Tracking Recurring Concepts from Evolving Data Streams using Ensemble Method

Yange Sun[1,2], Zhihai Wang[2], Jidong Yuan[2], and Wei Zhang[2]
[1]School of Computer and Information Technology, Xinyang Normal University, China
[2]School of Computer and Information Technology, Beijing Jiaotong University, China

Abstract: *Ensemble models are the most widely used methods for classifying evolving data stream. However, most of the existing data stream ensemble classification algorithms do not consider the issue of recurring concepts, which commonly exist in real-world applications. Motivated by this challenge, an Ensemble with internal Change Detection (ECD) was proposed to enhance performance by exploring the recurring concepts. It is done by maintaining a pool of classifiers, which dynamically adds and removes classifiers in response to the change detector. The algorithm adopts a two window change detection model, which adopts the Jensen-Shannon divergence to measure the distance of the distributions between old and recent data. When a change is detected, the repository of stored historical concepts is checked for reuse. Experimental results on both synthetic and real-world data streams demonstrate that the proposed algorithm not only outperforms the state-of-art methods on standard evaluation metrics, but also adapts well in different types of concept drift scenarios especially when concept s reappear.*

Keywords: *Data streams, ensemble classification, change detection, recurring concept, Jensen-Shannon divergence.*

Received August 10, 2016; accepted July 8, 2018

1. Introduction

Data stream mining has been receiving increased attention due to its presence in a wide range of application, such as sensor networks, financial applications, spam filtering systems, traffic control, and intrusion detection [11]. A data stream can be viewed as a potentially unbounded sequence of instances which arrive continuously with time-varying intensity.

Due to the non-stationary nature of data streams, the underlying distribution of the data may evolve over time, which is known as concept drift [29, 33]. For example, spam filtering systems and recommend systems of news both will be affected by concept drift because the features of spam usually change over time and the interested types of news for a person are probably different at different time [34]. Concept drift is best understood by customer preferences, which can be influenced by fashion trends or seasonal inclinations. Such changes lead to a drastic drop in classification accuracy. The greatest challenge in learning classifiers from data streams is reacting to concept drifts. A reasonable classifier should have the capability to recognize and respond to such changes quickly and accordingly.

Concept drift has been recently studied extensively [8, 13, 31]. Techniques developed to handle concept drift can be broken down into three main categories: instance selection, instance weighting and ensemble methods [29]. One of the most common approaches is based on instance selection and uses sliding window that moves over the recent instances, adds new instances while forgetting the oldest [3, 10]. Another way to track concept drift could be some kind of weighting instances. Instances can be weighted according to their age, and their competence with regard to the current concept [6, 18, 25]. Due to their modularity, ensemble methods also provide a natural way of adapting to change by modifying ensemble members. Ensemble classifiers consist of a set of concept descriptions from which only those chosen to be relevant to current concept are used in voting for the final decision of class label [5, 15, 19, 27, 30].

Concept drift can be categorized depending on their speed, into gradual and sudden drifts [13]. Sudden concept drift is characterized by large amounts of change between the underlying class distribution and the incoming instances in a relatively short amount of time. Gradual concept drift can require a very large amount of time to see a significant change in differences between the underlying class distribution and the incoming instances.

It is common in real-world data streams for previously-seen concepts may reappear in the future. For example, weather prediction models change according to the seasons, and a popular topic may appear in a social network at a particular time of the year (i.e., festivals or elections) [32]. This demonstrates a unique kind of drift, known as recurring concepts [32]. Although a lot of work has been done focusing on the concept drift problem, the recurring concept issue is largely unexplored. In fact, no matter what type of change occurs, the model should be able to track and adapt to changes accordingly.

If a concept reappears, the previously-learned classifiers should be reused, thus improving the performance of the learning algorithm. Since ensembles contain models built from different parts of the data stream, such models can be reused to classify new instances if they are drawn from a reoccurring concept. Therefore, a novel ensemble paradigm equipped with change detection mechanism is devised to cope with recurring concepts for data streams. The main contributions can be summarized as follows.

1. To address concept drift problem, the Jensen-Shannon divergence has been selected as a metric to measure the distribution between two consecutive windows which represent the older and the more recent instances respectively.
2. To solve recurring concept issue, an ensemble-based approach is introduced, which maintains a pool of classifiers (each classifier represents one of the existing concepts) and predicts the class of incoming instances using a weighted voting rule. Once a change occurs, a new classifier is learned and then identifying a new concept and added to the pool.
3. The performance of the proposed algorithm was evaluated on a variety of datasets, and a comprehensive comparison study of was presented. The results demonstrated that our method achieves better performance in terms of time consumption and classification accuracy than the state-of-the-art methods, especially when concepts reappear.

The remainder of this paper is organized as follows. Section 2 retrospects the related work. In section 3, the basic intuition behind our approach is discussed in detail. The algorithm is later analyzed and experimentally evaluated on real and synthetic datasets in section 4. Finally, some conclusions are drawn and future researches are discussed in section 5.

