Hierarchical reduced-space drift detection framework for multivariate supervised data streams

Zhang, Shuyi; Tino, Peter; Yao, Xin

DOI:
10.1109/TKDE.2021.3111756

Document Version
Early version, also known as pre-print

Citation for published version (Harvard):
Zhang, S, Tino, P & Yao, X 2023, 'Hierarchical reduced-space drift detection framework for multivariate supervised data streams', *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 3, pp. 2628-2640. https://doi.org/10.1109/TKDE.2021.3111756

Hierarchical Reduced-space Drift Detection Framework for Multivariate Supervised Data Streams

Journal:	*Transactions on Knowledge and Data Engineering*
Manuscript ID	TKDE-2020-08-0877
Manuscript Type:	Regular
Keywords:	Concept drift, Drift detection, Data stream mining, Online learning

Hierarchical Reduced-space Drift Detection Framework for Multivariate Supervised Data Streams

Shuyi Zhang, Peter Tino, *Fellow, IEEE* and Xin Yao, *Fellow, IEEE*

Abstract—In a streaming environment, the characteristics of the data themselves and their relationship with the labels are likely to experience changes as time goes on. Most drift detection methods for supervised data streams are performance-based, that is, they detect changes only after the classification accuracy deteriorates. This may not be sufficient in many application areas where the reason behind a drift is also important. Another category of drift detectors are data distribution-based detectors. Although they can detect some drifts within the input space, changes affecting only the labelling mechanism cannot be identified. Furthermore, little work is available on drift detection for high-dimensional supervised data streams. In this paper we propose an advanced **H**ierarchical **R**educed-space Drift **D**etection Framework for **S**upervised Data Streams (HRDS) which captures drifts regardless of their effects on classification performance. This framework suggests monitoring both marginal and class-conditional distributions within a lower-dimensional space specifically relevant to the assigned classification task. Experimental comparisons have demonstrated that the proposed HRDS not only achieves high-quality performance on high-dimensional data streams, but also outperforms its competitors in terms of detection recall, precision and F-measure across a wide range of different concept drift types including subtle drifts.

Index Terms—Concept drift, drift detection, data stream mining, online learning

1 Introduction

In real-world applications such as weather prediction, industrial quality control and spam or fraud detection, data often arrives in the form of a stream. Data that continuously flow can be generated by sources that change in time. When there is a change in the underlying data distribution and/or its relationship with labels, such phenomenon is called *concept drift* [1]. This problem has received growing attention not only because it may greatly affect the reliability of real time machine learning systems, but also because it is useful to find out the reason and the nature of the changes [2]. One way of categorizing drifts is by its influence on the target concept. Changes in the posterior class probabilities $P(Y|\mathbf{X})$ are called real drifts, whereas changes affecting the input distribution $P(\mathbf{X})$ only are called virtual drifts [3].

Various change detection tests (CDTs) have been proposed to explicitly mark out the drifts [4], [5]. Unfortunately, current detection methods cannot well address both types of drifts simultaneously and consistently. Most of them monitor over time either some classification-performance-related indicators [6], [7], [8], [9], [10], [11], [12], [13], or some data-distribution-related characteristics [14], [15], [16], [17], [18], [19], [20]. Existing detectors for supervised data streams primarily belong to the earlier category [4], [21]. They concentrate on addressing real drifts which lead to a decline in classification performance only. Some popular algorithms within this category are drift detection method (DDM) [6] and early drift detection method (EDDM) [7]. The former detects abrupt drift by applying statistical test on the false classification rate directly, whereas the latter monitors the distance between consecutive classification errors. Linear four rate (LFR) [10] is another detector which monitors all components of the confusion matrix. Although these detectors can be used in conjunction with any classifier since they utilize only the error stream, their detection performance is still dependent on the chosen base classifier [22]. Besides, they fail to detect drifts not deteriorating the classification performance.

In addition to the above, many detectors within the second category monitoring the underlying data features have also been proposed. Cumulative sum (CUSUM) control chart [14] monitors the cumulative sum of deviations for drift detection, and intersection of confidence intervals (ICI) CDT [17] carefully designs mean and variance-related features that follow a Gaussian distribution. For multivariate data streams, non-parametric distribution-based detectors

Manuscript received MM DD,YYYY ; revised MM DD,YYYY.

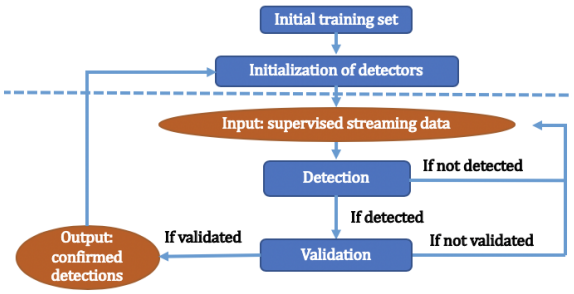


Figure 1: General framework of HCDT

either monitor the estimated empirical density of two windows [15], [20], [23] or adopt univariate statistical tests on each individual features for detection [17], [24]. These approaches tend to be problematic for higher dimensional data streams [9], [10]. Besides, while these detectors can be directly applied to supervised data streams, they do not consider any class information and thus cannot detect drift affecting the data labelling mechanisms only (e.g., a class swap) [25]. Since they monitor the overall input space, they tend to be insensitive to drifts affecting only a subregion of the input space (e.g., a single class drift).

Based on the vast scope of individual CDTs existing in the literature, more consolidated frameworks have been developed recently. Hierarchical change detection test (HCDT) presented in Fig. 1 is a general two-layer detection-and-verify framework [26]. HCDT incorporates in Layer-I a simple non-parametric online detector such as the CUSUM CDT or ICI-based CDT, and in Layer-II an offline two-sample test such as the Hotelling T2 test [27]. Once a potential drift is reported in Layer-I, the Layer-II test is activated to compare the training set with the most recent set so as to confirm (or deny) the validity of the suspected drift. HCDT has been shown to achieve more advantageous false positive rate (FPR) versus detection delay (DD) trade-off than its single CDT counterpart, but it has only been tested on non-labelled scalar data [26]. Direct application of this framework to multivariate supervised data streams still suffers from the aforementioned deficiencies of distribution-based detectors. Inspired by this framework, another hierarchical framework named HLFR for supervised data streams was proposed [28]. HLFR incorporates LFR as the base detector in Layer-I and a permutation test [29] in Layer-II. However, HLFR is purely classification performance-based, therefore, it cannot detect real and virtual drifts simultaneously.

Concept drifts can be incurred by many causes and they present differently in different periods [6]. Therefore, it is important to be aware of all drifts regardless of their effects on classification, especially in areas such as condition monitoring, adversarial attack detection and strategic planning. Furthermore, there is little work on drift detection on high-dimensional data streams. In this paper, we adopt the hierarchical structure and propose a new detection framework, HRDS (**H**ierarchical **R**educed-space **D**etection framework for multivariate **S**upervised data streams), to detect both real and virtual drift accurately and efficiently for multi-dimensional data streams. The key idea is to leverage the knowledge from supervised information to discover changes that may not be detected by the existing detection methods. To achieve this goal, first a lower-dimensional feature space for the given classification task is explicitly constructed using the training data. All incoming data are first projected to this space. Next we monitor not only the marginal distribution of the data stream, but also each individual class-conditional distribution. Finally, a novel method to reconfigure more informative retraining datasets after each detection is presented. HRDS can be used in conjunction with any classifiers and its performance is independent of the choice of classifier. The contributions of our work include:

1) This is the first hierarchical detection framework proposed for supervised data stream that detects both real and virtual drifts.
2) Compared with the existing HCDT framework, the data-distribution based HRDS is more accurate and efficient in terms of a high number of true detections, while maintaining a low number of false alarms, when operating on higher-dimensional data streams.
3) For both real and virtual drifts, HRDS performs no worse, and in many cases better, than state-of-the-art detection algorithms, whether they are performance-based or distribution-based, in terms of more true detections and lower false alarms within any specified acceptable detection delay range.

Detecting the concept drifts and then adapting a learner to them are two different mechanisms. From the practical point of view, an accurate detector very important in maintaining good classification performance in the long run. The focus of this paper is to detect drifts. How to build an appropriate classifier for a specific data stream is beyond the scope of this study.

The rest of the paper is organized as follows. **Section 2** formulates the problem of concept drift. **Section 3** explains each component of the proposed HRDS framework in detail. In **Section 4**, four sets of experiments are carried out on both synthetic and real-world data streams to demonstrate the superiority of HRDS in comparison with some state-of-the-art detectors, including distribution-based ones and performance-based ones. **Section 5** concludes the paper and points out potential future extensions of this work.

