $W_{ref}$, $W_{det}$ as empty;
      end
    elseif $type$ == MRW
      Move the oldest instance of $W_{det}$ to the end of $W_{ref}$;
      Delete the oldest instance of $W_{ref}$;
      if $m_{det} \geq m_{ref} + 3 * s_{ref} / \sqrt{n}$
        Learn a new classifier $f$ using the instances of $W_{det}$;
        Initialize $W_{ref}$, $W_{det}$ as empty;
      end
    elseif $type$ == ERW
      Move the oldest instance of $W_{det}$ to the end of $W_{ref}$;
      Delete the oldest instance of $W_{ref}$;
      $w = w + 1$; // increase the number of windows by one
      Calculate $m_w$ and $s_w$ from $W_{ref}$;
      Update the $\bar{X}_w$, $s_w$;
      if $\bar{X}_w + 3 * s_w / \sqrt{n} * c_4 \leq m_{det}$
        Learn a new classifier $f$ using the instances of $W_{det}$;
        Initialize the ensemble of $W_{ref}$ and $W_{det}$ as empty;
        $\bar{X}_w = 0$, $s_w = 0$, $w = 0$;
      end
    end
  end
end
```

“Declare that the concept drift has occurred; Break;”.

4. Experimental Results

4.1 Data Sets

To evaluate the performance capabilities of the proposed methods, we conducted experiments on synthetic data from the UCI machine learning repository [25]. A detailed description of data sets is given in Table 2. We manipulated the data artificially to create drift after the half of the data stream. We selected x classes and y features randomly, and for data samples in the selected classes the selected features were shuffled in the second half of the data stream. The change magnitude was set as $x = \#class/2$ and y = 50% respectively. In addition, the real streaming data that is known to have concept drift was used. Three real world datasets, Electricity, Forest Covertype, and Usenet were tested. The Forest Covertype dataset contains originally 581,012 data samples of seven classes. However, the two most frequently occurring classes occupy 85.22% of the data. Hence, we used the data samples from the two most frequently occurring classes for a comparison with the methods which only deal with two-class problems.

Table 2 The detailed description for the data sets

Data set	#Instances	#Features	#Class
Pendigits	10992	16	10
Satellite	6435	36	6
Nursery	12960	8	5
Waveform	5000	21	3
Magic	19020	10	2
Mushroom	8124	22	2
Kr-vs-kp	3196	36	2
Electricity	45312	8	2
Covertype	495141	54	2
Usenet	5931	658	2

4.2 Evaluation Measures

Under the limited access to class labels, after the change is detected, the classification model is retrained using the data with the true class labels acquired from the experts. As an evaluation measure for the drift detection methods, classification accuracy for the unlabeled data samples in data streams can be used, since accurate detection for concept drift makes the performance of a classifier be maintained.

Table 3 Experimental results using LDA as a base model

						The proposed methods		
		DDM	EDDM	EWMA	F-COMP	FRW	MRW	ERW
Pendigits	F1-measure	0.6857	0.2884	1.80E-04	-	0.7253	0.2949	**0.7363**
	accuracy	**0.86**	0.8433	0.8352	0.5284	0.8344	0.8327	0.8282
	usage (%)	100	100	100	0	3.9073	12.0667	3.6583
Satellite	F1-measure	0.1683	0.0207	0.0028	-	0.7093	0.4054	**0.7523**
	accuracy	**0.792**	0.7506	0.7219	0.64	0.7656	0.7633	0.7652