2. Related Work

2.1. Problem Statement

Suppose a supervised learning problem, where the learning algorithm observe sequences of pairs (x_i, y_i), where x_i is a vector of attribute values (nominal or numeric) and y_i is the class label for the ith instance in the stream. At each timestamp i, the learning algorithm outputs a class prediction $\hat{y}_i$ for the given feature vector x_i . A data stream S is a sequence instances $<s_1, s_2, \cdot \cdot \cdot >$, where each instance s_i is generated by some distribution P_i and each s_i is independent of the items that came before it. We say that a change drift has occurred if $P_i \neq P_{i+1}$, where P_i denotes the joint distribution at timestamp i between the set of input attributes and the class label. A recurring concept change occurs when the instances from a period k are generated from the same distribution as a previously observed period $P_k = P_{k-j}$.

As Gao *et al*. [14] have noted, the joint probability, which represents the data distribution $P(x, y) = P(y|x)\cdot P(x)$, can evolve over time in three different ways:

1. Changes in $P(x)$ known as virtual concept drift.
2. Changes in the conditional probability $P(y|x)$.
3. Changes in both $P(x)$ and $P(y|x)$ [13].

The change in the feature or input space $P(x)$ is called virtual concept drift and it may occur when the training instances are skewed. The change in the conditional distribution $P(y|x)$ is generally referred to as a real concept drift [13].

- *Definition* 1. Change detection is the process of segmenting a data stream into different segments by identifying the points where the stream dynamics changes [8].

To store datasets for comparison one can use a sliding window approach, i.e. one window refers to a contiguous segment of the data stream containing a specified number of data, another window that contains a small part of the entire dataset. The window model is chose for change detection in this study. The most common detecting strategy is based on monitoring the distance function between two distributions [29]. Kullback-Leibler divergence is the most popular distribution metric.

- *Definition* 2. Kullback and Leibler [20] divergence was introduced in, which measures the distance between the distributions of random variables, also called the relative entropy, is a robust metric for distance between two distributions. For two discrete distributions P and Q, there are two probability functions $p(x)$ and $q(x)$, Kullback-Leibler divergence is defined as

$$KL(P||Q)=\sum_{x\in\chi} p(x)\log\frac{p(x)}{q(x)} \quad (1)$$

Where x is the space of events. This measure is always nonnegative and is zero if and only if the distributions are identical. However, it is not symmetric and does not satisfy the triangle inequality.

To overcome this defect, Jensen-Shannon divergence [21] is adopted as the measure defined as Equation (2), which is a symmetric and bounded variant of the Kullback-Leibler divergence.

$$JS(P||Q)=\sum_{x\in\chi}(p(x)\log\frac{2p(x)}{p(x)+q(x)}+q(x)\log\frac{2q(x)}{p(x)+q(x)}) \quad (2)$$

When the underlying distribution changes from P to Q, the probability of observing certain subsequences under P is expected to be significantly higher than that under Q. Significance means that the ratio of the two probabilities is not smaller than a threshold. If the two distributions are identical, the value of JS is zero. The possibility of detecting recurring drifts allows reusing.

2.2. Classifiers for Dealing with Recurring Concepts

Most of ensemble approaches to deal with concept drift is to use some forgetting mechanism and train a new model [5, 15, 19, 27, 30]. In scenarios with recurrence, this implies to relearn a previously observed concept. Different classification algorithms or systems have been proposed, usually storing information about the concepts and, if necessary, reusing them.

Recurring concept drifts has been introduced by Widmer [32]. FLORA3 system [32] stores old concepts and reuses them when necessary. However, it is suitable for datasets with discrete attributes. Nishida and Yamauchi [23] introduced an online learning system called Adaptive Classifiers Ensemble (ACE), which generates ensemble of classifiers on sequential chunks of training instances. This property allows the algorithm to accurately react to recurring concepts by reusing previously trained classifiers. Similarly, an algorithm named Ensemble Building (EB) was introduced by Ramamurthy and Bhatnagar [24]. EB builds a global set of decision trees from sequential data chunks and a subset of them is used in the ensemble. Each classifier weight is based on its accuracy on the last data chunk.

Recurring Concept Drifts (RCD) is a framework that employs a multivariate non-parametric statistical test to determine whether both contexts are from the same distribution [16]. Abad *et al*. [1] proposed a system called MM-PRec, which integrates a meta-model mechanism and a fuzzy similarity function. In [26], a novel online approach named Extended Dynamic Weighted Majority with diversity (EDWM) was presented to deal with different types of drifts from slow gradual to abrupt drifts. Recently, a novel paradigm based on classifier graph for capturing and exploiting recurring concepts was proposed by Sun *et al*. [28].

3. Proposed Method

In this section, a novel concept drift detection will be presented first, and then an ensemble with internal drift detection is demonstrated in detail.