2 Terminology and Problem Formulation

In a streaming environment, a supervised data stream to be inspected for change is formed by observations $\{(\mathbf{X}_t, y_t), t \in \mathbb{Z}^+\}$. The generation process of the observations at time t can be denoted by the joint distribution $P_t(\mathbf{X}, Y)$. $\mathbf{X}_t \in \mathbb{R}^d$ represents the d-dimensional feature vector of the t^{th} observation and y_t is its class label. $y_t \in \{0, 1, ..., Q\}$ where $Q + 1$ is the number of available classes. For binary classification, $y_t \in \{0, 1\}$. A concept drift is said to occur when there is a change in the joint probability $P_t(\mathbf{X}, Y)$ [21]. The alternative hypotheses for assessing an abrupt change are formulated as follows: $(\mathbf{X}_t, y_t) \sim P_{t_0}(\mathbf{X}, Y)$, for $t < T^*$ and $(\mathbf{X}_t, y_t) \sim P_{t_1}(\mathbf{X}, Y)$, for $t \geq T^*$, where $t_0 < T^* \leq t_1$, $P_{t_1}(\mathbf{X}, Y) \neq P_{t_0}(\mathbf{X}, Y)$ and T^* is an unknown change point.

The joint probability $P_t(\mathbf{X}, Y)$ can be written as

$$P_t(\mathbf{X}, Y) = P_t(Y|\mathbf{X}) \cdot P_t(\mathbf{X}) \quad (1)$$

where $P_t(\mathbf{X})$ can be obtained through marginalization

$$P_t(\mathbf{X}) = \sum_{q=0}^{Q} P_t(Y = q) \cdot P_t(\mathbf{X}|Y = q) \quad (2)$$

Based on the probabilistic definition of a concept drift and the above decomposition, it is not difficult to tell that the change can manifest itself in different forms corresponding to the different components of the joint probability [30], [31]. Assuming $P(Y)$ is stationary over time, drift can occur in: 1) the marginal distribution over covariates $P(\mathbf{X})$; 2) the posterior class probability or classification concept $P(Y|\mathbf{X})$; 3) the class-conditional distributions $P(\mathbf{X}|Y)$.

Most existing work tackling drift in supervised data streams focused on the second type of drift or real concept drift, since it is considered to be most detrimental to classification accuracy. However, we consider the detection of all types of drift to be equally important for the following reasons. Firstly, even when a so-called virtual drift takes place and classification accuracy is not negatively affected, the optimal decision boundary is often likely to change. Retraining the classifiers can improve classification performance. Secondly, detection of such drifts provides insight into the underlying data streams, which can help understanding of the streaming. This information may also be beneficial when there is a pattern in a series of multiple drifts. The systematic study [4] has supported the view that all types of change are equally important, but there is a lack of research effort in the investigation of drifts not affecting classification accuracy.

Therefore, in this paper we do not explicitly distinguish between real and virtual drifts. We present a framework aiming to detect all types of drifts regardless of whether they affect classification or not. Then, practitioners can decide whether it is worth retraining the current classification model based on the specific application scenario.

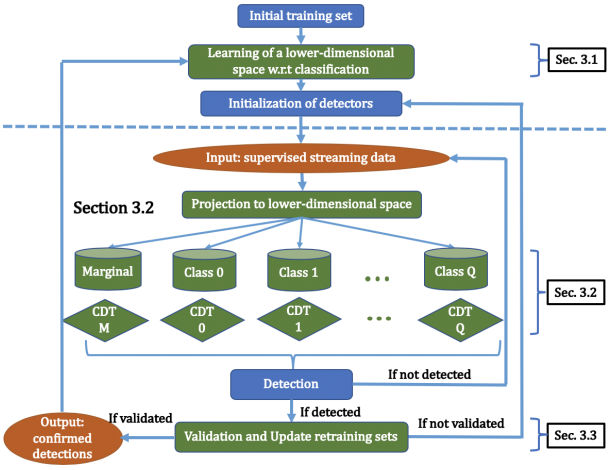


Figure 2: General framework of HRDS. The detailed descriptions for each novel component are provided in sub-sections 3.1, 3.2 and 3.3

3 Hierarchical Reduced-space Drift Detection Framework for Multivariate Supervised Data Stream

In this section we describe a novel change detection test framework named HRDS (**H**ierarchical **R**educed-space **D**etection framework for multivariate **S**upervised data stream) aiming to answer the following research questions.

1) How to detect both real and virtual drifts in supervised data streams regardless of their effect on classification performance?
2) How to improve the efficiency of data distribution-based detector for high-dimensional data stream?
3) How to improve detection performance to achieve high true detections and low false alarms within a specified delay range for all types of drifts even when the magnitude of drift is small?

HRDS adopts the hierarchical structure introduced in [26] but with three major novel components explained in the following sub-sections. The general outline of HRDS is presented in Fig. 2. This framework has a high degree of flexibility and may be customized by using different change-detection and validation techniques. The algorithmic version of HRDS is presented in Algorithm 1. Although we provide one possible realization for a binary classification problem as an illustrative example in this paper, it is worth noting that the general framework of HRDS is also suitable for multi-class data streams.

Algorithm 1: General framework of HRDS

1 **Input**: *initial training sets TS^M for marginal detector and $TS^0, TS^1, TS^2, ...$ for each class-conditional detector*
2 **Output**: *confirmed detections*
3 Find the lower-dimensional feature space $\mathcal{S}$;
4 Initialize the marginal and class-conditional detectors;
5 **while** *there is incoming data* **do**
6 Project data onto $\mathcal{S}$;
7 Perform concept drift detection within $\mathcal{S}$;
8 **if** *change detected in any of the CDTs at $\widehat{T}$* **then**
9 Estimate the potential drift starting point T_{ref};
10 Activate the validation layer on the respective stream;
11 **if** *change is validated* **then**
12 Record $\widehat{T}$ as a confirmed detection;
13 Define TS_C^M as all instances in $\{\mathbf{x}(t), t \in [T_{ref}, ..., \widehat{T}]\}$;
14 Update training sets $TS^M, TS^0, TS^1, ...$ accordingly and continue from line 3.
15 Output the confirmed changes.

3.1 Learning of a lower-dimensional subspace

The aim of this step is to identify a lower-dimensional space $\mathcal{S}$ that contains the most relevant information for the given classification task, that is, identify a feature subspace spanned by the training samples (line 3, Algorithm 1) so that the incoming multivariate data samples are projected onto this space (line 6, Algorithm 1). Then instead of monitoring the original input samples, the detection is carried out within this reduced feature space for the particular classification task. Comparing with the existing HCDT without this step, HRDS inherently reduces the possibility of false alarms as well as the computational burden because there are fewer dimensions to monitor. Meanwhile, valuable data characteristics relevant to classification are preserved.

Within a bi-class setting, we choose a recursive support vector machine (RSVM) [32] as a tool for identification of the relevant low-dimensional feature space $\mathcal{S}$. The detailed RSVM algorithm is presented in Algorithm 2, where l is the length of an initial training dataset and $\phi(\cdot)$ is the kernel function. If a linear kernel is applied, $\phi(x_i) = x_i$. RSVM was initially proposed for both dimensionality reduction and accuracy improvement for offline classification problems. It starts as a regular SVM [33] but can recursively derive new maximum margin features if the data cannot be well separated in terms of one direction only. Practitioners can a-priori restrict the dimension of the reduced feature space, or allow RSVM to automatically identify the number of components that are sufficient to account for the structure needed for successful classification.

Algorithm 2: RSVM [32]

1 Determine the vector $\tilde{w}_1 = \sum_{i=1}^{l} \alpha_i^1 \phi(x_i)$ by solving the dual optimization problem [33]
2 Let $w_{r-1} = \tilde{w}_{r-1}/||\tilde{w}_{r-1}||$ and generate the following training set for SVM problem by projecting the samples $x_i, 1 \leq i \leq l$ into a subspace that is orthogonal to w_{r-1}:

$$\phi(x_i^r) = \phi(x_i^{r-1}) - \langle \phi(x_i^{r-1}), w_{r-1} \rangle w_{r-1} \quad (3)$$

3 Go back to line 2 or Terminate if either $max\{||\phi(x_i^r)|| : 1 \leq i \leq l\} < \epsilon$ or the desired number of dimensions R has been reached.

Suppose we want to maintain a R-dimensional subspace. Based on an initial training set, Algorithm 2 provides us with one or several orthogonal directions $\{w_r, r = 1, ..., R\}$ which can be used as projectors to the R-dimensional subspace. Then each newly arrived instance $\mathbf{x}_j$ can be projected to the feature space as $\langle \phi(\mathbf{x}_j), w_r \rangle = \sum_{i=1}^{l} \alpha_i^r \kappa(\mathbf{x}_j, \mathbf{x}_i)$ for $r = 1, ..., R$ and $i = 1, ..., l$. It is worth pointing out that only $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ computation is concerned with the dual optimization problem, instead of the explicit kernel $\phi(x_j)$. From the second iteration, $\kappa(x_i^r, x_j^r)$ can be recursively computed by using (3) and $\kappa(\mathbf{x}_i^{r-1}, \mathbf{x}_j^{r-1})$, allowing different kernels to be adopted.

