CONCEPT DRIFT DETECTION WITH HIERARCHICAL HYPOTHESIS TESTING

Shujian Yu[1], *Zubin Abraham*[2]

[1]Department of Electrical and Computer Engineering, University of Florida, FL
[2]Robert Bosch Research and Technology Center, CA

ABSTRACT

When using statistical models (such as a classifier) in a streaming environment, there is often a need to detect and adapt to concept drifts so as to mitigate any deterioration in the models predictive performance over time. Unfortunately, the ability of popular concept drift approaches in detecting these drifts in the relationship of the response and predictor variable is often dependent on the distribution characteristics of the data streams, as well as its sensitivity on parameter tuning. This paper presents Hierarchical Linear Four Rate (HLFR), a framework that not only detects concept drifts for different data stream distributions (including imbalanced data), but also leverages its hierarchical nature to dynamically adapt the parameters of the hypothesis tests to better detect upcoming concept drifts. The performance of HLFR is compared to benchmark approaches using both simulated and real-world datasets spaning the breadth of concept drift types. HLFR is shown to significantly outperform benchmark approaches in terms of accuracy, G-mean, recall, delay in detection and adaptability across the various datasets.

I. INTRODUCTION

Numerous real-world applications such as fraud detection, user preference prediction, email filtering, etc. [1] capture intrinsically changes in the relationship of the incoming data streams. Current approaches to address these concept drift when using statistical models fall into two categories [2], [3]. The first approach is to automatically adapt the parameters of the statistical model in an incremental fashion, as new data is observed. The second approach is to have a concept drift detector in addition to the statistical model, whose purpose is to signal the need for retraining the statistical model, on account of having observed a concept drift in the data. Unlike the first approach that focuses only on mitigating the effect of concept drift, the second approach also help identify when concept drift has occurred. This paper focuses on concept drift detection approaches.

A popular concept drift detection approach is Drift Detection Method (DDM) [1]. The test statistic DDM monitors is the sum of overall classification error ($\hat{P}^{(t)}_{error}$) and its empirical standard deviation ($\hat{S}^{(t)}_{error} = \sqrt{\hat{P}^{(t)}_{error}(1-\hat{P}^{(t)}_{error})/t}$). Since DDM focuses on the overall error rate, it fails to detect a drift unless the sum of false positive and false negatives changes. This limitation is accentuated when detecting concept drift in imbalanced classification tasks. Early Drift Detection Method (EDDM) [4] was proposed to achieve better detection results when dealing with slow gradual changes via monitoring the distance between the two classification errors. However it requires to wait for a minimum of 30 classification errors before calculating the monitoring statistic at each decision point which is not well suited for imbalanced data. A third loss-monitoring method, $i.e.$, STEPD [5], compares the accuracy on a recent window with the overall accuracy excluding the recent window by applying a test of equal proportion.

Drift Detection Method for Online Class Imbalance (DDM-OCI) [6] address the limitation of DDM when data is imbalanced. However, DDM-OCI triggers a number of false positives due to an inherent weakness in the model. DDM-OCI assumes that the concept drift in imbalanced streaming data classification is indicated by the change of underlying true positive rate ($i.e.$, minority-class recall). This hypothesis unfortunately does not account for scenarios when concept drift occurs without affecting minority-class recall. For instance, it is very possible that the underlying concept drifts from imbalanced data to balanced data, while the true positive rate (TPR), positive predictive value (PPV) and F-measure remain unchanged. This type of drift is unlikely to be detected by DDM-OCI. The test statistic used by DDM-OCI $\hat{R}^{(t)}_{TPR}$ is also not approximately distributed as $\mathcal{N}(P^{(t)}_{TPR}, \frac{P^{(t)}_{TPR}(1-P^{(t)}_{TPR})}{t})$ under the stable concept[1]. Hence, the TPR rationale of constructing confidence levels specified in [1] is not suitable with the null distribution of $\hat{R}^{(t)}_{TPR}$. This is the reason DDM-OCI triggers false positives quickly and frequently. Linear Four Rates (LFR) was proposed to address the limitation of DDM-OCI by monitoring the four rates associated with the confusion matrix of the data stream [7]. Although, LFR performs much better than DDM-OCI, it still triggered false alarms.

To address the limitations of existing approaches, we present two-layered hierarchical hypothesis testing framework (HLFR) for concept drift detection. Unlike other proposed approaches, HLFR can not only detect all possible variants of concept drift with least false alarms, it can do so even in the presence of imbalanced class labels. HLFR is independent of the underlying classifier and outperforms existing approaches in terms of earliest detection of concept drift, with the least false alarms and best $\mathbf{Precision}$.

[1]$\hat{R}^{(t)}_{TPR}$ is a modified estimator of $P^{(t)}_{TPR}$, which satisfies $\hat{R}^{(t)}_{TPR} = \eta\hat{R}^{(t-1)}_{TPR} + (1-\eta)1_{y_t=\hat{y}_t}$ where η denotes a time decaying factor [6].

I-A. Problem Formulation

We are given a continuous stream of labeled streaming samples $\{\mathbf{X}_t, y_t\}$, $t = 1, 2, ...$, where $\mathbf{X}_t$ is a d-dimensional vector in a pre-defined vector space $\mathcal{X} = \mathbb{R}^d$ and $y_t \in \{0, 1\}$[2]. At every time point t we split the samples in a set S_A of n_A recent ones and a set S_B containing n_B examples that appeared prior to those in S_A. We would now like to know whether or not the mapping f from $\mathbf{X}_t$ to y_t in S_A were the same as in S_B.

A closely related topic to concept drift detection is the well-known change-point detection problem in machine learning and statistics community, whereas the goal of the latter is to detect changes in the generating distributions $P(\mathbf{X}_t)$ of the streaming data. The standard tools for change-point detection are methods from statistical decision theory. These methods usually compute a statistic from the available data, which is sensitive to changes between the two sets of examples. The measured values of the statistic are then compared to the expected value under the null hypothesis that both samples are from the same distribution. The resulting p-value can be seen as a measure of the strength of the drift. A good statistic must be sensitive to data properties that are likely to change by a large margin between samples from differing distributions.

Although a drift in generating distribution $P(\mathbf{X}_t)$ may result in a change in the learning problem, the detection of any type of distributional change remains a challenge, especially when $\mathbf{X}_t$ is high-dimensional data [3], [7]. In this paper, to achieve high detection accuracy, we adopt the principle of risk minimization [8] and solve the problem directly by monitoring the “significant” drift in the prediction risk (*i.e.*, classification loss) of the underlying predictor rather than the intermediate problem of change-point detection. This is motivated by the fact that any drift of $P(\hat{f}(\mathbf{X}_t), y_t)$ would imply a drift in $P(\mathbf{X}_t, y_t)$ with probability 1, where $\hat{f}$ is the incrementally learned predictor (or classifier).