3.1. Window-based Concept Drift Detection

Most of the previous change detections [2, 3, 12, 17, 22] focus on detecting change in the error-rate of the classifiers, without considering changes in the distribution of data. In this section, the two-window paradigm is exploited, and it is also the most extensively used method by comparing the data distribution between two consecutive windows.

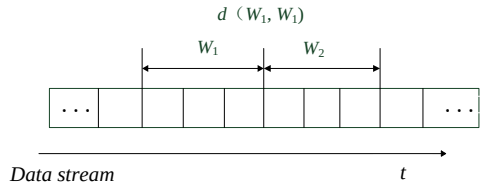

Figure 1. The two window change detection paradigm.

Let W_1 and W_2 denote reference window and current window respectively, and both the size of windows is n. As shown in Figure 1, the problem of change detection in data streams is to decide the null hypothesis H_0 against alternative hypothesis H_1 as below

$$\begin{cases} H_0 & d(W_1, W_2) \le \varepsilon \\ H_1 & d(W_1, W_2) > \varepsilon \end{cases} \tag{3}$$

Where $d(W_1, W_2)$ is a distance function which measures the dissimilarity of two windows and ε is a distance-based threshold used to decide whether a change occurs. A change occurs if the dissimilarity measure between two windows exceeds a threshold.

It can detect three kinds of concept drift: virtual concept drift, real concept drift and recurring concept drifts. For virtual concept drift, the formula of Jensen-Shannon divergence, named JS_{fc}, can be expressed as

$$JS_{fc}(W_1 \| W_2) = \sum_{x \in \mathcal{X}} \left(\hat{P}_{W_2}(x) \log \frac{2\hat{P}_{W_1}(x)}{\hat{P}_{W_1}(x) + \hat{P}_{W_2}(x)} + \hat{P}_{W_2}(x) \log \frac{\hat{P}_{W_2}(x)}{\hat{P}_{W_1}(x) + \hat{P}_{W_2}(x)} \right) \tag{4}$$

For real concept drift, the Equation of Jensen-Shannon divergence, named JS_{cc}, can be expressed as

$$JS_{cc}(W_1 \| W_2) = \sum_{x \in \mathcal{X}} \sum_{c \in Y} \left(\hat{P}_{W_2}(x|c) \log \frac{2\hat{P}_{W_1}(x|c)}{\hat{P}_{W_1}(x|c) + \hat{P}_{W_2}(x|c)} + \hat{P}_{W_2}(x|c) \log \frac{2\hat{P}_{W_2}(x|c)}{\hat{P}_{W_1}(x|c) + \hat{P}_{W_2}(x|c)} \right) \tag{5}$$

A concept drift is detected when the measure exceeds a threshold and a recurring concept is recognized when the measure results in a value of zero. The space partitioning scheme called the *kdq* tree [7] is used to construct multi-dimensional data stream, which approximate the underlying distribution. Meanwhile, Hoeffding Bound was adopted to get threshold.

- *Theorem* 1 (Hoeffding Bound) The Hoeffding bound [9] is stated as follows: with probability 1-δ, the estimated mean after n independent observations of range R will not differ from the true mean by more than ε, where

$$\varepsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}} \tag{6}$$

$\delta \in (0, 1)$ is a user defined confidence parameter.

The input of the change detection consists of a data stream S and window size n. The output is an alarm if a change occurs or a recurring concept is recognized. At the initial stage, the reference is initialized with the first batch of data stream. The current window moves

on the data stream and captures the next batch of data stream. When a change is detected, the change detector makes an alarm. If a change does not occur, the basic window W_2 slides step by step until the change detected. The pseudo-code of the concept drift detection is presented in Algorithm 1.

Algorithm 1. The pseudo-code of the window-based change detection

Parameters: window size n
Input: Stream S
Output: change alarm
01:Initialization t=0;
02: Set reference window $W_1=\{x_{t+1}, \ldots, x_{t+n}\}$;
03: Set current window $W_2=\{x_{t+n+1}, \ldots, x_{t+2n}\}$;
04: while not the end of S do
05: if $d(W_1, W_2) > \varepsilon$ *then*
06: t ← current time;
07: Report a change alarm at time t;
08: Clear all the windows and goto 02;
09: else if $d(W_1, W_2)==0$ *then*
10: alarm a recurring concept;
11: else slide W_2 *by 1 point;*
12: end if;
13: end if
14: end while
15:end.

3.2. Efficient Ensemble for Dealing with Recurring Concepts

We first give the general framework depicted in Figure 2, and then describe the particular model and its analysis. In this paper, stored models are combined in a dynamic weighted ensemble to deal with recurring concepts. Furthermore, a change detector to monitor the distribution between two consecutive windows which represents the older and the more recent instances respectively. Once a change occurs, a new classifier is learned and then identifying a new concept and added to the pool.