3.2 Class-based detection

While CDTs focus on detecting drifts by monitoring $P(Y|\mathbf{X})$ or $P(\mathbf{X})$, there has been a lack of attention paid to $P(\mathbf{X}|Y)$. Supervised information can be better utilized by class-conditional distributions because they focus on sub-regions of the input space. In HRDS, we suggest not only incorporating a distribution-based detector to inspect data features from the marginal distribution perspective, but also constructing one CDT for each class-conditional distribution $P(\mathbf{X}|Y = q)$, where $q \in \{0, 1, ..., Q\}$. The CDTs are initialized on its respective data stream (line 4, Algorithm 1). All CDTs are simultaneously placed on the incoming instances in the lower-dimensional subspace $\mathcal{S}$ derived earlier. Note that only the marginal detector and one of the class-conditional detectors are activated at each time stamp. Usually, the number of classes of a data stream is much lower than the number of dimensions. Therefore, HRDS is still expected to be computationally cheaper to implement than existing multivariate detectors that either try to estimate the

distribution density or monitor each dimension individually. By monitoring also the class-conditional distributions, HRDS captures both real and virtual drift, regardless of the effect on classification performance. Besides, since it synchronizes sub-regions of the input space, it is able to distinguish between drifted and non-drifted classes and its detection sensitivity over smaller drifts is enhanced.

Different techniques can be chosen as the base CDT for this component. ICI-based CDT has been used in the existing HCDT as a reference example. A dominant advantage of this sequential CDT is that it is endowed with a refinement procedure that directly provides the estimated drift starting time T_{ref} [34]. Thus, a new dataset representing the most recent concept is automatically identified. For other drift detectors, the method introduced in [35] is recommended to identify T_{ref}.

3.3 Knowledge base reconfiguration

Once a suspicious change is reported in the detection layer by at least one of the base detectors at time $\widehat{T}$, a potential drift starting time T_{ref} is estimated and the validation layer is activated (lines 8-9, Algorithm 1). Offline statistical test is used to compare the previous training set of the respective detector and instances from T_{ref} to $\widehat{T}$ to determine if the drift should be confirmed (line 10, Algorithm 1). If a drift is validated, the existing HCDT framework discards all past data and reconfigure based on the most recent data only. This approach may be over-conservative for a supervised data stream as a drift may have uneven effects on different classes. Unnecessary rejection of data in a relatively stationary class leads to information loss and can become problematic when available information is already scarce or expensive to obtain. Here we propose a novel way of reconstructing the re-training sets in order to maintain as much useful information as possible for detector reconfiguration. The idea can be summarized as follows:

1) for data streams where we can confirm that a change has taken place (with a detected and validated change), the respective detectors are immediately reconfigured based on the latest training dataset representing the current concept.
2) for data streams where there is ambiguity if a change has taken place (a detected but invalidated change), we do not make any amendment to the existing detector.
3) for data streams where we are inclined to believe that no drift has taken place (with no detection both from the class-conditional and the marginal detector), we combine the latest training instances with previous ones to form a more informative retraining set.

It should be noted that when one class-conditional detector reports and validates a change, it subsequently impacts the marginal distribution according to Equation (2), therefore in this case the marginal detector is always retrained. Herewith, the performance of the detectors is expected to improve as extra relevant instances are used for retraining.

Hotelling T2 test has been shown to be a suitable complementary validation test for ICI-based CDT in the existing HCDT framework. As a concrete realization under a bi-class scenario, the reconstruction scheme for all detectors after each detection can be summarized in Table 1.

4 Computational Studies

This section presents four sets of experiments that evaluate the effectiveness and efficiency of HRDS. Experiment 1 aims to demonstrate the effectiveness of each component of the HRDS framework which differentiates it from the existing HCDT. This set of experiments are carried out on data streams with a range of different dimensionalities to reveal its advantage on high-dimensional data. Experiment 2 illustrates the superiority of HRDS in drift detection on both real and virtual drifts over state-of-the-art methods. Experiment 3 validates that the superior performance provided by HRDS also benefits classification, even when integrated with a very simple classifier. Experiments 1-3 are based on datasets of synthetically generated sequences where the ground truth of drift occurrences is available. In Experiment 4, we demonstrate the role of HRDS on a real-world data stream. Finally, we provide a brief analysis on the computational time complexity of the approaches being considered in the experiments. All experiments were run on a CentOS 7.6 Computer with v4 2.20 GHz processor and 128 GB memory.

4.1 Performance metrics

A variety of performance metrics for drift detection have been used in the literature. For instance, when counting the number of True Positive (TP), False Negative (FN) and False Positive (FP), some authors focused more on whether a correct or wrong detection is raised on a drifted sequence, but not on the number of detections [36], [37]. Differently, some authors distinguish between *Detected*, *Late*, *Missed* and *False* detections based on sliding windows and paid attention to whether there were redundant detections after a *Detected* (TP) detection [19], [38]. In [9] all detections raised on a stream are taken into account and each single detection is categorized into TP or FP based on a

Table 1: Re-construction of training sets after drift detection. Without loss of generality, we assume the last instance received belongs to class 0. Analogous definitions can be made for class 1. TS^M, TS^0, TS^1 are the existing training sets for the marginal and class-conditional detectors respectively. TS_C^M is composed of all instances representing the current concept in $[T_{ref}, \widehat{T}]$. TS_C^0 (TS_C^1) denotes class 1 (class 0) instances from TS_C^M.

		Layer-I and II output from the Marginal Detector		
		Detected and Validated	Detected but Invalidated	No Detection
Layer-I and II output from Class 0 Conditional Detector	Detected and Validated	M: $TS^M = TS_C^M$ C0: $TS^0 = TS_C^0$ C1: $TS^1 = [TS^1, TS_C^1]$	M: $TS^M = TS_C^M$ C0: $TS^0 = TS0_C$ C1: $TS1 = [TS^1, TS_C^1]$	M: $TS^M = TS_C^M$ C0: $TS^0 = TS_C^0$ C1: $TS^1 = [TS^1, TS_C^1]$
	Detected but Invalidated	M: $TS^M = TS_C^M$ C0: $TS^0 = TS_C^0$ C1: $TS^1 = TS_C^1$	M: No retraining C0: No retraining C1: $TS^1 = [TS^1, TS_C^1]$	M: $TS^M = [TS^M, TS_C^M]$ C0: No retraining C1: $TS^1 = [TS^1, TS_C^1]$
	No Detection	M: $TS^M = TS_C^M$ CL0: $TS^0 = TS_C^0$ C1: $TS^1 = TS_C^1$	M: No retraining C0: $TS^0 = [TS^0, TS_C^0]$ C1: $TS^1 = [TS^1, TS_C^1]$	

specified window size. Later, the notion of acceptable delay Δ was formally introduced in [39]. Here, FPs are defined as detections outside of the acceptable detection interval $[t, t+\Delta]$, but extra detections within the interval are neglected. Based on the these previous evaluation paradigms, we would like to further distinguish between a true but delayed detection and a genuinely missed detection. In some real-world applications, these cases are associated with different penalties. In addition, from a practical point of view, distinguishing between various types of false alarms also helps the designer to understand which aspect of an algorithm needs to be modified to improve its performance.

Therefore, when analysing the results of a reactive detector, we propose a more realistic and comprehensive definition paradigm as in Fig. 3a. Based on a predefined acceptable detection delay range $[T^*, T^* + \Delta]$ where T^* is the real drifting time, we define a TP as the first detection within this range, a FN_missed as a missed alarm throughout the concept. We also distinguish between three types of FPs: FP_early, FP_duplicate and FP_late. A FP_early is the first false alarm before T^* related to algorithm initialization, FP_duplicate's are redundant false alarms related to algorithm reconfiguration, and a FP_late is the first detection in $[T^* + \Delta, T^{end}]$ when there is no alarm raised in $[T^*, T^* + \Delta]$. An illustrative example is presented in Fig. 3b.

The total number of FPs and FNs are therefore FP = FP_early + FP_duplicate + FP_late and FN = FN_late + FP_late respectively. Performance of the detector is evaluated via number of TPs, FPs, FNs or Recall, Precision and F-measure as defined in Fig. 3c. For each synthetic dataset in the experiments, 30 sequences are generated, and all reported figures are summations or averages. Detection performance is measured for several acceptable lengths $\Delta = \{500, 1000, 1500, 2000\}$ so as to limit the maximum detection delay allowed.

Experiment 1: Understanding HRDS

In order to better understand the novelty of HRDS relative to the existing hierarchical framework HCDT, we carry out a component-wise evaluation. The characteristics of HRDS and several variations containing only partial components are presented in Table 2. HCDT-M is the existing HCDT framework which monitors the marginal input distribution only. HCDT-CC is the existing HCDT framework applied to the class-conditional distributions. HDS is very similar to HRDS by employing univariate detectors on both marginal and class-conditional distributions but without the projection to the lower-dimensional feature space. The experiment is carried out on data streams of varying dimensionalities to find out how well HRDS can cope with multivariate data streams.