II. HIERARCHICAL LINEAR FOUR RATE (HLFR)

This paper presents a two-layered hierarchical hypothesis testing framework (HLFR) for concept drift detection. Once a potential drift is detected by the Layer-I test of HLFR, the Layer-II test is performed to confirm (or deny) the validity of the suspected drift. The results of the Layer-II feeds back to the Layer-I of the framework, which is then used to update the parameters of the Layer-I. The HLFR framework differs from the mere execution of two subsequent tests as the two layers cooperate by exchanging information about the detected drift to improve online detection ability[3], as shown in Fig.1.

HLFR treats the underlying classifier as a black-box and does not make use of any of its intrinsic properties. This modular property of the framework allows it be deployed

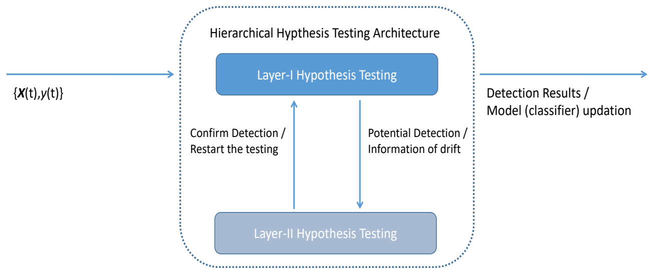

Hierarchical Hypthesis Testing Architecture
Layer-I Hypothesis Testing
{X(t),y(t)}
Detection Results / Model (classifier) updation
Confirm Detection / Restart the testing
Potential Detection / Information of drift
Layer-II Hypothesis Testing

Fig. 1: The architecture of the proposed hierarchical hypothesis testing framework for concept drift detection.

alongside any classifier (decision trees, neural networks, *etc.*) unlike concept drift detectors that are designed to only work with (*e.g.*) linear discriminant classifiers [9], or support vector machines (SVM) [10]. Without loss of generality, this paper selects soft margin SVM as the baseline classifier due to its universality and stability [11].

Given the robustness of detecting concept drift by monitoring the four rates of the confusion matrix streams (more specifically, true positive rate (tpr), true negative rate (tnr), positive predictive value (ppv) and negative predictive value (npv)), even in the presence of imbalanced class labels [7], HLFR uses hypothesis tests that monitor these four rates, in its Layer-I LFR. The second layer of HLFR uses the test statistic (or quantity) strictly related to that used at the Layer-I to conduct a permutation test. If a drift is confirmed, the HFLR framework signals a detection, otherwise the Layer-I detection output is considered to be a false positive and the test (eventually retrained) restarts to assess forthcoming data. Using Adaptive SVM [12] as its base classifier for prediction in Layer-I, extends HLFR to a concept drift-agnostic classifier.

Experiments on synthetic dataset and real world applications demonstrate that the proposed HLFR framework outperforms state-of-the-art methods by significantly reducing false positives and guaranteeing a low false negatives and detection delays. HLFR, is summarized in Algorithm 1.

Algorithm 1 Hierarchical Linear Four Rate (HLFR)

```
Require: Data $\{\mathbf{X}_t, y_t\}_{t=0}^{\infty}$ where $\mathbf{X}_t \in \mathbb{R}^d$, $y_t \in \{0, 1\}$.
Ensure: Concept drift time points $\{T_{cd}\}$.
 1: for $t = 1$ to $\infty$ do
 2:     Perform Layer-I hypothesis testing.
 3:     if (Layer-I detects potential drift point $T_{pot}$) then
 4:         Perform Layer-II hypothesis testing on $T_{pot}$
 5:         if (Layer-II confirms the potentiality of $T_{pot}$) then
 6:             $\{T_{cd}\} \leftarrow T_{pot}$
 7:         else
 8:             Discard $T_{pot}$; Reconfigure and restart Layer-I
 9:         end if
10:     end if
11: end for
```

II-A. Layer-I Hypothesis Testing

Layer-I uses a drift detection agorithm that work in an online fashion. Linear Four Rates (LFR) is used as the Layer-I

[2]This paper only considers binary classification as majority of previous works. We leave investigations on multiclass classification as future works.

[3]Note that the Layer-I and Layer-II test can also be executed separately and independently, *i.e.*, merely operating the Layer-I test, the HLFR framework reduces to our previously proposed LFR [7]; merely operating the Layer-II test may also achieve satisfactory results at the cost of expensive computation.

test, as it has been experimentally proven that LFR outperforms benchmarks in terms of recall , accuracy and detection delay in majority of the cases.

The underlying concept for LFR strategy is straightforward: under a stable concept (*i.e.*, $P(\mathbf{X}_t, y_t)$ remains unchanged), P_{tpr},P_{tnr},P_{ppv},P_{npv} remains the same. Thus, a significant change of any $P_\star$ ($\star \in tpr, tnr, ppv, npv$) implies a change in underlying joint distribution $P(\mathbf{X}_t, y_t)$, or concept. More specifically, at each time instant t, LFR conducts statistical tests with the following null and alternative hypothesis:

$$\begin{aligned} H_0 &: \forall_\star, P(\hat{P}_\star^{(t-1)}) = P(\hat{P}_\star^{(t)}) \\ H_A &: \exists_\star, P(\hat{P}_\star^{(t-1)}) \neq P(\hat{P}_\star^{(t)}) \\ &\star \in \{tpr, tnr, ppv, npv\} \end{aligned}$$

The concept is stable under H_0 and is considered to have potential drift if H_A holds. LFR modifies $P_\star^{(t)}$ with $R_\star^{(t)}$ as employed in [6], [13] (also see footnote 2). $R_\star^{(t)}$ is essentially a weighted linear combination of classifier's previous performance and current performance. Given the fact that $R_\star^{(t)}$ follows a weighted *i.i.d.* Bernoulli distribution (see proof in [7]), we are able to obtain the bound table BoundTable by Monte-Carlo simulations. Having computed the bounds, the framework considers that a concept drift is likely to occur and sets the warning signal ($warn.time \leftarrow t$), when any $R_\star^{(t)}$ crosses the corresponding warning bounds (warn.bd) for the first time. If any $R_\star^{(t)}$ reaches the corresponding detection bound (detect.bd), the concept drift is affirmed at ($detect.time \leftarrow t$). Interested readers can refer to [7] for more details.