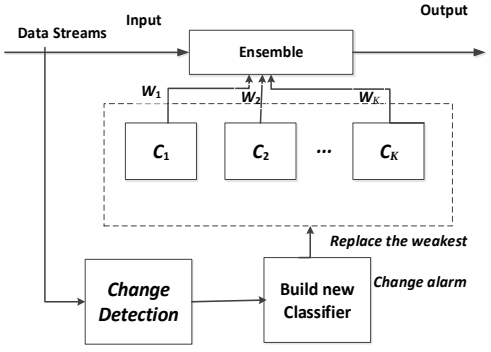

Figure 2. The framework of our method.

Let S be a data stream, E represents the ensemble. For each incoming instance x_i, each ensemble member $C_i \in E$ is weighted according to the expected prediction error on the most recent data according to Equation (7).

$$\begin{aligned} \text{Weight}(C_i) &= MSE_r - MSE_i \\ MSE_i &= \frac{1}{|W|}\sum_{(x_i, y_i)\in W}(1 - f^i_{y_i}(x_i))^2 \\ MSE_r &= \sum_{y_i} p(y_i)(1 - p(y_i))^2 \end{aligned} \tag{7}$$

Where MSE_i estimates the prediction error of classifier C_i on the instances of the most recent window W_2, while MSE_r is the mean square error of a randomly predicting classifier and is used as a reference point to the current class distribution. Function $f^i_{y_i}(x_i)$ denotes the probability given by classifier C_i that x_i is an instance of class y_i. The pseudo-code of algorithm which deals with the recurring concepts is presented in Algorithm 2.

Algorithm 2. The pseudo-code of ECD

Input: Stream S; pool size k;
Output: E: a pool of classifiers;
01: $E \leftarrow \varnothing$;
02: for each instances $x_t \in S$ *do*
03: $W \leftarrow W \cup \{x_t\}$;
04: if change detected==true then
05: if the concept is known then
07: reuse the classifier in E;
08: else create a new classifier C';
09: if $|E|<k$ *then* $E \leftarrow E \cup C'$;
10: else prune the most infrequent classifier;
11: end if;
12: end if;
13: end if;
14: $t++$;
15: end for;
16: end.

Instead of using the fixed threshold, Hoeffding Bound is employed to estimate a suitable threshold dynamically. In order to keep the number of classifier in a reasonable range, a forgetting strategy was designed to discard the most infrequent used classifier. Hoeffding Tree is selected as the base classifier, but one could choose any online learning algorithm.

4. Experimental Evaluation

The experiments are implemented in Java with help of Massive Online Analysis (MOA) [4]. The experiments were conducted on 3.0 GHz Pentium PC machines with 16 GB of memory, running Microsoft Windows 10.

4.1. Datasets

In our experiments, we adopt three synthetic and three real-world datasets. These datasets are summarized in Table1.

Table 1. Characteristics of the datasets.

Dataset	Instances	Attributes	Classes	Noise
RBF	1M	10	2	0%
HyperPlane	1M	10	2	5%
SEA	1M	3	4	10%
Elist	1, 500	913	2	-
Spam	9, 324	850	2	-
Sensor	2, 219, 803	5	54	-

In our experiments, the synthetic datasets containing three types of drift: gradual, sudden and recurring concept drift.

Radial Basis Function (RBF) generator creates a user specified number of centroids and assigns each incoming instance to one centroid with the probability given by that center's weight. This generator was used to produce a stream contains 1, 000, 000 instances with no drift.

HyperPlane is a two-class dataset that models a rotating hyperplane in a *d*-dimensional space. It is represented by the set of points x that satisfy $\sum_{i=1}^{d} w_i x_i = w_0$, where x_i is the ith coordinate of x. Instances for which $\sum_{i=1}^{d} w_i x_i \geq w_0$ are labeled positive, and instances for which $\sum_{i=1}^{d} w_i x_i < w_0$ are labeled negative. This generator was used to create a dataset contains 1, 000, 000 instances with gradual drifts by the modification weight w_i changing by 0.001 with each instance, and added 5% noise to streams.

SEA dataset consists of three attributes, where only two are a relevant attributes. The points of the dataset are divided into four blocks with different concepts. In each block, the classification is done using $f_1+f_2 \leq \Theta$, where f_1 and f_2 represent the first two attributes and Θ is a threshold value. The most frequent values are 9, 8, 7 and 9.5 for the data blocks. It contains 1, 000, 000 instances with sudden drifts reappearing every 250, 000 instances, and 10% of noise.

When under the real-world situations, it is not possible to know exactly which type of drift is present, or even if there really is a drift. Hence, it can better verify the performance of the algorithm. We simulated all the datasets into streams by MOA generators.

Emailing list (Elist) contains a stream of emails on various topics which are shown to the user one after another and are marked as interesting or junk. It is composed of 1, 500 instances with 913 attributes and is divided into 5 periods every 300 instances. It represents the problem of sudden concept drift and recurring concept. It can be obtained at http://mlkd.csd.auth.gr/ concept_drift.html.