	usage (%)	100	100	100	0	5.2675	12.6615	4.9076
Nursery	F1-measure	0.2567	0.2765	0.03	-	0.2387	0.2629	**0.2863**
	accuracy	**0.8638**	0.8575	0.8178	0.385	0.8113	0.8244	0.8147
	usage (%)	100	100	100	0	1.9818	11.3873	1.6569
Waveform	F1-measure	0.1133	0.0917	0	-	0.4117	**0.4265**	0.4167
	accuracy	**0.8701**	0.8589	0.8671	0.5738	0.7436	0.7822	0.7552
	usage (%)	100	100	100	0	4	12.5474	3.7474
Magic	F1-measure	**0.1973**	0.0999	0	-	0.0662	0.1449	0.0804
	accuracy	0.8221	0.7722	0.813	0.685	0.7074	**0.8442**	0.7074
	usage (%)	100	100	100	0	1.4057	14.7546	1.45
Mushroom	F1-measure	**0.3137**	0.2462	0	-	0.2609	0.2518	0.248
	accuracy	**0.9557**	0.9089	0.9424	0.5841	0.8902	0.9424	0.8893
	usage (%)	100	100	100	0	3.7061	19.2303	4.0689
Kr-vs-kp	F1-measure	0.0567	0.155	0	-	0.13	**0.506**	0.16
	accuracy	0.9431	**0.9444**	0.9424	0.5446	0.8077	0.8654	0.8142
	usage (%)	100	100	100	0	2.5033	13.2411	3.0303
Electricity	Accuracy	**0.7339**	0.731	0.7198	0.5966	0.6158	0.6455	0.6537
	#drifts	241	352	591	-	17	77	63
	usage (%)	100	100	100	0	7.8985	35.7757	29.271
Covertype	Accuracy	0.9293	0.9067	**0.935**	0.4951	0.5852	0.7891	0.7657
	#drifts	6690	1409	6227	-	43	953	626
	usage (%)	100	100	100	0	1.8283	40.5202	26.6166
Usenet	Accuracy	0.5752	0.5897	**0.6175**	0.5295	0.5308	0.5567	0.5692
	#drifts	2	24	12	-	5	6	5
	usage (%)	100	100	100	0	17.7494	21.2993	17.7494

In addition to prediction accuracy, we used Precision, Recall, and F1-measure in order to evaluate the performance of detecting the concept drift points correctly in artificially drifting data sets. All instances of drift that appear before the actual drift are termed FP (False Positive) cases, and the first change detection point after the actual drift is defined as a TP (True Positive) case. When there are no changes detected after the actual drift, we set the case as a FN (False Negative). With these three indexes, the measures can be calculated as shown below.

$$Precision(PR) = TP/(TP + FP),$$
$$Recall(RC) = TP/(TP + FN),$$
$$F1 - measure = (2 * PR * RC)/(PR + RC).$$

For the performance comparison of the drift detection methods in the situation with no access to class labels, F1-measure was only used.

4.3 Experimental Results under the Limited Access to Class Labels

In order to evaluate the performance of the drift detection methods, we conducted the experiments on synthetic data and real streaming data sets. The proposed drift detection methods FRW, MRW, and ERW were compared with various methods belonging to three categories as follows.

(1) *DDM* [11], *EDDM* [12], *EWMA* [13]: Drift detection methods are used along with a incrementally updatable classifier which require all data samples to be labeled.

(2) *uMD (using Margin Density)* [7], *CDBD (Confidence Distribution Batch Detection)* [6]: Using SVM as a classifier, it performs drift detection on unlabeled data streams. When drift is detected, a new classifier is built with labeled data within a drift warning interval.

(3) *F-COMP. (FAST-COMPOSE: FAST COMPacted Object Sample Extraction)* [20], [26]: A semi-supervised learning algorithm which does not require any more labeled data samples except initially labeled data.