Linear Four Rates Testing (Layer-I Hypothesis Testing) takes as input, data stream $\{(\mathbf{X}_t, y_t)\}_{t=1}^{\infty}$ where $\mathbf{X}_t \in \mathbb{R}^d$ and $y_t \in \{0, 1\}$ and a binary classifier $\hat{f}$ and returns potential concept drift time $\{T_{pot}\}$. Time decaying factors $\eta_\star$, warn significance level $\delta_\star$ and detect significance level $\epsilon_\star$ are user defined parameters. The test is initiated by setting $\hat{P}_\star^{(0)} \leftarrow 0.5$, $R_\star^{(0)} \leftarrow 0.5$, where $\star \in \{tpr, tnr, ppv, npv\}$ and confusion matrix $C^{(0)} \leftarrow [1, 1; 1, 1]$; For each incoming data point,

$$\hat{y}_t \leftarrow \hat{f}(\mathbf{X}_t) \quad \text{and} \quad C^{(t)}[\hat{y}_t][y_t] \leftarrow C^{(t-1)}[\hat{y}_t][y_t] + 1$$

$R_\star^{(t)}$ is updated using the following rule:

$$R_\star^{(t)} = \begin{cases} R_\star^{(t)} \leftarrow R_\star^{(t-1)} & (y_t, \hat{y}_t)) \overset{noimpact}{\rightarrow} \star \\ \eta_\star R_\star^{(t-1)} + (1-\eta_\star)\mathbf{1}_{\{y_t = \hat{y}_t\}} & else \end{cases}$$

$\hat{P}_\star^{(t)}$ is updated using the following rule:

$$\hat{P}_\star^{(t)} = \begin{cases} \dfrac{C^{(t)}[\mathbf{1}_{\{\star=tpr\}}, \mathbf{1}_{\{\star=tpr\}}]}{N_\star} & \star \in \{tpr, tnr\} \\ \dfrac{C^{(t)}[\mathbf{1}_{\{\star=ppv\}}, \mathbf{1}_{\{\star=ppv\}}]}{N_\star} & \star \notin \{tpr, tnr\} \end{cases}$$

The warning and detection is trigged under the following conditions. **If** (any $R_\star^{(t)}$ exceeds warn.bd$_\star$ & warn.time is NULL), **Then** warn.time $\leftarrow t$, **Else** (no $R_\star^{(t)}$ exceeds warn.bd$_\star$ & warn.time is not NULL) and warn.time $\leftarrow$ NULL.

The warning bound and detection bound are set as follows:

$$\text{warn.bd}_\star \leftarrow \text{BoundTable}(\hat{P}_\star^{(t)}, \eta_\star, \delta_\star, N_\star)$$

$$\text{detect.bd}_\star \leftarrow \text{BoundTable}(\hat{P}_\star^{(t)}, \eta_\star, \epsilon_\star, N_\star)$$

Once a detection is triggered, i.e, (any $R_\star^{(t)}$ exceeds detect.bd$_\star$) , detect.time $\leftarrow t$; relearn $\hat{f}$ using $\{(\mathbf{X}_t, y_t)\}_{t=\text{warn.time}}^{\text{detect.time}}$; reset $R_\star^{(t)}$, $\hat{P}_\star^{(t)}$, $C^{(t)}$; return $\{T_{pot}\} \leftarrow t$.

HLFR includes two modifications to LFR[7]. First, we update the time decaying factor $\eta_\star$ with[4]:

$$\eta_\star^{(t)} = \begin{cases} (\eta_\star^{(t-1)} - 1)e^{-(R_\star^{(t)} - R_\star^{(t-1)})} + 1 & R_\star^{(t)} \geq R_\star^{(t-1)} \\ (1 - \eta_\star^{(t-1)})e^{R_\star^{(t)} - R_\star^{(t-1)}} + (2\eta_\star^{(t-1)} - 1) & R_\star^{(t)} < R_\star^{(t-1)} \end{cases}$$

This adaptation is motivated from the adaptive signal processing domain [14]. The key idea is that once $R_\star$ is increased, the system tends to perform better with recent data, which suggests the feasibility of a larger time decaying factor, and vise versa. Moreover, we made the BoundTable have explicit mathematical expression of only a few parameters by surface fitting to bring the benefit of memory saving and increased table resolution. Numerous experiments (results not shown in this paper) validated the effectiveness of modifications, especially in a low memory environment. Unless otherwise specified, the LFR mentioned in this paper refers to the modified one.

II-B. Layer-II Hypothesis Testing

The second-level testing aims at validating detections raised by the Layer-I, as such it is activated only when the Layer-I detection occurs. In particular, we rely on the value T_{pot} provided by the Layer-I testing to partition the streaming observations into two subsequences (aiming at representing observations before and after the suspected drift instant T_{pot}) and then we elect to another statistical hypothesis test for comparing the inherent properties of these two subsequences, assessing possible variations in the joint distribution $P(\hat{f}(\mathbf{X}_t), y_t)$.

In this section, we present the employed permutation test procedure (see Algorithm 2). Permutation tests are theoretically well founded and does not require a priori information about the monitored process or nature of the drift [15]. The only thing we want to put emphasis on is that the selection of test statistic (or quantity) used at the Layer-II should be strictly related to that used at the Layer-I. To this end, we choose zero-one loss over the ordered train-test split, as our test statistic in the Layer-II testing. Zero-one loss is easy to calculate and also directly related to aforementioned four rates. The intuition behind this scheme is that if no concept drift has occurred, the prediction loss on the ordered split should not deviate too much from that of the shuffled splits, especially if the learning algorithm has algorithmic stability.

[4]Note that, this time-varying representation of $\eta_\star^{(t)}$ is hyperparameter free and self-bounded between $(2\eta_\star^{(t-1)} - 1, 1)$ with a "sigmoid" like curve.

Algorithm 2 Permutation Test (Layer-II)

```
Require: Potential drift time $T_{pot}$; Permutation window size $W$; Algorithm $\mathbf{A}$; Permutation number $P$; Significant rate $\eta$.
Ensure: Test decision (Is $T_{pot}$ $True$ or $False$?).
 1: $S_{ord} \leftarrow$ streaming segment before $T_{pot}$ of length $W$.
 2: $S'_{ord} \leftarrow$ streaming segment after $T_{pot}$ of length $W$.
 3: Train classifier $f_{ord}$ at $S_{ord}$ using $\mathbf{A}$.
 4: Test classifier $f_{ord}$ at $S'_{ord}$ to get the zero-one loss $\hat{E}_{ord}$.
 5: for $t = 1$ to $P$ do
 6:     $(S_i, S'_i) \leftarrow$ random split of $S_{org} \bigcup S'_{org}$.
 7:     Train classifier $f_i$ at $S_i$ using $\mathbf{A}$.
 8:     Test classifier $f_i$ at $S'_i$ to get the zero-one loss $\hat{E}_i$.
 9: end for
10: if $\frac{1+\sum_{i=1}^{P} \mathbf{1}[\hat{E}_{ord} \leq \hat{E}_i]}{1+P} \leq \eta$ then
11:     decision$\leftarrow True$
12: else
13:     decision$\leftarrow False$
14: end if
15: return decision
```

III. EXPERIMENTS

Four sets of experiments are presented to demonstrate the effectiveness and superiority of HLFR over baseline approaches. First, we demonstrate the benefits of a two-layered architecture for concept drift detection over single-layer-based approaches such as DDM [1], *etc.* Second, quantitative metrics and visual evaluation is presented to show HLFR's superior performance over standard DDM [1], EDDM [4], DDM-OCI [6], STEPD [5] as well as our recently proposed LFR [7]. Third, experiments of HLFR with various classifiers (soft-margin SVM classifier, k-nearest neighbor (KNN) and quadratic discriminant analysis (QDA)) is presented to demonstrate that the superiority of HLFR is independent of the choice of classifiers. Finally, in section III-D, we validate, via the application of spam email filtering, the effectiveness of the proposed HLFR on streaming data classification and the accuracy of its detected concept drift points.