Spam dataset is a real-world text data stream that represents a gradual concept drift. It consists of 9, 324 instances with 850 attributes. There are two classes that represent the two types of messages: legitimate and spam messages. Spam messages constitute around 20% of the dataset. It can be obtained at http://mlkd.csd.auth.gr/ concept_drift.html.

Sensor dataset contains data collected from 54 sensors in the Intel Berkeley Research Laboratory over a period of 2 months. It consists of 2, 219, 803 instances and 5 attributes, which contain the temperature, humidity, light, voltage of the sensor and the sensor ID. The sensor ID is the class label, it contains 54 classes. As the attributes such as brightness and temperature will change over time during a day, concept drift may occur. It can be obtained at http://www.cse.fau.edu/~xqzhu/ stream.html.

4.2. Results and Discussion

The proposed algorithm was evaluated against the following methods: Adaptive Random Forest (ARF), Recurring Concept Drifts (RCD) and Hoeffding Tree (HT). To the best of our knowledge, ARF is the state-of-the-art algorithm on these data streams, so we will compare it against our algorithm. RCD was chosen as it is the classifier which tried to improve upon. Hoeffding Tree was chosen as representatives of single classifiers. HT and ARF are available implementation in the MOA framework. RCD is available at http://sites.google.com/ site/moaextensions/ as MOA extensions. All algorithms chose the default MOA parameters.

To evaluate the effectiveness of the methods, we adopted the prequential evaluation method [4]. All the tested ensembles used k=15 component classifiers. Hoeffding Tree with default MOA parameters was chosen as base learner in all tested methods.

4.2.1. Parameter Sensitiveness

The experiment is designed to analyze the influence of window size. The size of window n was varied from 500 to 2000 to see how it affects the performance of the algorithms. Table 2 presents the accuracy of ECD on different datasets while using different window size. Figure 3 presents the accuracy of the comparison algorithms on SEA while using different window sizes.

Table 2. Accuracy using different window sizes.

	window size			
	500	**1000**	**1500**	**2000**
RBF	81.16	81.36	80.95	**82.01**
HyperPlane	71.69	72.23	71.61	71.25
SEA	76.68	**78.69**	78.02	77.01
Elist	72.23	72.59	72.15	72.03
Spam	93.16	**93.26**	93.08	93.03
Sensor	80.29	84.78	**89.01**	88.21

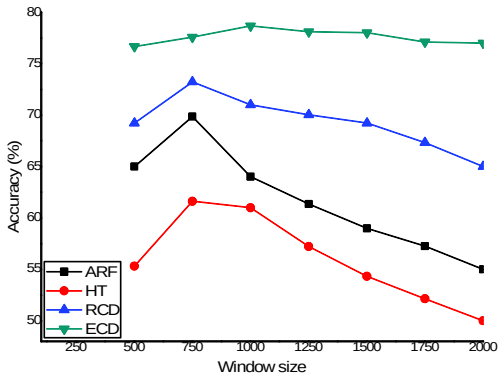

Figure 3. Accuracy using different window sizes.

The shape of curves that demonstrate the relationship between the size of the time window and the accuracy of the classification is shown in Figure 3. We observe that a too small window size will not provide enough amounts of data for a new classifier to

be accurate, whereas a too large window size may contain more than one concept, failing to adapt to current concept drifts quickly. ECD is not quite sensitive to the size of window. As there is no strong dependency upon the window size in terms of accuracy, but time consume is proportional, so we chose w=1000 as the value in the comparison experiments.

4.2.2. Comparative Performance Study

This part demonstrated the experimental results regarding the effectiveness and efficiency of our proposed method. The performance can be evaluated in terms of classification accuracy and time consumption.

Table 3. Comparison of Classification Accuracy (%).

	RCD	ARF	HT	ECD
RBF	72.27 (4)	**84.53 (1)**	82.79 (2)	81.36 (3)
HyperPlane	67.21 (3)	**78.57 (1)**	63.47 (4)	72.23 (2)
SEA	73.24 (2)	68.84 (3)	60.63 (4)	**81.69 (1)**
Elist	65.36 (2)	61.37 (4)	61.45 (3)	**72.59 (1)**
Spam	89.23 (3)	89.73 (2)	75.36 (4)	**93.26 (1)**
Sensor	84.32 (3)	**87.79 (1)**	80.12 (4)	86.28 (2)
Average Rank	2.83	2.00	3.50	1.67

Table 4. Comparison of Time Consumption (Cpu Seconds).

	RCD	ARF	HT	ECD
RBF	**8.38 (1)**	80.23 (4)	11.42 (3)	10.21 (2)
HyperPlane	14.40 (2)	103.23 (4)	**13.47 (1)**	15.35 (3)
SEA	4.56 (2)	14.03 (4)	6.41 (3)	**3.87 (1)**
Elist	1.36 (3)	4.37 (4)	**0.98 (1)**	1.24 (2)
Spam	17.36 (4)	11.80 (3)	**4.36 (1)**	7.02 (2)
Sensor	57.28 (4)	40.67 (2)	**38.25 (1)**	45.28 (3)
Average Rank	2.67	3.50	1.67	2.17

1. In terms of accuracy, ECD obtains the overall best performance on most of the datasets. On the dataset with no drift (RBF), ECD, ARF and RCD performed almost identically, with HT being slightly less accurate. For the dataset with gradual concept drift (HyperPlane), ARF is the best, followed by ECD. However, ECD seems to be the most accurate in the case of sudden changes (SEA). On Elist, ECD is the most accurate followed by RCD. On Spam, our method clearly outperformed all the other algorithms. On Sensor, ARF and ECD are obviously superior to HT and RCD.

