Concept Drift Detection Through Resampling

Maayan Harel MAAYANGA@TX.TECHNION.AC.IL
Koby Crammer KOBY@EE.TECHNION.AC.IL
Ran El-Yaniv RANI@CS.TECHNION.AC.IL
Shie Mannor SHIE@EE.TECHNION.AC.IL
Technion - Israel Institute of Technology, Haifa, Israel.

Abstract

Detecting changes in data-streams is an important part of enhancing learning quality in dynamic environments. We devise a procedure for detecting concept drifts in data-streams that relies on analyzing the empirical loss of learning algorithms. Our method is based on obtaining statistics from the loss distribution by reusing the data multiple times via resampling. We present theoretical guarantees for the proposed procedure based on the stability of the underlying learning algorithms. Experimental results show that the method has high recall and precision, and performs well in the presence of noise.

1. Introduction

Effective techniques for analyzing streaming data are required in many big data applications and pose new machine learning challenges. One major challenge arises when the underlying source generating the data is not stationary, in which case the concept to be learned changes through time. For example, in prediction tasks, such as fraud detection or user preference prediction, the performance of a static predictor that was previously trained, is bound to degrade over time, as the nature of fraudulent attacks, or personal preferences is evolving. A reliable detection of such changes can be used to maintain high performance and meaningful analyses of data streams. This work is concerned with detection of *change in the context of a prediction problem*, and proposes a detection scheme for online identification of times along the stream in which such changes have occurred.

Changes of the prediction task can be caused by changes in the input data, the target concept, or both. One option is to try detecting the change in the data generating distribution, a renowned problem in statistics termed *change-point detection*; see Section 1.1 for further discussion. However, while a drift in the generating distribution may result in a change in the learning problem, this is not a necessity. For example, if a sensor is damaged at some point, yet these changes affect features with a low correlation with the label, or the changes do not affect the decision boundary of the predictor, they should be ignored. Detecting these types of irrelevant changes, and consequently initiating a new training phase, may degrade the quality of the resulting predictor by unnecessarily shrinking the training set size. Moreover, detection of any type of distributional change is a challenging task, especially in high dimensional data. Therefore, when high prediction quality is the goal, we adopt the principle of Vapnik (1998) and propose to solve the problem directly by monitoring the drift in the prediction loss of the underlying predictor rather than the intermediate problem of change-point detection.

The main idea is to consider concept change only with respect to some hypothesis class. This approach enjoys no false detection of changes that are irrelevant to the end task. Using this approach, the sample and computation complexity are conveniently controlled by the choice of the hypothesis class. As an example consider the task of predicting the rating of a stream of book reviews of a user, where the genre of the books being reviewed changes with time, and the hypothesis class is restricted to linear predictors with constrained L1 norm. For simplicity of exposition consider using a very small dictionary instead of the L1 norm constraint. If the dictionary is {bad, good, informative}, the change from ‘fantasy’ to ‘educational’ books will be easily detected, due to the correlation of the word “informative” with a good educational book. If the dictionary is {bad, good, kindle}, we won’t detect the change of the user to kindle, as kindle is not correlated with the sentiment. However, if we inspect the distributional change in the input we may detect an irrelevant change.

The idea of focusing on the change in the target concept through the error of the underlying predictor instead of focusing on changes in the stream distribution has been previously proposed only in the context of classification; see

Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32. Copyright 2014 by the author(s).

Section 1.1 for details. Yet, a general-purpose model with theoretical guarantees for any prediction task, be it classification, regression, clustering etc., has not yet been considered, and is the focus of this work.

In this paper, we term changes in the prediction loss *concept drifts* and focus on their detection in streaming data. Our definition of a concept drift (see Definition 2) slightly deviates from the standard definition, which associates drifts only with changes in the target concept (Tsymbal, 2004), while our terminology links a concept drift also to the learning algorithm and its function class.

Our main contribution is a general approach for detection of change in a prediction problem for different types of concept drifts. Our detection mechanism relies on the idea of random permutations of the examples, which generate multiple train-test splits from the stream. The proposed approach is invariant to the learning problem, specifically to the loss function and to the learning algorithm. We analyze the detection quality of the proposed scheme for different types of changes in the prediction problem, from the simplest scenario of an abrupt change to that of a slow noisy gradual change. We also show empirically the success of the permutation-based detection.

1.1. Related Work

A closely related problem to *concept drift detection*, pursued in this paper, is that of *change-point detection*; the goal of the first is to detect changes that affect the mapping from the input space to the target concept, while the goal of the latter is to detect changes in the generating distributions of the time-series. A great deal of work has been invested in methods for change-point detection; some reference books include (Basseville & Nikiforov, 1993; Chen & Gupta, 2012), for parametric methods, and (Brodsky & Darkhovsky, 1993), for non-parametric methods.