III-A. Benefits of a Hierarchical Architecture

Before a thorough evaluation on HLFR framework, we first experimentally verified the benefits and flexibility of hierarchical architecture. Note that, this is not a trivial task, as the fundamental difference between HLFR framework and previous concept drift detection methods lies on the deployed hierarchical architecture. Besides, one may have the question: since the introduction of second-layer test aims to removing false positives as much as possible while remaining true positives, can a single-layer-based method, *e.g.*, DDM or LFR, plays the same role with precise parameter tuning? For example, by decreasing the warning and detection thresholds of DDM, we may expect more potential concept drift points.

This section gives an intuitive answer: although parameter tuning can control the number of detected potential drift points, thus provide a plausible way for removing false positives. It also removes true positives as well. Without loss of generality, we consider the benchmark USENET1 [16] dataset as a representative herein. This dataset simulates a stream of email messages from different topics that are sequentially presented to a user who then labels them as interesting or junk according to his/her personal interests. It consists of 5 time periods of 300 examples. At the end of each period, the user's interest in a topic alters in order to simulate the occurrence of concept drift.

We first consider four learning scenarios based on our previously proposed LFR framework, where the warning and detection significant levels are set to $\{\delta_\star = 0.01, \epsilon_\star = 0.001\}$, $\{\delta_\star = 0.01, \epsilon_\star = 0.0001\}$, $\{\delta_\star = 0.01, \epsilon_\star = 0.00005\}$ and $\{\delta_\star = 0.01, \epsilon_\star = 0.00001\}$, respectively. We then perform HLFR using the same parameter setting. The concept drift detection results are plotted in Fig. 2(a) and Fig. 2(c). Next, to demonstrate the flexibility of second-layer test, we consider the combination of benchmark DDM [1] and EDDM [4] with permutation test. The second and forth rows of Fig. 2(b) and Fig. 2(d) plot the detection results merely using DDM or EDDM with different parameter settings. While the first and third rows demonstrate the detection results for DDM or EDDM combined with permutation test under the same learning scenarios.

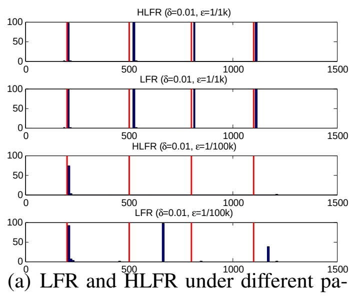
From Fig. 2, it can be concluded that the hierarchical architecture presents an intrinsic advantage over single-layer-based methods: firstly, it will not influence the performance of given single-layer-based methods if they already perform very well; besides, it can "correct" the errors made by the single-layer-based methods with the help of second-layer test; moreover, the second-layer test is a flexible module within hierarchical architecture, it can be logically combined with any other single-layer-based method in practical; finally, it is worth noting that the second-layer test is not limited to permutation test, we leave investigations of novel test as our future works.

III-B. Concept Drift Detection with HLFR

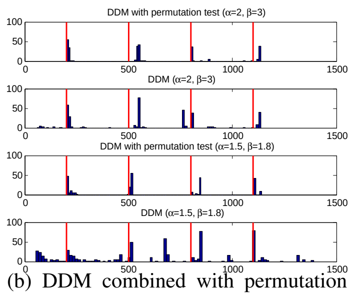
In this section, we compare the performance of our proposed HLFR framework in the concept drift detection setting with several benchmarks. Five algorithms are considered for evaluations: DDM [1], EDDM [4], DDM-OCI [6], STEPD [5][5] as well as our recently proposed LFR [7]. We use both synthetic data and drifts synthesized from real data, thus controlling ground truth drift points and allowing precise quantitative analysis. During comparison, we set the warning and detection thresholds of DDM (EDDM) to $\alpha = 3$ ($\alpha = 0.95$) and $\beta = 2$ ($\beta = 0.90$), the warning and detection significant levels of STEPD to $w = 0.05$, $d = 0.01$[6], and the parameter of DDM-OCI varies depending on data properties. The warning and detection significant levels of LFR, *i.e.*, $\delta_\star$ and $\epsilon_\star$, are set to

[5]We re-implemented STEPD using chi-square test with Yates's continuity correction, as it is equivalent to Fisher's exact test used in the original paper [17].

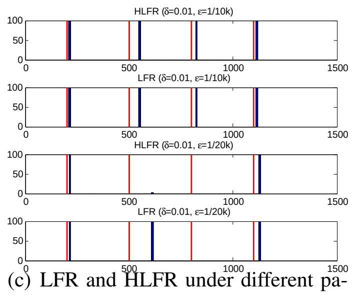
[6]These hyper parameters were not optimized. The parameter d in STEPD corresponds to a P-value and are thus set to a standard significance value. The parameters used for DDM and EDDM were taken as recommended by their authors.



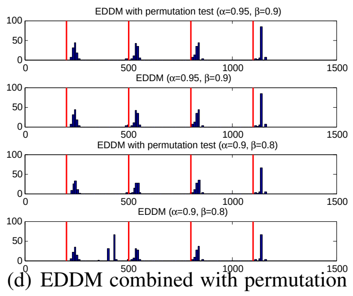
(a) LFR and HLFR under different parameter settings



(b) DDM combined with permutation test under different parameter settings



(c) LFR and HLFR under different parameter settings



(d) EDDM combined with permutation test under different parameter settings

Fig. 2: Comparison between the histogram of detected drift points over USENET1 dataset for LFR [7], DDM [1] and EDDM [4] combined with permutation test. The red columns denote the group truth of drift points, the blue columns represent the histogram of detected drift points generated from 100 Monte-carlo simulations.

0.01 and 0.0001, respectively. Besides, the significant rate η in HLFR is fixed to 0.05. Quantitative comparison are performed by evaluating detection quality. To this end, a True Positive (TP) detection is defined as a detection within a fixed delay range after the precise concept change time. A False Negative (FN) is defined as missing a detection within the delay range, and a False Positive (FP), as a detection outside this range or an extra detection in the range. The detection quality is then measured both by the $\mathbf{Recall} = TP/(TP + FN)$ and $\mathbf{Precision} = TP/(TP + FP)$ of the detector.