For the proposed drift detection methods FRW, MRW, and ERW, the LDA and the SVM classifier with a linear kernel were used as the base model. The LDA classifier was coded by Matlab and the SVM classifier with the linear kernel is based on LibSVM. Three methods DDM, EDDM, EWMA require class labels for all the data samples and update the classification model incrementally. Therefore, SVM is not applicable for those methods because the incremental updating formula for SVM is not available. While uMD and CDBD using a linear SVM are only applicable for a two-class problem, the proposed methods are feasible for both a two-class problem and a multi-class problem by using LDA and SVM. FAST-COMPOSE applies the cluster-and-label method for class prediction of unlabeled data samples where majority vote is used in each cluster obtained by k-means clustering. Unlabeled data with predicted labels are used

Table 4 Experimental results using SVM as a base model

				The proposed methods		
		uMD	CDBD	FRW	MRW	ERW
Pendigits	F1-measure			**0.6914**	0.2394	0.6734
	accuracy	-	-	0.918	**0.9259**	0.7867
	usage (%)			4.118	15.0163	4.2712
Satellite	F1-measure			0.5467	0.3552	**0.5707**
	accuracy	-	-	0.8197	**0.8352**	0.7513
	usage (%)			4.6458	15.5079	4.7113
Nursery	F1-measure			0.1993	0.1948	**0.2683**
	accuracy	-	-	0.8693	**0.8753**	0.8673
	usage (%)			1.9984	16.0195	3.1194
Waveform	F1-measure			0.5523	0.4028	**0.5737**
	accuracy	-	-	0.7792	**0.8075**	0.7001
	usage (%)			5.3053	15.8316	5.9789
Magic	F1-measure	0.0739	**0.343**	0.0183	0.1171	0.0335
	accuracy	**0.8672**	0.8047	0.6883	0.8491	0.6939
	usage (%)	14.8874	3.8962	1.1511	18.3187	1.1401
Mushroom	F1-measure	0.2005	**0.547**	0.27	0.2123	0.3422
	accuracy	**0.956**	0.9415	0.9013	**0.957**	0.8615
	usage (%)	12.4919	5.7017	3.8875	22.4699	4.4318
Kr-vs-kp	F1-measure	0.2842	0.469	0.4157	**0.5237**	0.3797
	accuracy	**0.9375**	0.9235	0.8819	0.9114	0.8402
	usage (%)	21.5145	19.7628	6.2582	18.1159	6.39
Electricity	Accuracy	0.569	0.6525	**0.655**	0.6423	0.6328
	#drifts	167	1	17	71	40
	usage (%)	38.7957	0.4646	7.8985	32.988	18.5848
Covertype	Accuracy	0.7088	0.5077	0.5621	**0.7677**	0.7277
	#drifts	0	1	6	937	532
	usage (%)	0	0.0425	0.2551	39.8398	22.6199
Usenet	Accuracy	0.5777	0.5053	0.5488	**0.5781**	0.5428
	#drifts	17	0	5	7	5
	usage (%)	30.1739	0	17.7494	24.8491	17.7494

as labeled data for the application of the cluster-and-label method in the next chunk of unlabeled data.

To build the initial classification model, the first 5% of the data stream was used as the training data. The experiments were repeated 100 times by generating the stream data in random order and average prediction accuracy and F1-measure were computed. The size of the window in all the compared methods was set as $n = 200$. For k-means clustering in FAST-COMPOSE, k was set as the number of classes multiplied by 3/2.

When using the LDA as a base model, the experimental results are shown in Table 3, where "usage" means the proportion of required labeled data among the total data samples. F1-measure and accuracy in each case are denoted as a bold face. The proposed methods FRW, MRW, and ERW are compared with DDM, EDDM, EWMA, and FAST-COMPOSE. The proposed methods generally obtained high F1-measure on synthetic data, indicating that they detect concept changes accurately. Because the methods DDM, EDDM, EWMA update the model incrementally for each of data samples, we conjecture that they do not detect well small magnitudes of changes as drift. FAST-COMPOSE does not have the capability to detect concept drift, therefore F1-measure is not given. With regard to prediction accuracy, since FAST-COMPOSE intends to face the type of gradual drift and does not require any labeled data except initially given labeled data, it obtained low performance over all the data sets. DDM, EDDM, EWMA show slightly better results than the proposed methods. However, it is remarkable that the proposed methods used only approximately 7% of the labeled data on average, while DDM, EDDM, and EWMA use 100% of the labeling data. This indicates that the proposed methods are very effective at reducing the time and cost for obtaining the labels of the data, still maintaining the competitive prediction accuracy of a classifier. For the three real data sets, when using LDA as base model, the proposed methods have a little low prediction accuracy compared to three methods DDM, EDDM, and EWMA. However, labeling information is used at an approximate rate of 26%.