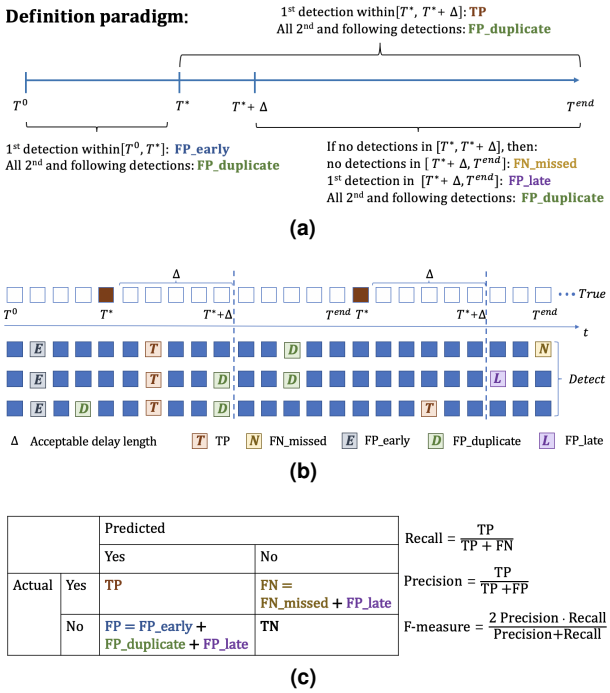


Figure 3: Detection performance definition paradigm. (a) describes how FP, TP and FN

Table 2: Compared detection frameworks in Experiment 1.

	detection			reconfiguration	
Algorithm	monitor $P(\mathbf{X})$	monitor $P(\mathbf{X}\|Y)$	feature space	single CDT	multiple CDTs
HRDS	✓	✓	✓		✓
HCDT-M (HCDT)	✓			✓	
HCDT-CC		✓		✓	
HDS	✓	✓			✓

Table 3: Synthetic data generation of d-dimensional hyperplane datasets.

Concept	$d-dimensional\ hyperplane$
1	$a_0^1 = -1.5; \quad a_i^1 = i \times 0.1 \ \forall\, i \in \{1, ..., d\}$
2	$a_0^2 = a_0^1 - 1; \quad a_i^2 = a_i^1 - 0.5 \ \forall\, i \in \{1, ..., d\}$

RSVM introduced in Algorithm 2 is chosen to find the appropriate lower-dimensional space in HRDS. For simplicity, we project the data onto a one-dimensional space in the experiment as it contains the most relevant information regarding classification. Then a univariate ICI-based CDT is used as a base detector. When comparing different structures of data-driven hierarchical frameworks, it should be noted that the performance of algorithms depends on specific parameters of the base CDT regulating the possibility of FPs. The following confidence parameter values $\Gamma \in \{2.25, 2.5, ..., 3.5\}$ are considered. Higher values of Γ reduce the probability of FPs at the expense of longer detection delays and possibly more FNs. Following [26], the initial training set length and the minimum retraining set size were set to 400 and 80, respectively.

Synthetic data generated for this experiment is a set of d-dimensional moving hyperplanes $y = -a_0 + \sum_i^d a_i x_i$, where $d = [5, 10, 15, 20, 25], x_i \in [0, 1]$ and $y \in [0, d]$. This is a popular dataset which has been used very often in the field [40], [41], [42]. The generation mechanism also allows easy alteration of the dataset dimension. Details of the data generation parameters can be found in Table 3. Each class contains 5000 data points with 5% of noise added. The data sequence consists of $10,000$ instances with one abrupt change at timestamp 5001. Total number of TPs and FPs are computed to compare the performance.

Due to the page limit, we report only the results for 2 selected acceptable delay length values Δ = 1000 and 2000 in Table 4. The following findings are also applicable to Δ = 500 and 1500. Firstly, we notice from Table 4 that HRDS achieves the highest TP in almost all cases. This is true even for a tight Δ, indicating that HRDS is able to not only detect the drifts, but also detect them earlier than the existing HCDT and other variations being considered. Meanwhile, HRDS always reports the lowest FP. In contrast, other methods monitoring each dimension of the data stream within the input space individually lead to much higher FP. HDS, which is also based on this novel reconfiguration scheme but does not project data into the low-dimensional space as HRDS does, always ranked second in terms of both TP and FP. In addition, comparing with the results of HCDT-CC, we can conclude that HRDS is very different from the existing HCDT applied on each class. The novelty of HRDS lies in not only class-based inspections, but also the projection of data to the low-dimensional space and the utilization of both marginal and class-conditional information in detector reconfiguration. As dimension increases, the number of FP detections raised by the compared methods increases dramatically and the superiority of HRDS becomes more dominant, confirming its ability to operate efficiently even for higher dimensional data streams. Also, comparing the performance presented in Table 4 horizontally, it can be seen that HRDS is relatively insensitive to the choice of base detector parameter Γ, making it a more reliable and stable detection framework among the compared methods.

Experiment 2: Drift detection ability

In this subsection we aim to compare the drift detection ability of HRDS on a wide range of drifts with state-of-the art methods, namely HCDT [26] and HLFR [43] introduced in Section 1. These consolidated frameworks have been shown to perform better than their individual base detector counterparts. We also compare HRDS with two classic performance-based benchmarks, DDM [6] and EDDM [7], which have not been used as base change detectors in the above mentioned frameworks. All hyper parameters of the detection and validation tests of the algorithms were directly taken as recommended and used by their authors. Recall that the detection result from performance-based detectors is contingent on the choice of classifier. Therefore, two classifiers from the range of classifiers that have been used in the original papers for HLFR, DDM and EDDM are adopted: an SVM and a decision tree. The SVM classifier adopts a linear kernel except for one non-linearly separable dataset, Rotating Checkerboard, where a radial basis function (RBF) kernel is applied. Following [26], the

Table 4: Performance comparison on data streams with various dimensions.

	Δ =	1000												2000											
		TP						FP						TP						FP					
	Γ =	2.25	2.5	2.75	3.0	3.25	3.5	2.25	2.5	2.75	3.0	3.25	3.5	2.25	2.5	2.75	3.0	3.25	3.5	2.25	2.5	2.75	3.0	3.25	3.5
5D	HRDS	**30**	**27**	**22**	**16**	11	7	**18**	**10**	**16**	**15**	19	23	**30**	**30**	**30**	**30**	**30**	**30**	18	7	8	**1**	**0**	**0**
	HCDT-M	5	2	3	0	0	0	31	27	27	26	21	**18**	24	23	24	11	8	4	**12**	**6**	**6**	15	13	14
	HCDT-CC	26	23	19	14	11	9	22	17	17	17	21	22	30	30	30	30	30	30	18	10	6	1	2	1
	HDS	23	25	19	16	**12**	**9**	19	11	18	16	**18**	22	29	26	29	30	30	30	13	10	8	2	0	1
10D	HRDS	**30**	**30**	**28**	**27**	**24**	**21**	**21**	**12**	**10**	**9**	**9**	**11**	**30**	**30**	**30**	**30**	**30**	**30**	**21**	**12**	**8**	**6**	3	**2**
	HCDT-M	27	27	26	21	20	18	157	66	35	13	10	17	28	29	30	28	29	29	156	64	31	6	**1**	6
	HCDT-CC	29	28	29	25	20	13	130	83	57	56	47	53	30	30	30	30	30	30	129	81	56	51	37	36
	HDS	25	28	28	25	24	21	114	81	59	21	24	15	27	29	30	29	29	30	112	80	57	17	19	6
15D	HRDS	**30**	**30**	**30**	**30**	**29**	**27**	**11**	**12**	**9**	8	**4**	**4**	**30**	**30**	**30**	**30**	**30**	**30**	**11**	**12**	**9**	8	**3**	1
	HCDT-M	23	27	28	28	28	24	344	203	83	**4**	16	6	27	29	29	29	30	30	340	201	82	**3**	14	**0**
	HCDT-CC	22	23	26	21	18	12	134	106	82	60	58	61	25	25	30	28	29	29	131	104	78	53	47	44
	HDS	21	25	28	29	28	27	216	176	119	40	40	13	27	28	30	29	30	30	210	173	117	40	38	10
20D	HRDS	**30**	**30**	**30**	**30**	**30**	**28**	**15**	**10**	**9**	**9**	**9**	7	**30**	**30**	**30**	**30**	**30**	**30**	**15**	**10**	**9**	**9**	**9**	5
	HCDT-M	19	21	24	26	28	27	405	211	123	40	18	**3**	23	23	26	26	29	30	401	209	121	40	17	**0**
	HCDT-CC	10	13	9	16	10	14	107	95	56	36	38	30	16	20	13	19	17	24	101	88	52	33	31	20
	HDS	17	19	18	20	22	27	238	164	96	68	58	22	23	21	21	22	25	29	232	162	93	66	55	20
25D	HRDS	**30**	**30**	**30**	**29**	**29**	**29**	**21**	**19**	**21**	**19**	**18**	**15**	**30**	**30**	**30**	**30**	**30**	**30**	**21**	**19**	**21**	**18**	**18**	**14**
	HCDT-M	29	29	28	26	24	19	842	515	278	155	92	66	30	30	30	30	30	30	841	514	276	151	86	55
	HCDT-CC	30	30	24	19	21	16	281	213	156	134	103	92	30	30	30	30	30	30	281	213	150	123	94	78
	HDS	29	29	28	27	24	23	425	301	195	117	85	60	29	30	30	30	30	30	425	300	193	114	79	53

Table 5: Synthetic data generation of 4D Multivariate Gaussian. The illustration is given in Fig. 4.