In the following drift detection tasks the base classifier was soft margin SVM with linear kernel (except for USENET1 [16], where the soft margin SVM with RBF kernel ($\sigma = 1$) is selected as the base classifier). The first stream, denoted "SEA" [1], represents abrupt drift with label noise. This dataset has 60000 examples, 3 attributes. Attributes are numeric between 0 and 10, only two are relevant. There are 4 concepts, 15000 examples each, with different thresholds for the concept function, which is if the sum of two relevant features is larger than threshold then example label is 0. Threshold values are 8, 9, 7 and 9.5. The second stream, denoted "Checkerboard" [18], presents a more challenging concept drift with label noise, where the examples are sampled uniformly from the unit square and the labels are set by a checkerboard with 0.2 tile width. At each concept drift, the checkerboard is rotated in an angle of $\phi/8$ radians. The third stream, the most challenging synthetic dataset, denoted "Rotating hyperplane" , constitutes 60000 examples and 5 uniformly-spaced abrupt concept drift points. Within any of the two adjacent abrupt drift points, there are 10000 examples demonstrating a slow gradual concept drift which is specified by the (k,t) pairs, where k denotes the total number of dimensions whose weights are changing and t measures the magnitude of such change in attributes. In our experiments, the (k,t) pairs are set to (2,0.1), (2,0.5), (2,1.0), (5,0.1), (5,0.5) and (5,1.0) successively. The last stream, USENET1 [16], represents drifts synthesized from real data. Table I summarized the data properties and drift types for each stream. A yes indicates that the stream data have corresponding data property or concept drift type, and vice versa. Clearly, the selected datasets span the breadth of concept drift types.

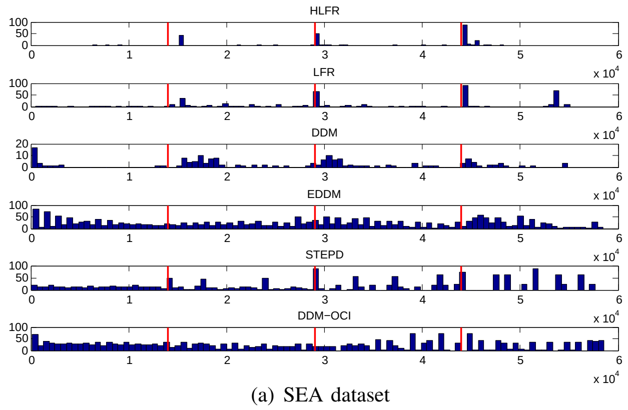
Table I: Summary of properties of selected datasets

Data property	SEA	Checkerboard	Hyperplane	USENST1
high dimensional	*yes*	*yes*	*yes*	*yes*
imbalance	*no*	*yes*	*yes*	*no*
recurrent	*no*	*no*	*yes*	*yes*
abrupt	*yes*	*yes*	*no*	*yes*
gradual	*no*	*no*	*yes*	*no*

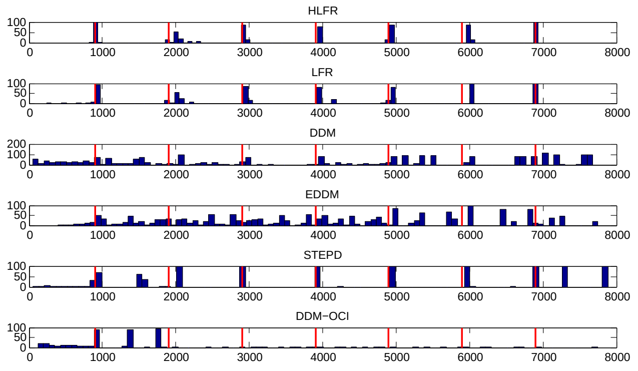
Each stream was independently generated 100 times, and $P = 1000$ reshuffling splits were used in HLFR. We summarized the detected concept drift detection points for each method over these 100 independent trails. Fig. 3 demonstrates the detection comparison results. As can be seen, HLFR and LFR dominate other four approaches in terms of early detections and fewer false or missing detections. Besides, compared with LFR, HLFR can effectively remove the false positives. Quantitative evaluations for $\mathbf{Precision}$ and $\mathbf{Recall}$ are presented in Fig. 4: rows correspond to performance measurements and columns to different datasets. The detection $\mathbf{Precision}$ is significantly improved with HLFR. Besides, the $\mathbf{Recall}$ of HLFR and LFR is similar (except for Rotating hyperplane dataset). This is not surprising, as the purpose of Layer-II test serves to confirm or deny the potentiality of first layer detection results, it cannot compensate for the errors of missing a detection made by Layer-I test. The relatively lower $\mathbf{Recall}$ can also be anticipated because the Layer-II test is conservative, $i.e.$, it may deny true positives although the probability is very low. STEPD seems to provide much better $\mathbf{Recall}$ on SEA and Rotating hyperplane datasets, the results is meaningless in practical as it generate much more false alarms (see the fifth row of Fig. 3(a) and Fig. 3(c)). Moreover, the detection $\mathbf{Precision}$ of STEPD on these two datasets are consistently less than 0.15, which further verified the invalidity of its high $\mathbf{Recall}$ values. Table II summarized the detection delays for all competing algorithms. Again, quantitative evaluations corroborate the qualitative observations.

Table II: Detection delay comparison for all competing algorithms

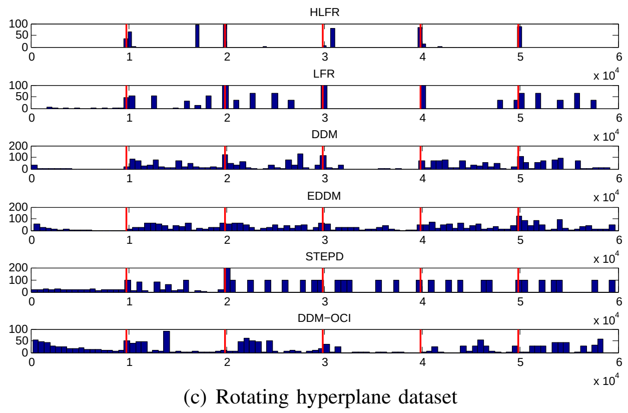
Algorithms	SEA	Checkerboard	Hyperplane	USENST1
HLFR	482(502)	**55**(37)	**120**(85)	**17**(7)
LFR	**458**(486)	56(36)	127(112)	17(7)
DDM	1209(450)	69(56)	125(128)	26(17)
EDDM	939(550)	93(50)	166(121)	36(8)
DDM-OCI	844(461)	58(40)	198(112)	26(17)
STEPD	463(423)	57(58)	140(118)	19(7)