The results when using linear SVM as a base model are shown in the Table 4. While uMD and CDBD are not applicable for multi-class problems, FRW and ERW show the good F1-measure overall and MRW has best prediction accuracy. For the binary class problems, the proposed method MRW has higher F1-measure than uMD, which implies fewer false detection instances. For the three real data sets, the experimental results with SVM as a base model show that MRW among the proposed methods has the highest prediction accuracy. When compared in terms of the usage, the proposed methods use the labeling information less than uMD and more than CDBD.

4.4 Experimental Results with no Access to Class Labels

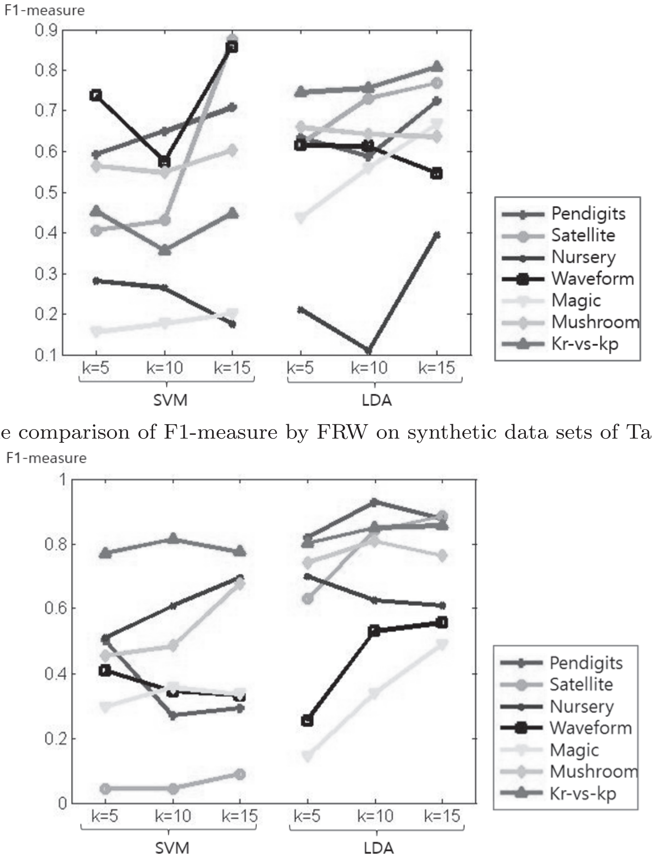
The proposed method was compared with *WW (Multivari-*

Table 5 The comparison by F1-measure on synthetic data sets

			SVM			LDA		
	WW	Error	FRW	MRW	ERW	FRW	MRW	ERW
Pendigits	0.0035	0.332	0.7082	0.1023	0.5917	**0.7237**	0.1294	0.7424
Satellite	0.0026	0.1505	**0.8763**	0.3234	0.5962	0.7683	0.2805	0.815
Nursery	0.0084	0.1255	**0.3943**	0.0216	0.3504	0.1751	0.0893	0.281
Waveform	0.003	0.14	**0.8572**	0.3299	0.7708	0.5462	0.2835	0.6536
Magic	0.0021	0.0719	0.2002	0.0686	0.2064	**0.6665**	0.0421	0.6615
Mushroom	0.0054	0.1312	0.6038	0.0267	0.6223	0.6362	0.2051	**0.6988**
Kr-vs-kp	0.0084	0.11	0.4472	0.4265	0.435	0.807	0.6128	**0.8578**

Table 6 The comparison of Fl-measure on synthetic data when the concept drift was generated according to the approach in the paper [21]

			SVM			LDA		
	WW	Error	FRW	MRW	ERW	FRW	MRW	ERW
Pendigits	0.0242	0.2	0.2816	0.4493	0.292	0.7921	0.7217	**0.8771**
Satellite	0.0168	0.5335	0.0919	0.3787	0.0902	0.8485	0.7222	**0.8839**
Nursery	0.0273	0.211	0.5619	0.1501	**0.6954**	0.5375	0.1413	0.6088
Waveform	0.0196	0.1	0.288	0.3807	0.3312	0.5515	0.4988	**0.5554**
Magic	0.0152	**0.706**	0.3181	0.0095	0.3355	0.4571	0.0312	0.4897
Mushroom	0.0035	0.1154	0.6367	0.0617	0.6774	0.7381	0.2892	**0.7627**
Kr-vs-kp	0.0021	0.1704	0.7696	0.4898	0.7747	0.8156	0.5018	**0.8553**



(a) the comparison of F1-measure by FRW on synthetic data sets of Table 5.