HT performed worse than the ensemble methods. It might be attributed to the fact that ensemble classifiers adopt a combination of models to obtain better predictive performance than the single model. Moreover, HT lacks of a detection mechanism and therefore adapts ineffectively to drift. Also, notice that RCD and ARF did not perform as well as our method due to the fact that they had to wait to accumulate instances into a batch before learning. It can be seen from Table 3 that RCD seems to improve classification accuracy only on certain datasets. These results clearly demonstrate that ECD consistently boosts the accuracy on both synthetic and real-world scenarios. This is partly because the management of the recurrent change detection mechanism is capable of reusing previous concepts and gains the better performance in different situations, particularly under recurring concept drift environments.

2. As shown in Table 4, single classifier likes HT requires the least time, following by our algorithm, and ARF is the longest time-consuming. An interesting finding is that although HT consumes less time, but achieves the least classification accuracy rate. It is demonstrated that the time usage of the distribution change detection strategy. The ECD is activated very quickly if the concepts that they represent reappear. It is partly because that our system does not relearn if a historic concept reappears. RCD manages to recover faster in all cases exploiting and updating older models. On the datasets with recurrent concept drift (SEA, Elist), RCD and ECD are able to track of changes much more efficiently and quickly adapt their model to novel characteristics of incoming data. Additionally, RCD requires labels to detect changes, so detect a drift may cause a delay, or even be impossible to use when we have a limited access to true class labels. ECD does not require any true class labels, only according to the distribution of incoming data. So, it consumes less time.

When analyzing the detailed results of the SEA with recurring sudden concept drifts. Analyzing the values in Figure 4, all the curves suffer significant decrease when the concept drifts occur. Unfortunately, HT does not perform very well in the presence of changes, possibly due to the fact that HT always learns a concept from scratch upon trigger detection no matter whether this concept is recurrent. ECD is able to track these changes immediately, and reused the classifiers in real-time. These results confirm that ECD is obviously superior to ARF and RCD in this recurring sudden change environment. Block-based approach (ARF and RCD) did not performing as well as ECD the fact that they are designed to cope with gradual concept drifts.

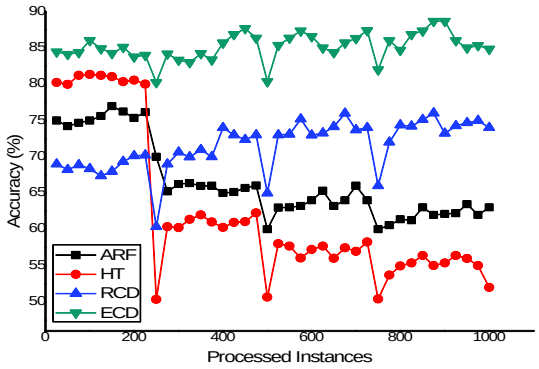

Figure 4. Accuracy on the SEA.

Real-world stream environment can be better verifying the capability of deal with different kinds of drift (sudden, gradual, real, and virtue). Figure 5 depicts the accuracy curves on Elist. HT and ARF

perform similarly. Notice the sudden drops of all methods at drift time points (after 300, 600, 900 and 1, 200 instances), except our method. HT shows a sudden drop in accuracy when changes occur. ARF does not incorporate a drift detection mechanism to cope with recurrent concept drift. ECD and RCD manage to recover faster in all cases exploiting and updating older models. The curve of the ECD was relatively stable, subjecting to data concept drift minimal impact on real data, which showed that the algorithm had better adaptability for real-world environment. The results clearly demonstrate that the realization of the recurring concept of repeated reuse, improves the performance of the classifier when the occurrence of a drift.

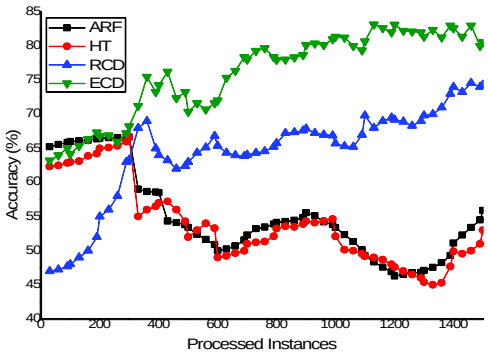

Figure 5. Accuracy on the Elist.