Concept	$4D\_Gaussian$			
	(a)	(b)	(c)	(d)
1	$\boldsymbol{\mu}^1_{C_0} = [0,0,0,0]$ $\boldsymbol{\mu}^1_{C_1} = [0.8,0.8,0.8,0.8]$ $\boldsymbol{\Sigma}^1_{C_0} = \boldsymbol{\Sigma}^1_{C_1} = \mathbb{1}_4$			
2	$\boldsymbol{\mu}^2_{C_0} = [-0.2,0.1,$ $-0.2,0.1]$	$\boldsymbol{\mu}^2_{C_0} = [-0.2,-0.2,$ $-0.2,-0.2]$	$\boldsymbol{\mu}^2_{C_0} = [0.4,-0.3,$ $0.4,0.4]$	$\boldsymbol{\mu}^2_{C_0} = [-0.3,-0.4,$ $-0.4,0.4]$
	$\boldsymbol{\mu}^2_{C_1} = \boldsymbol{\mu}^1_{C_1}$ $\boldsymbol{\Sigma}^2_{C_0} = \boldsymbol{\Sigma}^2_{C_1} = \mathbb{1}_4 + 0.2 \times (\mathbb{J}_4 - \mathbb{1}_4)$			

Table 6: Synthetic data generation of 6D Multivariate Gaussian datasets. The illustration is given in Fig. 5.

Concept	$6D\_Gaussian$
1	$\boldsymbol{\mu}^1_{C_0} = [2,2,3,3,4,4]$; $\boldsymbol{\mu}^1_{C_1} = [1,1,2,2,3,3]$; $\boldsymbol{\Sigma}^1_{C_0} = \boldsymbol{\Sigma}^1_{C_1} = \mathbb{1}_6$
2	$\boldsymbol{\mu}^1_{C_0} = [2.6,2.6,3.8,3.8,4.2,4.2]$ $\boldsymbol{\mu}^2_{C_1} = \boldsymbol{\mu}^1_{C_1}$; $\boldsymbol{\Sigma}^2_{C_0} = \boldsymbol{\Sigma}^2_{C_1} = \mathbb{1}_6$
3	$\boldsymbol{\mu}^1_{C_0} = [2.2,2.2,3.4,3.4,4.4,4.4]$ $\boldsymbol{\mu}^3_{C_1} = \boldsymbol{\mu}^2_{C_1}$; $\boldsymbol{\Sigma}^3_{C_0} = \boldsymbol{\Sigma}^3_{C_1} = \mathbb{1}_6$
4	$\boldsymbol{\mu}^1_{C_0} = [2.8,2.8,3.4,3.4,3.8,3.8]$ $\boldsymbol{\mu}^4_{C_1} = \boldsymbol{\mu}^3_{C_1}$; $\boldsymbol{\Sigma}^4_{C_0} = \boldsymbol{\Sigma}^4_{C_1} = \mathbb{1}_6$
5	$\boldsymbol{\mu}^1_{C_0} = [2.6,2.6,3.0,3.0,3.4,3.4]$ $\boldsymbol{\mu}^5_{C_1} = \boldsymbol{\mu}^4_{C_1}$; $\boldsymbol{\Sigma}^5_{C_0} = \boldsymbol{\Sigma}^5_{C_1} = \mathbb{1}_6$

parameter Γ is set to 2.5 and the initial training set length and the minimum retraining set size are both set to 160 for both HCDT and HRDS. Detection recall, precision and F-measure are plotted with y-axis corresponding to the selected acceptable delay length $\Delta = \{500, 1000, 1500, 2000\}$.

In this experiment we first test on data streams with one abrupt drift only. With drift affecting $P(Y|\mathbf{X})$ or not and its magnitude being small or large, there are 4 possible scenarios for a single drift. These cases will be examined individually. We then consider two more complicated multiple-drift datasets where more forms of drifts are present. The following synthetic datasets are generated for the experiment:

1) **4D Multivariate Gaussian** (Fig. 4): Here we synthetically generate drifts affecting the target concept differently by changing the class-distribution independently. For simplicity, we assume only one class drifted. Possible drift scenarios are visualized in Fig. 4. In order to reflect the 4 scenarios, 4 groups of 4D Multivariate Gaussian datasets are generated. Each data sequence consists of $10,000$ observations and a single

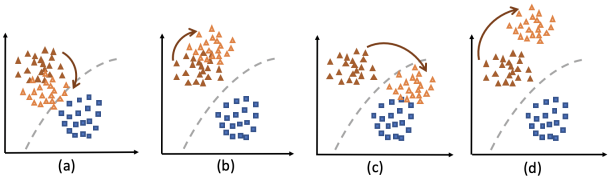


Figure 4: Illustration of various drift types of 4D Multivariate Gaussian. (a) small drift affecting $P(Y|\mathbf{X})$; (b) small drift not affecting $P(Y|\mathbf{X})$; (c) large drift affecting $P(Y|\mathbf{X})$; (d) large drift not affecting $P(Y|\mathbf{X})$. Data generation details are given in Table 5.

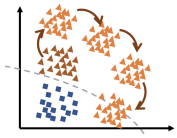


Figure 5: Illustration of 6D Multivariate Gaussian. Data generation details are given in Table 6

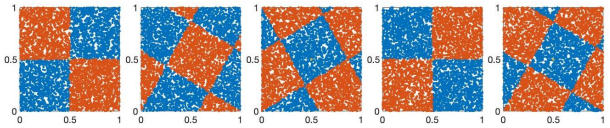


Figure 6: Illustration of Rotating Checkerboard

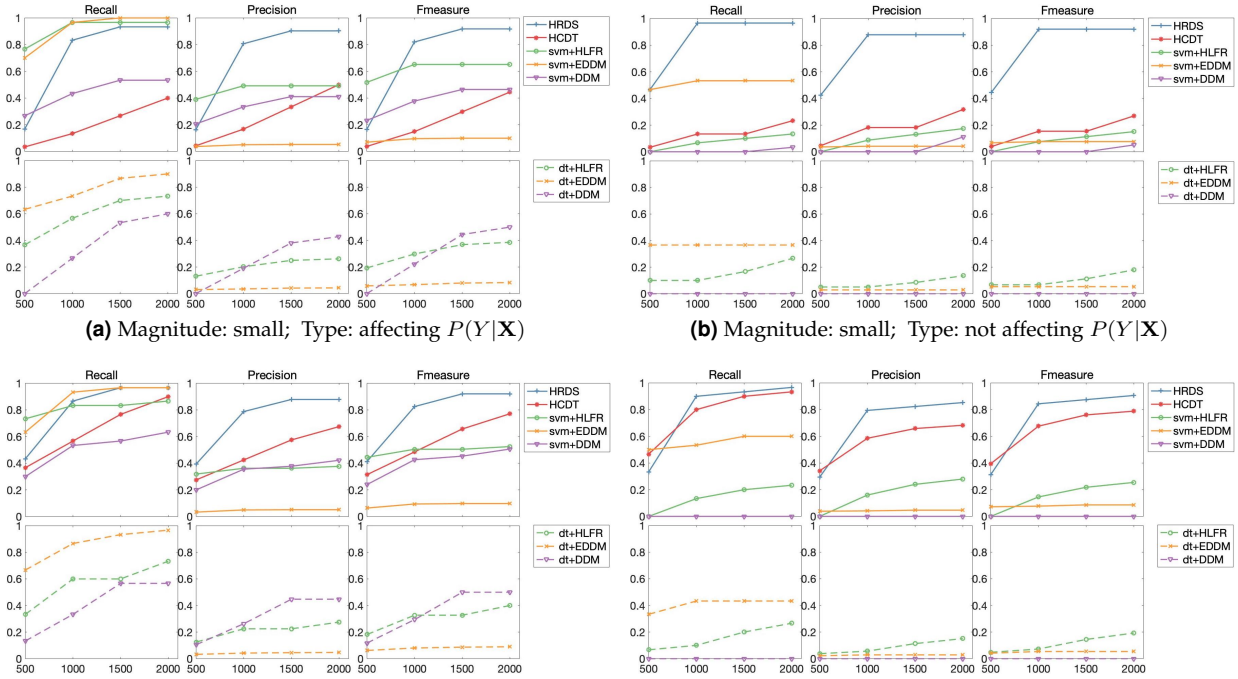
abrupt change takes place at instance 5001. The magnitude of drift is controlled by the change in within-class distance d_w and the effect on the target concept is controlled by the change in between-class distance d_b. The (d_w, d_b) pair for the stationary concept is always (0,0). For drifts with smaller magnitudes, the (d_w, d_b) pair for concept 2 is set to (0.5, -0.9) and (0.5, 0.8) reflecting scenarios (a-b) respectively. For drifts with greater magnitudes, the (d_w, d_b) pair for concept 2 is set to (1.0, -0.8) and (1.0, 1.1) reflecting scenarios (c-d) respectively. Data generation details can be found in Table 5.