(b) the comparison of F1-measure by ERW on synthetic data sets of Table 6.

Fig. 4 The comparison of F1-measure when *k* was set as 5, 10, 15

ate Wald-Wolfowitz test) [9] and *Error* [21]. We conducted the experiments only on the synthetic data where the drift was manipulated artificially and therefore the drift point is known. As in the experiments in Sect. 4.3, the window size was set as 200 for all the compared methods. For the proposed method, grouping structure was obtained by k-means clustering method and virtual classifiers by using the LDA and SVM were constructed. Through all the experiments, *k* was set as 15.

Table 5 compares the performance by F1-measure on the synthetic data where concept drift was generated by feature shuffling in the same way as Sect. 4.3. FRW and ERW give the better results than all the compared methods. On the other hand, WW has low F1-measure values because it is very sensitive on the small magnitude of drift and issues too much of detection signals. Table 6 shows the experimental results on synthetic data when the artificial drift was generated according to the approach in the paper [21]. The concept drift occurs when the stream of data samples belonging to the largest class is switched to the data samples from the second largest class. The proposed method ERW based on LDA has higher F1-measure on six cases out of the seven synthetic data sets.

The graphs in Fig. 4 compare the performance when using different *k* values for the k-means clustering method in the proposed methods FRW and ERW. *k* was set as 5, 10, 15. The graph in Fig. 4 (a) shows F1-measure values obtained by the method FRW on synthetic data sets of Table 5. The graph in Fig. 4 (b) displays F1-measure values obtained by the method ERW on synthetic data sets of Table 6. In most of cases, the performance tends to get higher as *k* increases.

5. Conclusion

Concept drift detection is necessary for effective processing of data streams. In this paper, three drift detection approaches are proposed when the labeled data are limited or not available. The proposed method effectively combines the following aspects.

- When the access to the labels is limited, it utilizes the probability estimates for class prediction in order to describe the behavior patterns of classifier.
- Three approaches based on two types of windows, a reference window and a detection window, are proposed.
- When the access to the labels is not available at all, drift detection using a virtual classifier is performed.

In the experiments, the proposed methods give high F1-measure meaning that the proposed methods detect the change accurately. It can be said that the proposed methods are effective when used to reducing the labeling cost and the time required to obtain labels.

References

[1] I. Zliobaite, “Learning under Concept Drift: An Overview,” Tech. rep., Vilnius University, 2009.

[2] D. Malekian and M.R. Hashemi, “An adaptive profile based fraud detection framework for handling concept drift,” Information Security and Cryptology (ISCISC), 2013 10th International ISC Conference on, pp.1–6, IEEE, 2013.

[3] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, “A survey on concept drift adaptation,” ACM Computing Surveys, vol.46, no.4, pp.1–37, 2014.

[4] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, “Learning in nonstationary environments: a survey,” IEEE Comp. Int. Mag., vol.10, no.4, pp.12–25, 2015.

[5] I. Žliobaite, “Change with delayed labeling: when is it detectable?,” Proceedings of the IEEE international conference on data mining workshops, ICDMW ’10, IEEE Computer Society, Washington, DC, pp.843–850, 2010.

[6] P. Lindstrom, B.M. Namee, and S.J. Delany, “Drift detection using uncertainty distribution divergence,” IEEE 11th Int. Conf. Data Mining Work-shops, pp.604–608, 2011.

[7] T.S. Sethi and M. Kantardzic, “Don’t pay for validation: Detecting drifts from unlabeled data using margin density,” INNS Conference on Big Data, vol.53, pp.103–112, 2015.

[8] D.M.D. Reis, P. Flach, S. Matwin, and G. Batista, “Fast Unsupervised Online Drift Detection Using Incremental Kolmogorov-Smironv Test,” ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp.1545–1554, 2016.