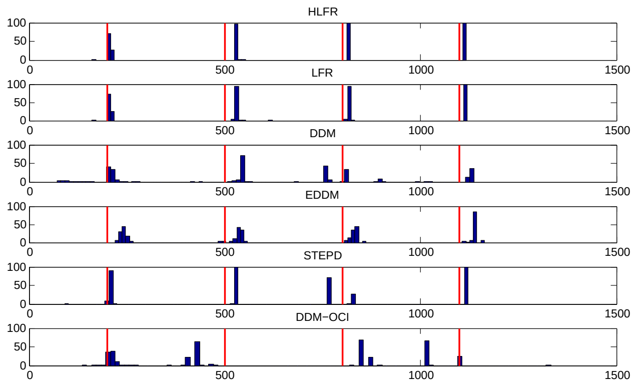
(a) SEA dataset



(b) Checkerboard dataset

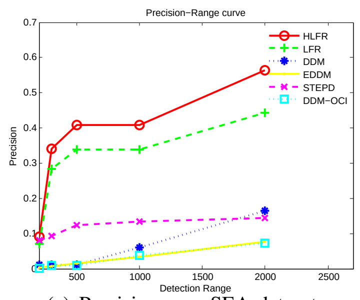


(c) Rotating hyperplane dataset

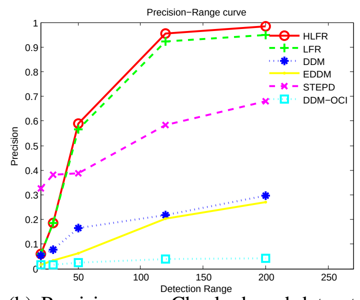


(d) USENET1 dataset

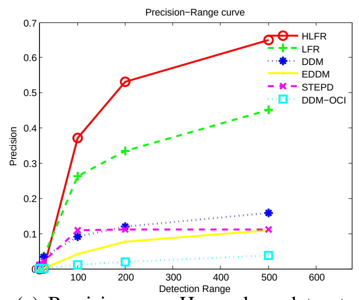
Fig. 3: Comparison between the histogram of detected drift points over (a) SEA dataset; (b) Checkerboard dataset; (c) Rotating hyperplane dataset and (d) USENET1 dataset. The red columns denote the group truth of drift points, the blue columns represent the histogram of detected drift points generated from 100 Monte-carlo simulations.



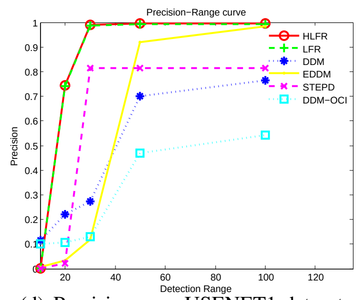
(a) Precision over SEA dataset



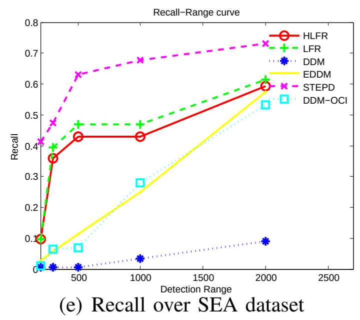
(b) Precision over Checkerboard dataset



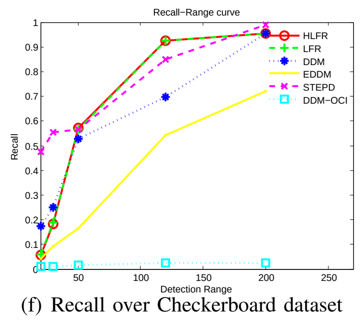
(c) Precision over Hyperplane dataset



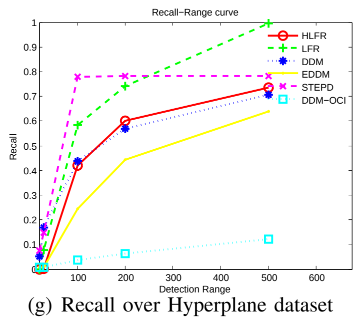
(d) Precision over USENET1 dataset



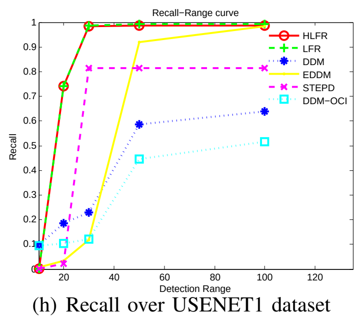
(e) Recall over SEA dataset



(f) Recall over Checkerboard dataset



(g) Recall over Hyperplane dataset



(h) Recall over USENET1 dataset

Fig. 4: Summary of **Precision** and **Recall** over SEA, Checkerboard, Rotating hyperplane and USENET1 datasets. The X-axis in each figure represents the pre-defined detection delay range, whereas the Y-axis denotes the corresponding **Precision** and **Recall** values. For a specific delay range, a higher **Precision** or **Recall** value suggests better performance.

III-C. HLFR's Performance is Independent of Classifier Choice

In this section, we show that the superiority of HLFR is independent of classifier. To this end, instead of using soft-margin SVM, two different other classifiers, *i.e.*, k-nearest neighbor (KNN) classifier and quadratic discriminant analysis (QDA), are applied separately on all the competing algorithms. Fig. 5 shows the concept drift detection results over USENET1 dataset and Checkerboard datasets using these two classifiers, which, obviously, coincide with the simulation results in Section III-B. HLFR can consistently achieve the best performance over all its counterparts. Note that, although almost all the methods perform unpleasant on Checkerboard dataset using QDA classifier, only HLFR can provide reasonable detection results on the third, fifth, sixth and seventh drifts.

III-D. Classification With Concept Drift Using HLFR

Finally, we present a real application on the problem of classification to streaming data with concept drifts, aiming to validate the effectiveness and rationality of the proposed HLFR on concept drift detection as well as its potentiality on streaming data classification. To this end, a case study is performed on a representative real-world concept drifting dataset from the email domain. Spam filtering dataset [19], consisting of 9324 instances and 500 attributes, is selected herein. The spam ratio is approximately 20%. It has been demonstrated that spam filtering dataset contains natural concept drifts [19], [20]. A soft-margin SVM classifier is generated from the training dataset and evaluated (also adapted) on the test dataset. The DDM-OCI performance has been omitted only for the reason that it fails to detect any "reasonable" concept drift points.

On Parameter Tuning and Experimental Setting. A common phenomenon for classification of real world streaming data with concept drifts and temporal dependency is that "*the more random change alarms the classifier fires, the better the accuracy* [21]". Thus, to provide a fair comparison, the parameters of all competing algorithms are tuned to detect similar number of concept drifts (except for LFR [7], where slightly more potential drift points are allowed). Table III summarized the key parameters regarding significant levels (or thresholds) of different algorithms. It is also of importance to mention that we made an extensive search to look for an appropriate partition of training set and testing set upon two criterions: 1) the training set is sufficient to achieve "significant" classification performance on both majority and minority classes; and 2) there is no strong autocorrelations in the classification residual sequence of training set. With these two considerations, the length of training set is set to 600.

Fig. 6 plots the concept drift detection results over training set. Before evaluating detection results, we recommend interested readers to refer to Fig. 6 of [19], in which the k-means and expectation maximization (EM) clustering algorithms have been applied to the conceptual vectors of spam filtering dataset. According to [19], there are three dominating clusters (*i.e.*, concepts) distributed in different time periods and concept

Table III: Parameter settings in spam email filtering.