2) **6D Multivariate Gaussian** (Fig. 5): moving to multiple drift scenarios, we first consider a scenario where a series of drifts is not detrimental to classification at the beginning, but can eventually impair the accuracy after several evolutions. A simple illustration of this situation is presented in Fig. 5. Each sequence is of length $25,000$ and contains 5 concepts. The drift magnitude is also controlled by (d_w, d_b) pairs. The evolution of concept is summarized as (0, 0), (0.4, 2.4), (0.4, 1.9), (0.4, -1.4) and (0.4, -2.1) successively. Details of the data generation process can be found in Table 6.

3) **Rotating Checkerboard** (Fig. 6): here we consider a common benchmark dataset first used by [44] for concept drift problems. In this dataset all 4 drifts lead to a strong change in classification boundary. Each stream is of length $25,000$ and contains 5 concepts. Examples are sampled uniformly from the unit square with dimensionality of 2 and the labels are set by a checkerboard with 0.5 tile width. At each concept drift, the checkerboard is rotated by an angle of $\pi/6$ radians.

Detection performance for **4D Multivariate Gaussian** is summarized in Fig. 7. Overall, HRDS ranked first in 14 out of the 16 cases (4 datasets and 4 Δ's) in terms of F-measure, indicating its ability to achieve the best trade-off between recall and precision. Performance-based detectors HLFR, EDDM and DDM only secure high recall values for real drifts affecting $P(Y|\mathbf{X})$ that cause an evident degradation in classification accuracy (Fig. 7a and Fig. 7c). Although the recall values attained by these methods are sometimes higher than HRDS, the precision values for these cases are rather low, indicating that they lead to much higher number of false alarms. For drifts not decreasing classification performance, i.e., drifts not affecting $P(Y|\mathbf{X})$ (Fig. 7b and Fig. 7d), performance-based detectors fail and the data-based detector HCDT becomes the second best detector after HRDS in terms of detection F-measure. In addition, HRDS also surpasses HCDT by a great amount when drift magnitude is small (Fig. 7a and Fig. 7b). This is due to the fact that the detection mechanism monitoring class-conditional distributions makes HRDS more sensitive for even the lightest changes on the overall input space. For drifts with greater magnitude (Fig. 7c and Fig. 7d), the performance of HCDT improves, but still it falls behind HRDS not only in terms of F-measure, but also in terms of recall and precision, except for one case.

Moving to the multiple-drifts scenarios, HRDS also outperforms its competitors in all 8 cases in terms of F-measure as shown in Fig. 8 and Fig. 9. For **6D Multivariate Gaussian** (Fig. 8), since the magnitude of each single drift is relatively small, it takes two or more consecutive drifts in order for the effect of the drift series to be sufficiently noticeable by HCDT, which monitors the input marginal distribution. Performance-based detectors HLFR, EDDM and DDM are only able to detect the last one or two drifts in Fig. 5, since earlier drifts do not deteriorate classification performance. On the **Rotating Checkerboard** data streams, the effectiveness of HRDS can also be clearly identified from Fig. 9. As expected, HCDT does not perform well because purely distribution-based detectors fail to detect changes affecting the labelling mechanism only [45]. The distribution of overall input space of this dataset remains unchanged. This phenomenon demonstrates that detecting concept drift by monitoring the class-conditional distributions is helpful. For this dataset, $P(Y|\mathbf{X})$ is significantly affected by all drifts, allowing the performance-based detectors to capture the drifts more acutely. Therefore HLFR, EDDM and DDM achieved very high recall values. However, this may not be a preferable outcome as significantly more false alarms are triggered if we examine the precision plot. Therefore, HRDS, which secures the highest F-measure, is still the most reliable



(a) Magnitude: small; Type: affecting $P(Y|\mathbf{X})$

(b) Magnitude: small; Type: not affecting $P(Y|\mathbf{X})$

(c) Magnitude: large; Type: affecting $P(Y|\mathbf{X})$

(d) Magnitude: large; Type: not affecting $P(Y|\mathbf{X})$

Figure 7: Detection performance for 4D Multivariate Gaussian against acceptable delay lengths. Subfigures (a-d) correspond to scenarios (a-d) in Fig. 4 respectively.

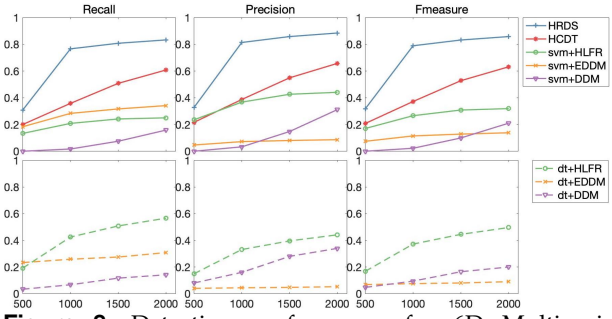


Figure 8: Detection performance for 6D Multivariate Gaussian. For performance-based detectors HLFR, EDDM and DDM: Linear SVM as the base classifier (top); decision tree as the base classifier (bottom).

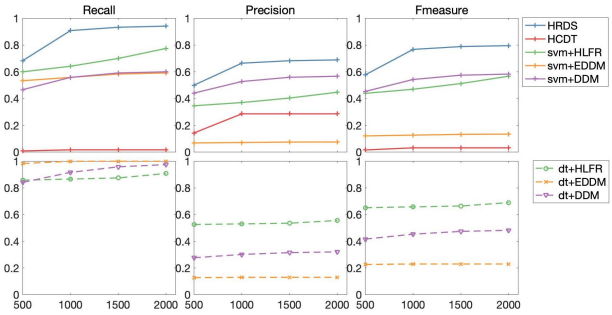


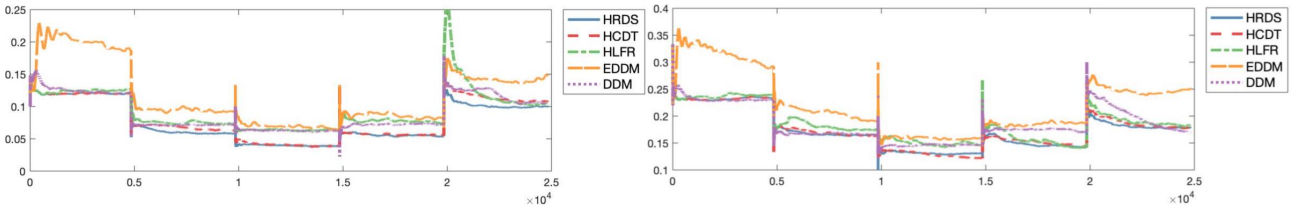
Figure 9: Detection performance for Rotating Checkerboard. For performance-based detectors HLFR, EDDM and DDM: RBF SVM as the base classifier (top); decision tree as the base classifier (bottom).

choice. Another interesting finding from Fig. 8 and Fig. 9 is that when a decision tree is used as the base classifier, HLFR and EDDM achieve much better than when an SVM classifier is used. This confirms that the choice of classifier plays an important role in performance-based drift detection. In contrast, the performance achieved by HRDS is invariant of the base classifier.

It can be concluded from this subsection of experiments that for real drifts affecting $P(Y|\mathbf{X})$, HRDS performs no worse, and in many cases better than existing performance-based detectors. For virtual drifts not directly affecting $P(Y|\mathbf{X})$, HRDS performs better than both distribution-based and performance-based detectors. HRDS also performs particularly better than the comparative methods when the changes have minor effect on the overall input distribution.