To sum up, ECD performs better than the other methods from the following viewpoints:

1. ECD is able to construct a satisfactory model for handling different kinds of concept drifts (sudden, gradual, real, and virtue) and has been specifically built to cope with recurring concepts.
2. Compared to other ensembles, ECD can achieve better predicting accuracy and significantly reduced the time consumption, especially when concept recurring.

5. Conclusion and Future Work

In this study, we focus on the issue of recurring concept drifts, a special subtype of concept drifts, which has not yet drawn enough attention from the research community. In this paper, we introduce an efficient ensemble equipped with internal change detector, which can reuse previously-trained classifiers, to handle recurring concepts. It works by storing past concepts as classifiers and instances used to construct the classifiers attempt to find one that is fit for the current situation. Meanwhile, a two-window change detection paradigm chooses the Jensen-Shannon divergence as a measure to compare the distributions between old and recent data, whose distribution under the null hypothesis of coming from the same distribution is approximated. Once a concept drift is detected, a learnt model is stored. Therefore, it can be reused in the situations of recurrence. The experimental results demonstrate that our method is both stable enough on the datasets with gradual concept drifts and flexible enough to adapt to sudden concept drifts and recognize recurring concept.

Despite the growing number of efforts, there still remains much work to be done in data streams that exhibit recurring concept. In our future work, we plan to devise an effective procedure which will eliminate redundant classifiers without decreasing the ability to deal with recurring concepts.

Acknowledgments

This work is supported by National Natural Science Foundation of China (No. 61672086, 61702030, 61771058, 61702550, 61563001), Beijing Natural Science Foundation (No. 4182052), the Innovation Team Support Plan of University Science and Technology of Henan Province (No. 19IRTSTHN014), Key Scientific Research Projects of Higher Education Institutions in Henan Province(20B520030), Fundamental Research Funds for the Central Universities (2018JBM014) and Excellent Young Teachers Program of XYNU (No. 2016GGJS-08).

References

[1] Abad M., Gomes J., and Menasalvas E., "Predicting Recurring Concepts on Data-Streams By Means of A Meta-Model And A Fuzzy Similarity Function," *Expert Systems with Applications*, vol. 46, no. 1, pp. 87-105, 2016.

[2] Baena-García M., Campo-Ávila D., and Fidalgo R., "Early Drift Detection Method," *in Proceedings of the 4th International Workshop on Knowledge Discovery from Data Streams*, New York, pp. 77-86, 2006.

[3] Bifet A. and Gavalda R., "Learning from Time-Changing Data with Adaptive Windowing," *in Proceedings of the 7th SIAM International Conference on Data Mining*, Minneapolis, pp. 443-448, 2007.

[4] Bifet A., Holmes G., Kirkby R., and Pfahringer B., "MOA: Massive Online Analysis," *Journal of Machine Learning Research*, vol. 11, pp. 1601-1604, 2010.

[5] Brzezinski D. and Stefanowski J., "Reacting to Different Types of Concept Drift: The Accuracy Updated Ensemble Algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 81-94, 2014.

[6] Cohen E. and Strauss M., "Maintaining Time-Decaying Stream Aggregates," *Journal of Algorithms*, vol. 59, no. 1, pp. 19-36, 2006.

[7] Dasu T., Krishnan S., Venkatasubramanian S., and Yi K., "An Information-Theoretic Approach To Detecting Changes in Multi-Dimensional Data Streams," *in Proceedings of the 38th*

Symposium on the Interface of Statistics, Computing Science, and Applications, Pasadena, pp. 1-23, 2006.

[8] Ditzler G., Roveri M., Alippi C., and Polikar R., "Learning in Nonstationary Environments: A Survey," *IEEE Computational Intelligence Magazine*, vol. 10, no. 4, pp. 12-25, 2015.

[9] Domingos P. and Hulten G., "Mining High-Speed Data Streams," *in Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Boston, pp. 71-80, 2000.

[10] Dries A. and Ruckert U., "Adaptive Concept Drift Detection," *Statistical Analysis and Data Mining*, vol. 2, no. 5-6, pp. 311-327, 2009.

[11] Gama J., *Knowledge Discovery from Data Streams*, CRC Press, 2010.

[12] Gama J., Medas P., Castillo G., and Rodrigues P., "Learning with Drift Detection," *in Proceedings of the 17th Brazilian Symposium on Artificial Intelligence*, São Luís, pp. 286-295, 2004.

[13] Gama J., Žliobaitė I., Bifet A., Pechenizkiy M., and Bouchachia A., "A Survey on Concept Drift Adaptation," *ACM Computing Surveys*, vol. 46, no. 4, pp. 231-238, 2014.

[14] Gao J., Fan W., Han J., and Yu P., "A general Framework for Mining Concept-Drifting Data Streams with Skewed Distributions," *in Proceedings of the 7th SIAM International Conference on Data Mining*, Minneapolis, Minnesota, pp. 3-14, 2007.