Algorithms	Parameter settings on significant levels (or thresholds)
HLFR	$\delta_\star = 0.01$, $\epsilon_\star = 0.00001$, $\eta = 0.01$
LFR	$\delta_\star = 0.01$, $\epsilon_\star = 0.00001$
DDM	$\alpha = 3$, $\beta = 2.5$
EDDM	$\alpha = 0.95$, $\beta = 0.90$
STEPD	$w = 0.005$, $d = 0.0003$

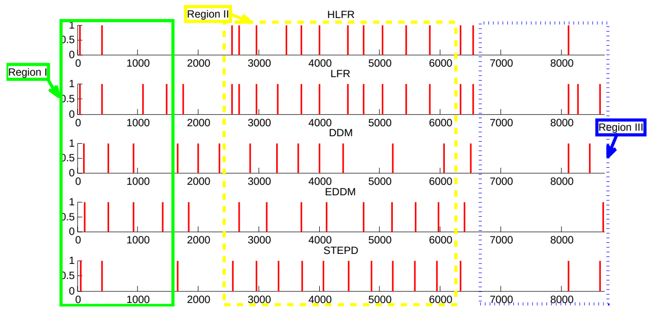
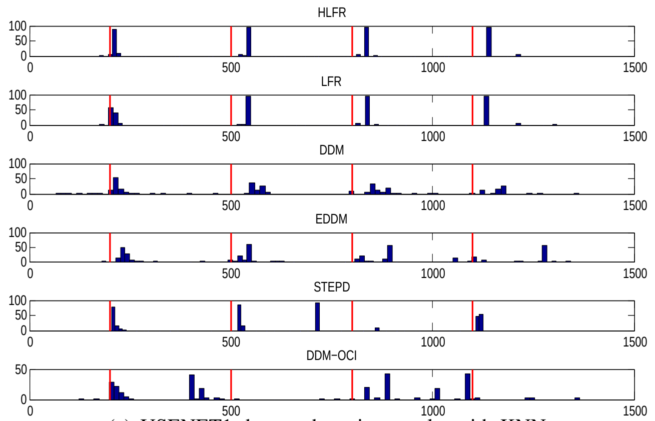


Fig. 6: Concept drift detection results over all competing algorithms.

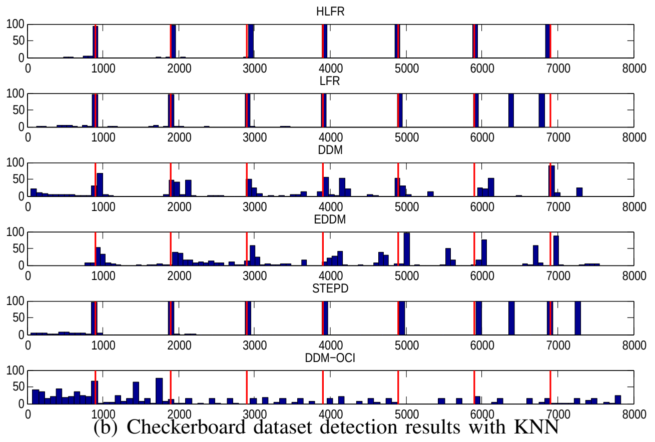
drifts occurred approximately in the neighbors of point 200 in Region I, point 8000 in Region III, the ending location (point 1800) of Region I as well as the start and ending locations (point 2300 and 6200) of Region II. Besides, there are many abrupt drifts in the Region II. A possible reason for these abrupt and frequent drifts may be batches of outliers or noisy messages. Obviously, the detection results of HLFR best matches these descriptions, except that it missed a potential drift point around point 1800. LFR detected this point, but it also feedbacks some false positives. Other methods, like DDM or EDDM, not only missing obvious drift points, but also reporting unreasonable drift locations in Region I or Region III.

To further bridge the connections between our detection results and clustering results in [19], a recently developed measurement - Kappa Plus Statistic (KPS) [22], [23] - have been proposed. KPS, defined as $\kappa^+ = \frac{p_0 - p'_e}{1-p'_e}$, aims to evaluate data stream classifier performance taken into account temporal dependence as well as the effectiveness (or rationality) of classifier adaptation, where p_0 is the classifier's prequential accuracy and p'_e is the accuracy of No-Change classifier. We segment the training set to approximately 30 periods. The KPS prequential representation over these periods is shown in Fig. 7. As can be seen, the HLFR adaptation is most effective in period 1-5 but suffer from a performance drop on period 6-10. These observations coincide the our detection results, as HLFR accurately detected the first drift point with no false positives in Region I, but missed a target in Region II.

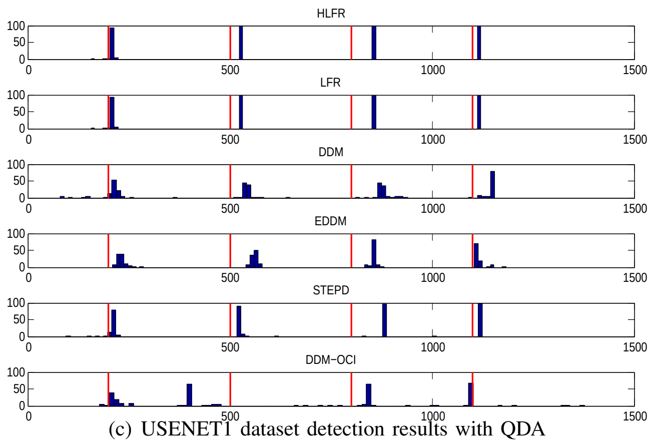
Regarding the problem of streaming data classification. Several different quantitative measurements are used for a thorough evaluation. First, while overall accuracy (OAC) is an important and commonly used metric, it is not an adequate evaluation measure for streaming data classification (especially for imbalanced data). Therefore, we also include the F-measure and minority class G-mean value. All metrics are obtained at each time step, creating a time-series representation (just like learning curves in the general field of adaptive learning



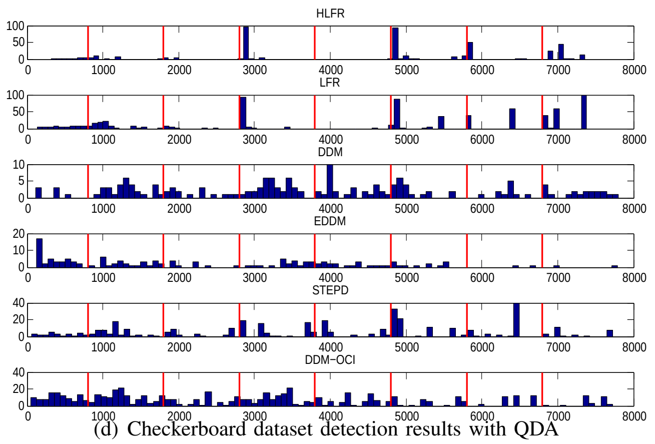
(a) USENET1 dataset detection results with KNN



(b) Checkerboard dataset detection results with KNN



(c) USENET1 dataset detection results with QDA



(d) Checkerboard dataset detection results with QDA

Fig. 5: Comparison between the concept drift detection results over USENET1 and Checkerboard datasets using KNN or QDA classifier. The red columns denote the group truth of drift points, the blue columns represent the histogram of detected drift points generated from 100 Monte-carlo simulations.