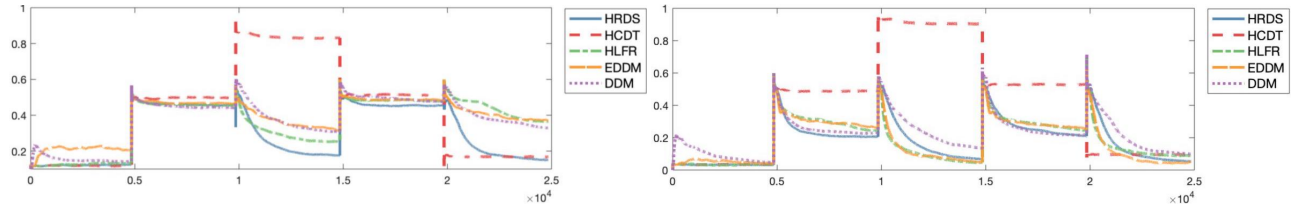
Experiment 3: Role in classification

The focus of this paper is to propose a new drift detector framework HRDS. Intuitively, accurate detection and localisation of drifts would help to improve classification because it leads to just-in-time model retraining. Which classifier and what classifier training techniques achieve the lowest classification error in a reactive streaming environment is a matter for future work. However, in order to evaluate the role of a more accurate and efficient detector in streaming data classification environments, we present the prequential



(a) Linear SVM as the base classifier **(b)** Decision tree as the base classifier

Figure 10: Prequential classification error plot against timestamp for 6D Multivariate Gaussian.



(a) RBF SVM as the base classifier **(b)** Decision tree as the base classifier

Figure 11: Prequential classification error plot against timestamp for Rotating Checkerboard.

classification error rate for **6D Multivariate Gaussian** and **Rotating Chekerboard** datasets. The prequential error rate [46] with a decay factor $0 < \lambda < 1$ [1] at timestamp i is defined as

$$E_i = \frac{S_i}{B_i} = \frac{L(y_i, \hat{y}_i) + \lambda S_{i-1}}{1 + \lambda B_{i-1}},$$

where $L(y_i, \hat{y}_i)$ is a classification 0-1 loss function, $S_1 = L_1$ and $B_1 = 1$. The performance-based detectors automatically outputs classification results for each instance. For distribution-based detectors HCDT and HRDS, a simple detected-then-retrain technique is adopted. Each time a detection is raised, the classifier is retrained based on the most recent 160 data instances, which is also the minimum length of retraining set for reconfiguring all detectors. Experiments are carried out with two classifiers, an SVM and a decision tree.

On the **6D Multivariate Gaussian** dataset, Fig. 10 clearly shows that HRDS helps to achieve a lower classification error rate for both SVM and decision tree classifiers. Recall that the first two drifts are virtual. The classification task actually became easier as the classes moved further away from each other. Performance-based detectors do not detect such drifts. However, even in these cases, HRDS, which accurately detects all types of drifts, leads to an even lower error rate than the performance-based detectors. This supports the hypothesis that when the optimal decision boundary has shifted but performance is not deteriorated, retraining the classifier can still be beneficial.

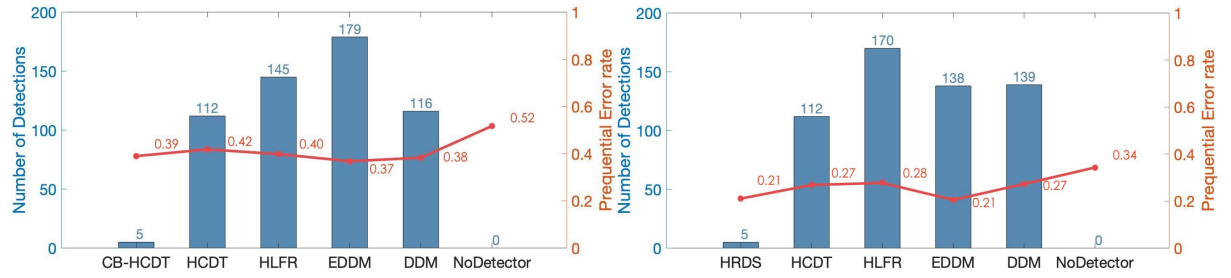
On the **Rotating Chekerboard** dataset, HRDS also leads to lower prequential classification error rate comparing with the other methods. This is very clear in Fig. 11a. In Fig. 11b, the classification error rate with EDDM appears to converge slightly quicker than HRDS on 2 out of the 5 concepts. Referring to the plots in Fig. 9, it can be seen that EDDM raises many FP detections. Hence, the respective classifier is almost being regularly retrained on the most recent data, so it adapts the new concept quicker. However, the unnecessarily frequent retraining of the classifier and the detector results in a high computational burden. Therefore, this is not an optimal outcome in practice. In contrast, the failure of data distribution-based detector HCDT in detecting drifts affecting the labelling mechanism only resulted in much higher classification error rate than the performance-based detectors.

Overall, HRDS can help in reducing classification error rate regardless of the drift type. For both real and virtual drifts, incorporating HRDS in a classification model can achieve a lower or at least comparable classification error rate than both performance-based and distribution-based detectors.

Experiment 4: Real-world scenarios

In the above experiments, synthetic data streams are used to better understand the functionality, efficiency and effectiveness of HRDS. For real-world data streams, there is no ground truth regarding the existence or location of drifts. Therefore, the performance metrics used for synthetic datasets cannot be employed. Here we report the number of detections and prequential classification error rate to compare the methods. We also compare the error rate with the situation where no detector is adopted. A classification system that achieves the lowest number of detections as well as the lowest classification error is the most desirable.

1. λ is set to 0.999



(a) Linear SVM as the base classifier **(b)** Decision tree as the base classifier

Figure 12: Detection and classification performance for the Electricity data stream. The bar plot represents the number of detections each method raised, and the line plot records the prequential classification error rate at the end of the data stream.

Electricity [47] is a dataset collected from the Australian New South Wales Electricity Market. It contains 45,312 instance and each example is described by 8 features. The class label identifies the change of the price relative to a moving average of the last 24 hours. (i.e., up and down). We note that there has been a dispute regarding the usage of this dataset for concept drift detection analysis due to the temporal correlation within the data [48]. Nonetheless it is still one of the most commonly used real-world data streams in this area of research [49].

Number of detections and the classification error rate obtained by the methods concerned are presented in Fig. 12. From the line plot representing the error rate, it can be seen that adding a drift detector always help reducing the classification error rate since all methods lead to lower error rate when comparing with the no detector situation. From the bar plot representing the number of detections, it can be seen that HRDS always ranked first with only 5 detections regardless of the choice of base classifier, meaning that it bears a very low computational burden from retraining. In contrast, its competitors all raise over 100 detections, causing a much higher overhead cost. Although HRDS requires the least number of retraining, it still helps to maintain a satisfying classification error rate. In Fig. 12a where an SVM is used as the base classifier, the error rate obtained by HRDS ranked third and is only 2% higher than that obtained with the first-place detector EDDM. Meanwhile, EDDM requires 174 more times of classifier and detector reconfiguration than HRDS. In Fig. 12b where a decision tree is adopted, HRDS and EDDM both ranked first in terms of classification error rate, but the difference between their required numbers of reconfiguration is as high as 133. Examining the number of detections and error rate simultaneously, we may conclude that in summary, HRDS achieves a better trade-off between classification performance and computational cost on this real-world data stream.

Table 7: Average runtime for each reported detection (s.).

Dataset	HRDS	HCDT	HLFR	EDDM	DDM
4D Gaussian	0.2496	0.1044	49.0637	0.2645	2.1117
6D Gaussian	0.6129	0.2989	63.3693	0.5735	3.0722
Rotating Checkerboard	0.2004	0.4088	36.2308	1.1619	6.9635

Computational time complexity analysis

DDM [6] and EDDM [7] have a constant time complexity ($\mathcal{O}(1)$) at each time point, since they monitor a single error-rate based statistics. Although the base detector LFR [10] in HLFR [43] also has complexity ($\mathcal{O}(1)$), the validation layer requires extra training of P classifiers (P=1000 in the original paper). Assuming $\mathcal{O}(K)$ is the computational complexity of training a new classifier, the time complexity for HLFR is $\mathcal{O}(KP)$, which is usually much higher than ($\mathcal{O}(1)$). HCDT [26] adopts a univariate test on each dimension in the detection layer and one offline test in the validation layer. If the complexity of the base detector (e.g., ICI-based CDT) is ($\mathcal{O}(1)$), the complexity of the overall framework is ($\mathcal{O}(d)$) where d is the dimensionality of input space. HRDS has a similar structure but adopts a univariate test on each dimension of the low-dimensional feature space for each class in the detection layer. The time complexity is ($\mathcal{O}(rQ)$) (r = 1 and Q = 2 in this paper so ($\mathcal{O}(rQ)$) is close to ($\mathcal{O}(1)$)). For multivariate data streams of higher dimensionality, the advantage of HRDS will become more significant since the number of classes is usually much lower than the number of dimensions. The average time for a reported detection for **4D Multivariate Gaussian**, **6D Multivariate Gaussian** and **Rotating Checkerboard** is summarized in Table 7. It is worth noting that performance-based detectors generally have longer runtime since they also include a classifier training procedure which data distribution-based detectors do not. Practitioners should take this into consideration when choosing the appropriate detector depending on the application scenario.