[9] J.H. Friedman and L.C. Rafsky, “Multivariate generalizations of the Wald-Wolfowitz and Smirnov two-sample tests,” Annals of Statistics, vol.7, no.4, pp.697–717, 1979.

[10] Y.-I. Kim and C.H. Park, “Concept Drift Detection on Streaming Data under Limited Labeling,” IEEE International Conference on Com-puter and Information Technology, pp.273–280, 2016.

[11] J. Gama, P. Medas, G. Castillo, and P. Rpdrigues, “Learning with drift detection,” Proceedings of SBIA Brazilian Symposium on Artificial Intelli-gence, vol.3171, pp.286–295, 2004.

[12] M. Baena-Garcia, J. Campo-Avilla, R. Fidalgo, A. Bifet, R. Gavalda, and R. Moales-Bueno, “Early drift detection method,” Proceedings of ECML PKDD 2006 Workshop on Knowledge Discovery from Data Streams, 2006.

[13] G.J. Ross, N. Adams, D. Tasoulis, and D. Hand, “Exponentially weighted moving average charts for detecting concept drift,” Pattern recognition letters, vol.33, no.2, pp.191–198, 2012.

[14] A. Bifet and R. Gavalda, “Learning from Time-Changing Data with Adaptive Windowing,” Proc. SDM, pp.443–448, 2007.

[15] H. Wang, W. Fan, P.S. Yu, and J. Han, “Mining concept-drifting data streams using ensembe classifiers,” Proc. KDD, pp.226–235, 2003.

[16] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavaldà, “New ensemble methods for evolving data streams,” Proc. KDD, pp.139–148, 2009.

[17] X. Wu, P. Li, and X. Hu, “Learning from concept drifting data streams with unlabeled data,” Neurocomputing, vol.92, pp.145–155, 2012.

[18] P. Zhang, X. Zhu, and L. Guo, “Mining data streams with labeled and Uunlabeled training examples,” Proc. ICDM, pp.627–636, 2009.

[19] M.M. Masud, J. Gao, L. Khan, J. Han, and B. Thuraisingham, “A practical approach to classify evolving data streams: training with limited amount of labeled data,” Proc. ICDM, pp.929–934, 2008.

[20] K.B. Dyer, R. Capo, and R. Polikar, “COMPOSE: A semisupervised learning framework for initially labeled nonstationary streaming data,” IEEE Trans. Neural Netw. Learning Syst, vol.25, no.1, pp.12–26, 2014.

[21] A. Dries and U. Ruckert, “Adaptive concept drift detection,” Statistical Analysis and Data Mining, Special issue on the Best of SDM’09, vol.2, no.5-6, pp.311–327, 2009.

[22] S. Hido, T. Ide, H. Kashima, H. Kubo, and H. Matsuzawa, “Unsupervised Change Analysis using Supervised Learning,” Advances in Knowledge Discovery and Data Mining, pp.148–159, 2008.

[23] W. Navidi, Statistics for engineers and scientists, McGraw Hill, New York, 2006.

[24] D.C. Montgomery, Introduction to Statistical Quality Control, 6th edition, John Wiley & Sons.

[25] “UCI Machine Learning Repository,” http://archive.ics.uci.edu/ml/

[26] M. Umer, C. Frederickson, and R. Polikar, “Learning under extreme verification latency quickly: FAST COMPOSE,” Proc. 2016 IEEE Symp. Ser. Comp. Intell., pp.1–8, 2016.

Youngin Kim received her M.S. at the Department of Computer Science and Engineering, Chungnam National University, Korea. She is currently working in the information technology management division, Agency for Defense Development. Her research interests include data mining and machine learning.

Cheong Hee Park received her Ph.D. in Mathematics from Yonsei University, Korea in 1998. She received the M.S. and Ph.D. degrees in Computer Science at the Department of Computer Science and Engineering, University of Minnesota in 2002 and 2004 respectively. She is currently in the dept. of Computer Science and Engineering, Chungnam National University, Korea as a professor. Her research interests include pattern recognition, data mining, bioinformatics and machine learning.