[15] Gomes H., Bifet A., Read J., Barddal J., Enembreck F., Pfharinger B., and Holmes G., "Adaptive Random Forests for Evolving Data Stream Classification," *Machine Learning*, vol. 106, no. 9-10, pp. 1469-1495, 2017.

[16] Gonçalves P. and Barros R., "RCD: A Recurring Concept Drift Framework," *Pattern Recognition Letters*, vol. 34, no. 9, pp. 1018-1025, 2013.

[17] Kifer D., Ben-David S., and Gehrke J., "Detecting Change in Data Streams," *in Proceedings of the 30th International Conference on Very Large Data Bases*, Toronto, pp. 180-191, 2004.

[18] Klinkenberg R., "Learning Drifting Concepts: Example Selection vs. Example Weighting," *Intelligent Data Analysis*, vol. 8, no. 3, pp. 281-300, 2004.

[19] Kolter J. and Maloof M., "Dynamic Weighted Majority: An Ensemble Method for Drifting Concepts," *Journal of Machine Learning Research*, vol. 8, pp. 2755-2790, 2007.

[20] Kullback S. and Leibler R., "On information and Sufficiency," *The Annals of Mathematical Statistics.* , vol. 22, no. 1, pp. 79-86, 1951.

[21] Lin J., "Divergence Measures Based on The Shannon Entropy," *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 145-151, 1991.

[22] Nishida K., Yamauchi K., and Omori T., "ACE: Adaptive Classifiers-Ensemble System for Concept-Drifting Environments," *in Proceedings of the 6th International Workshop Multiple Classifier Systems*, Seaside, pp. 176-185, 2005.

[23] Nishida K. and Yamauchi K., "Detecting Concept Drift Using Statistical Testing," *in Proceedings of the 10th International Conference on Discovery Science*, Sendai, pp. 264-269, 2007.

[24] Ramamurthy S. and Bhatnagar R., "Tracking Recurrent Concept Drift in Streaming Data Using Ensemble Classifiers," *in Proceedings of the 6th International Conference on Machine Learning and Applications*, Cincinnati, pp. 404-409, 2007.

[25] Ross G., Adams N., Tasoulis D., and Hand D., "Exponentially Weighted Moving Average Charts for Detecting Concept Drift," *Pattern Recognition Letters*, vol. 33, no. 2, pp. 191-198, 2012.

[26] Sidhu P. and Bhatia M., "Online Approach to Handle Concept Drifting Data Streams Using Diversity," *The International Arab Journal of Information Technology*, vol. 14, no. 3, pp. 293-299, 2017.

[27] Street W. and Kim Y., "A Streaming Ensemble Algorithm for Large-Scale Classification," *in Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, pp. 377-382, 2001.

[28] Sun Y., Wang Z., Bai Y., Dai H., and Nahavandi S., "A Classifier Graph Based Recurring Concept Detection and Prediction Approach," *Computational Intelligence and Neuroscience*, vol. 2018, pp. 1-13, 2018.

[29] Tsymbal A., the Problem of Concept Drift: Definitions and Related Work, Technical Report, Department of Computer Science, Trinity College, 2004.

[30] Wang H., Fan W., Yu P., and Han J., "Mining Concept-Drifting Data Streams Using Ensembles Classifiers," *in Proceedings of 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, pp. 226-235, 2003.

[31] Webb G., Hyde R., Cao H., Nguyen H., and Petitjean F., "Characterizing Concept Drift," *Data Mining and Knowledge Discovery*, vol. 30, no. 4, pp. 964-994, 2016.

[32] Widmer G., "Tracking Context Changes Through Meta-Learning," *Machine Learning*, vol. 27, no. 3, pp. 259-286, 1997.

[33] Widmer G. and Kubat M., "Learning in the Presence of Concept Drift and Hidden Contexts," *Machine Learning*, vol. 23, no. 1, pp. 69-101, 1996.

[34] Zliobaite I., Pechenizkiy M., and Gama J., *Big Data Analysis: New Algorithms for a New Society, Studies in Big Data*, Springer, 2016.

Yange Sun received the M.S. degree from the Central China Normal University, in 2007, both in computer science. She is currently pursuing thePh.D. degree with the Department of Electrical andComputer Engineering, Beijing Jiaotong University. Her research interests include data mining and machine learning.

Zhihai Wang received the Doctor's Degree in Computer Application from Hefei University of Technology in 1998. He is now a Professor in School of Computer and Information Technology, Beijing Jiaotong University. He has published dozens of papers in international conferences and journals. His research interest includes data mining and artificial intelligence.

Jidong Yuan received the M.S. degree and Ph.D. degree in Computer Science and Technology from Beijing Jiaotong University, in 2012 and 2016, respectively. He is currently a lecturer in the School of Computer and Information Technology, Beijing Jiaotong University. His research interests include data miningand pattern recognition.

Wei Zhang received the M.S. degree in Computational Mathematics from Guilin University of Electronic Technology, in 2015. He is currently a Ph.D. candidate in School of Computer and Information Technology, Beijing Jiaotong University. His research interests include machine learning and data mining.