5 Conclusion

We have proposed a data distribution-driven and class-based hierarchical change detection framework (HRDS) for multivariate supervised data streams. The proposed framework first maps the data to a lower dimensional space and then detects drifts in that space relevant to the given classification task. It utilizes information from both marginal input distribution and class-conditional distributions of the supervised data stream. Based on the effect of drift on each class, a novel reconfiguration scheme aiming to maintain as many as possible relevant instances for retraining is incorporated within the algorithm. HRDS detects both real and virtual drifts, regardless of their effects on classification. HRDS is capable of detecting subtle drifts which can hardly be captured by existing distribution-based detectors. It is also computationally light and efficient when operating on higher-dimensional data streams. The proposed approach outperformed others by achieving a better recall-precision trade off within the given acceptable delay length when compared with the latest distribution-based and performance-based detectors in the literature.

The framework can be used with different types of base change detectors. Currently the HRDS framework is only tested on bi-class data streams with abrupt drifts. The framework can be easily extended for applications with multi-class data streams, or scenarios with gradual drifts by switching the tool of dimensionality reduction and the base detector. Moreover, since HRDS utilizes class-conditional distributions, modification to accommodate imbalanced-class data streams is also worth investigation.

References

[1] A. Tsymbal, "The problem of concept drift: definitions and related work," *Computer Science Department, Trinity College Dublin*, vol. 106, no. 2, p. 58, 2004.

[2] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346–2363, 2018.

[3] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Woźniak, and F. Herrera, "A survey on data preprocessing for data stream mining: Current status and future directions," *Neurocomputing*, vol. 239, pp. 39–57, 2017.

[4] S. Wang, L. L. Minku, and X. Yao, "A systematic study of online class imbalance learning with concept drift," *IEEE Transactions on Neural Networks and Learning Systems*, 2017.

[5] H. Hu, M. Kantardzic, and T. S. Sethi, "No free lunch theorem for concept drift detection in streaming data classification: A review," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 10, no. 2, p. e1327, 2020.

[6] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Brazilian symposium on artificial intelligence*. Springer, 2004, pp. 286–295.

[7] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno, "Early drift detection method," *Proceedings of the 4th ECML PKDD International Workshop on Knowledge Discovery From Data Streams (IWKDDS'06)*, 2006.

[8] 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.

[9] M. Harel, S. Mannor, R. El-Yaniv, and K. Crammer, "Concept drift detection through resampling," in *International Conference on Machine Learning*, 2014, pp. 1009–1017.

[10] 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.

[11] R. S. Barros, D. R. Cabral, P. M. Gonçalves Jr, and S. G. Santos, "Rddm: Reactive drift detection method," *Expert Systems with Applications*, vol. 90, pp. 344–355, 2017.

[12] A. Pesaranghader, H. L. Viktor, and E. Paquet, "Mcdiarmid drift detection methods for evolving data streams," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–9.

[13] D. R. de Lima Cabral and R. S. M. de Barros, "Concept drift detection based on fisher's exact test," *Information Sciences*, vol. 442, pp. 220–234, 2018.

[14] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[15] T. Dasu, S. Krishnan, S. Venkatasubramanian, and K. Yi, "An information-theoretic approach to detecting changes in multi-dimensional data streams," in *In Proc. Symp. on the Interface of Statistics, Computing Science, and Applications*. Citeseer, 2006.

[16] R. Sebastião and J. Gama, "Change detection in learning histograms from data streams," in *Portuguese Conference on Artificial Intelligence*. Springer, 2007, pp. 112–123.

[17] C. Alippi, G. Boracchi, and M. Roveri, "Change detection tests using the ici rule," in *Neural Networks (IJCNN), The 2010 International Joint Conference on*. IEEE, 2010, pp. 1–7.

[18] G. Ditzler and R. Polikar, "Hellinger distance based drift detection for nonstationary environments," in *2011 IEEE symposium on computational intelligence in dynamic and uncertain environments (CIDUE)*. IEEE, 2011, pp. 41–48.

[19] N. Lu, G. Zhang, and J. Lu, "Concept drift detection via competence models," *Artificial Intelligence*, vol. 209, pp. 11–28, 2014.

[20] L. Bu, D. Zhao, and C. Alippi, "An incremental change detection test based on density difference estimation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 10, pp. 2714–2726, 2017.

[21] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM computing surveys (CSUR)*, vol. 46, no. 4, p. 44, 2014.

[22] R. Benenson, M. Mathias, T. Tuytelaars, and L. Van Gool, "Seeking the strongest rigid detector," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3666–3673.

[23] S. Liu, M. Yamada, N. Collier, and M. Sugiyama, "Change-point detection in time-series data by relative density-ratio estimation," *Neural Networks*, vol. 43, pp. 72–83, 2013.

[24] W. J. Faithfull, J. J. Rodríguez, and L. I. Kuncheva, "Combining univariate approaches for ensemble change detection in multivariate data," *Information Fusion*, vol. 45, pp. 202–214, 2019.

[25] P. Sobolewski and M. Wozniak, "Concept drift detection and model selection with simulated recurrence and ensembles of statistical detectors." *J. UCS*, vol. 19, no. 4, pp. 462–483, 2013.

[26] C. Alippi, G. Boracchi, and M. Roveri, "Hierarchical change-detection tests," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 2, pp. 246–258, 2017.

[27] W. Härdle and L. Simar, *Applied multivariate statistical analysis*. Springer, 2007, vol. 22007.

[28] S. Yu, Z. Abraham, H. Wang, M. Shah, Y. Wei, and J. C. Príncipe, "Concept drift detection and adaptation with hierarchical hypothesis testing," *Journal of the Franklin Institute*, vol. 356, no. 5, pp. 3187–3215, 2019.

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

[30] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean, "Characterizing concept drift," *Data Mining and Knowledge Discovery*, vol. 30, no. 4, pp. 964–994, 2016.

[31] J. Gao, W. Fan, J. Han, and P. S. Yu, "A general framework for mining concept-drifting data streams with skewed distributions," in *Proceedings of the 2007 SIAM International Conference on Data Mining*. SIAM, 2007, pp. 3–14.

[32] Q. Tao, D. Chu, and J. Wang, "Recursive support vector machines for dimensionality reduction," *IEEE Transactions on Neural Networks*, vol. 19, no. 1, pp. 189–193, 2008.

[33] V. Vapnik, "Pattern recognition using generalized portrait method," *Automation and remote control*, vol. 24, pp. 774–780, 1963.

[34] C. Alippi, G. Boracchi, and M. Roveri, "Adaptive classifiers with ici-based adaptive knowledge base management," in *International Conference on Artificial Neural Networks*. Springer, 2010, pp. 458–467.

[35] M. Basseville, I. V. Nikiforov *et al.*, *Detection of abrupt changes: theory and application*. prentice Hall Englewood Cliffs, 1993, vol. 104.

[36] C. Alippi, G. Boracchi, and M. Roveri, "A hierarchical, nonparametric, sequential change-detection test," in *The 2011 International Joint Conference on Neural Networks*. IEEE, 2011, pp. 2889–2896.

[37] Y. Kim and C. H. Park, "An efficient concept drift detection method for streaming data under limited labeling," *IEICE Transactions on Information and systems*, vol. 100, no. 10, pp. 2537–2546, 2017.

[38] F. Gu, G. Zhang, J. Lu, and C.-T. Lin, "Concept drift detection based on equal density estimation," in *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 24–30.

[39] A. Pesaranghader and H. L. Viktor, "Fast hoeffding drift detection method for evolving data streams," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2016, pp. 96–111.

[40] L. L. Minku, A. P. White, and X. Yao, "The impact of diversity on online ensemble learning in the presence of concept drift," *IEEE Transactions on knowledge and Data Engineering*, vol. 22, no. 5, pp. 730–742, 2010.

[41] N. Lu, J. Lu, G. Zhang, and R. L. De Mantaras, "A concept drift-tolerant case-base editing technique," *Artificial Intelligence*, vol. 230, pp. 108–133, 2016.

[42] M. Tennant, F. Stahl, O. Rana, and J. B. Gomes, "Scalable real-time classification of data streams with concept drift," *Future Generation Computer Systems*, vol. 75, pp. 187–199, 2017.

[43] S. Yu, Z. Abraham, H. Wang, M. Shah, and J. C. Príncipe, "Concept drift detection and adaptation with hierarchical hypothesis testing," 2017, pp. 768–776.

[44] 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.

[45] C. Alippi, G. Boracchi, and M. Roveri, "Just-in-time classifiers for recurrent concepts," *IEEE transactions on neural networks and learning systems*, vol. 24, no. 4, pp. 620–634, 2013.

[46] J. Gama, R. Sebastião, and P. P. Rodrigues, "Issues in evaluation of stream learning algorithms," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 329–338.

[47] M. Harries and N. S. Wales, "Splice-2 comparative evaluation: Electricity pricing," *Artificial Intelligence Group, School of Computer Science and Engineering, University of New South Wales*, 1999.

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

[49] 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.

Shuyi Zhang

Peter Tino

Xin Yao