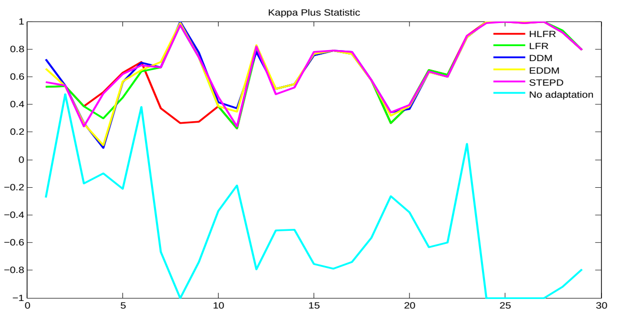


Fig. 7: Kappa Plus Statistic prequential representations.

systems [24]). Fig. 8 plots the time series representations of OAC, F-measure and G-mean in the learning scenario. It is easy to summarize some key observations from these figures:
1) There are severe concept drifts in the given data, as the performance of a non-adaptive classifier deteriorate significantly as time evolves.
2) HLFR typically provides a significant improvement in F-measure and G-mean compared to its DDM, EDDM and LFR counterparts, while maintaining good OAC.
3) STEPD seems to demonstrate the best overall classification performance in this case, but HLFR provides more accurate (or rational) concept drift detections which coincide with cluster assignments results in [19].

IV. CONCLUSIONS

This paper presents a concept drift detection framework (HLFR) that dynamically adapts its parameters based on the concept drifts encountered to reduce false alarms. Using Adaptive SVM [12] as its base classifier for prediction, extends HLFR to a concept drift-agnostic classifier. The performance of HLFR is compared to benchmark approaches using both simulated and real-world datasets spanning the gamut of concept drift types. Results demonstrate that HLFR significantly outperforms benchmark approaches in terms of accuracy, G-mean, recall, delay in detection of concept drift across the various datasets.

V. REFERENCES

[1] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, “Learning with drift detection,” in *Brazilian Symposium on Artificial Intelligence*. Springer, 2004, pp. 286–295.

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

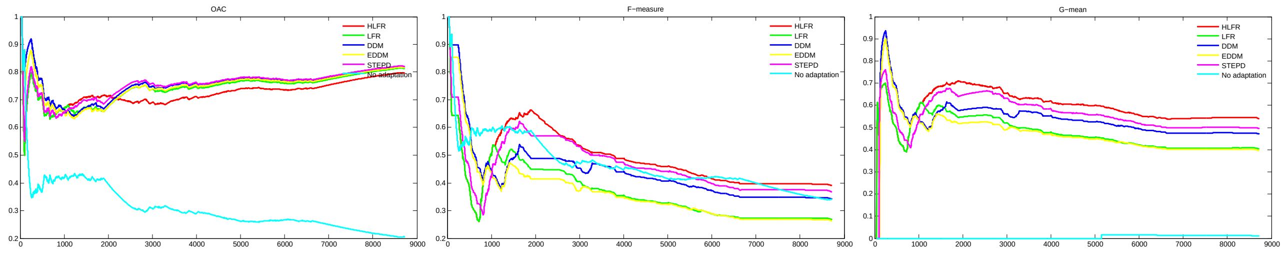


Fig. 8: The time series representations for all competing algorithms over OAC, F-measure and G-mean.

[3] M. Harel, S. Mannor, R. El-Yaniv, and K. Crammer, "Concept drift detection through resampling." in *ICML*, 2014, pp. 1009–1017.

[4] M. Baena-Garcıa, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno, "Early drift detection method," in *Fourth international workshop on knowledge discovery from data streams*, vol. 6, 2006, pp. 77–86.

[5] K. Nishida and K. Yamauchi, "Detecting concept drift using statistical testing," in *International conference on discovery science*. Springer, 2007, pp. 264–269.

[6] S. Wang, L. L. Minku, D. Ghezzi, D. Caltabiano, P. Tino, and X. Yao, "Concept drift detection for online class imbalance learning," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 2013, pp. 1–10.

[7] H. Wang and Z. Abraham, "Concept drift detection for streaming data," in *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2015, pp. 1–9.

[8] V. Vapnik, "Principles of risk minimization for learning theory," in *NIPS*, 1991, pp. 831–838.

[9] L. I. Kuncheva and C. O. Plumpton, "Adaptive learning rate for online linear discriminant classifiers," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 2008, pp. 510–519.

[10] R. Klinkenberg and T. Joachims, "Detecting concept drift with support vector machines." in *ICML*, 2000, pp. 487–494.

[11] O. Bousquet and A. Elisseeff, "Stability and generalization," *Journal of Machine Learning Research*, vol. 2, no. Mar, pp. 499–526, 2002.

[12] J. Yang, R. Yan, and A. G. Hauptmann, "Adapting svm classifiers to data with shifted distributions," in *Seventh IEEE International Conference on Data Mining Workshops (ICDMW 2007)*. IEEE, 2007, pp. 69–76.

[13] S. Wang, L. L. Minku, and X. Yao, "A learning framework for online class imbalance learning," in *Computational Intelligence and Ensemble Learning (CIEL), 2013 IEEE Symposium on*. IEEE, 2013, pp. 36–45.

[14] S. S. Haykin, *Adaptive filter theory*. Pearson Education India, 2008.

[15] P. Good, *Permutation tests: a practical guide to resampling methods for testing hypotheses*. Springer Science & Business Media, 2013.

[16] I. Katakis, G. Tsoumakas, and I. P. Vlahavas, "An ensemble of classifiers for coping with recurring contexts in data streams." in *ECAI*, 2008, pp. 763–764.

[17] E. L. Lehmann and J. P. Romano, *Testing statistical hypotheses*. Springer Science & Business Media, 2006.

[18] R. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *IEEE Transactions on Neural Networks*, vol. 22, no. 10, pp. 1517–1531, 2011.

[19] I. Katakis, G. Tsoumakas, and I. Vlahavas, "Tracking recurring contexts using ensemble classifiers: an application to email filtering," *Knowledge and Information Systems*, vol. 22, no. 3, pp. 371–391, 2010.

[20] ——, "Dynamic feature space and incremental feature selection for the classification of textual data streams," *Knowledge Discovery from Data Streams*, pp. 107–116, 2006.

[21] I. Zliobaite, "How good is the electricity benchmark for evaluating concept drift adaptation," *arXiv preprint arXiv:1301.3524*, 2013.

[22] A. Bifet, J. Read, I. Žliobaitė, B. Pfahringer, and G. Holmes, "Pitfalls in benchmarking data stream classification and how to avoid them," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2013, pp. 465–479.

[23] I. Žliobaitė, A. Bifet, J. Read, B. Pfahringer, and G. Holmes, "Evaluation methods and decision theory for classification of streaming data with temporal dependence," *Machine Learning*, vol. 98, no. 3, pp. 455–482, 2015.

[24] S. S. Haykin, *Neural networks and learning machines*. Pearson Upper Saddle River, NJ, USA:, 2009, vol. 3.