Change-point detection is associated with homogeneity testing, in which, given two samples, one has to determine whether they were generated by the same distribution. Many attempts have been made to extend classical statistical tests for homogeneity for detection of change in time-series (Kifer et al., 2004; Lung-Yut-Fong et al., 2011). Other methods are based on a predefined parametric model, such as the generating distribution (Basseville & Nikiforov, 1993; Gustafsson, 1996; Lavielle & Teyssiere, 2006), autoregressive models (Takeuchi & Yamanishi, 2006), and state-space models (Moskvina & Zhigljavsky, 2003; Kawahara et al., 2007). Their success depends on the compatibility of the preassigned model. Nonparametric alternatives often rely on the estimation of density functions (Dasu et al., 2006; Sebastião & Gama, 2007), which tends to be problematic in high dimensional problems. A different direction is to directly estimate the density ratio (Kawahara & Sugiyama, 2012; Liu et al., 2013), while Desobry et al. (2005) propose to segment the series using one-class support vector machine. Vovk et al. (2003) pursue a related problem: exchangeability of the samples in the stream[1].

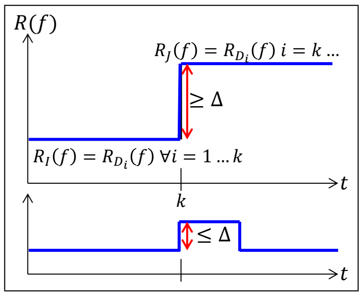
The above list is merely a small sample of the numerous attempts at the challenging change-point detection problem. We suggest to evade its complexity when the goal is to detect concept drift. Existing approaches for detection of a *concept-drift* are commonly based on some heuristic that utilizes the error rate, and draw upon intuition derived from learning theory. For example, in Gama et al. (2004), a change is detected when the error trend increases as the size of the stream grows. Another detection method (Baena-García et al., 2006), detects a concept drift once the distance between the classification errors decreases. A third error-based method, compares the accuracy on a recent window with the overall accuracy excluding the recent window by applying a test of equal proportion (Nishida & Yamauchi, 2007). Other methods search directly for the "optimal" window (Klinkenberg & Joachims, 2000; Bifet & Gavalda, 2007). For example, Klinkenberg & Joachims search for a window that minimizes the approximate error of an SVM classifier. The advantage of most error-based methods is their simple employment as wrappers to any classification algorithm. A limitation of known detection schemes is that they only treat classification settings under the zero-one loss. Theoretical guarantees for existing schemes are also scarce.

Other approaches for learning in the presence of concept drift don't apply detection but adapt to the change or employ an ensemble of learners. Adaptation methods retrain the learner after a fixed window size, or re-weigh instances by appearance recency (Klinkenberg, 2003; 2004). The choice of a proper window size balances a tradeoff between adaptation speed and generalization. Some adaptation methods are tailored to a specific learning algorithm, such as decision trees, and modify elements of the model when drift is suspected by some heuristic strategy (Black & Hickey, 1999; Hulten et al., 2001). In ensemble methods, a weighted voting scheme of multiple learners, trained on different windows over the stream is applied (Street & Kim, 2001; Stanley, 2003; Wang et al., 2003; Kolter & Maloof, 2007; Masud et al., 2009). There is no explicit detection of the drift, but initiation of new classifiers.

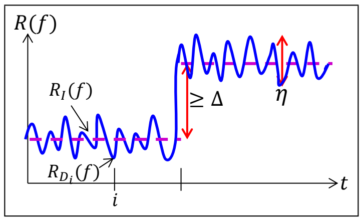
2. Detection Scheme

A prediction problem is defined on an input space $\mathcal{X}$, an output space $\mathcal{Y}$, and consists of finding a predictor $h : \mathcal{X} \to \mathcal{Y}$, from some fixed function class $\mathcal{H}$. During the

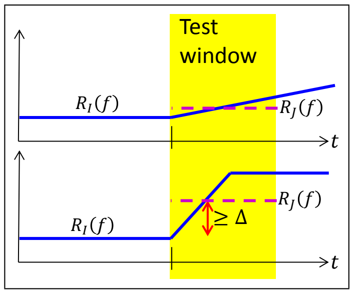
[1]This work has a similar spirit to our approach as it also employs permutations, however, the hypotheses tested are distinct.


$R(f)$
$R_J(f) = R_{D_i}(f)\ i = k \dots$
$\geq \Delta$
$R_I(f) = R_{D_i}(f)\ \forall i = 1 \dots k$
t
k
$\leq \Delta$
t

(a) Abrupt change



(b) Noisy abrupt change



(c) Gradual change

Figure 1. Different types of concept drifts.

detection process we observe a sub-sequence $Z_n \doteq Z_1^n = \{z_1, z_2, ..., z_n\}$ of independent observations emitted from a stochastic process called a stream, where each observation, $z_t \in \mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, is generated by some distribution $\mathcal{D}_t$ over $\mathcal{Z}$. Under this definition, even within a single concept, the distribution generating different examples may vary. The risk of $h \in \mathcal{H}$ with respect to $\mathcal{D}_t$ is $R_{\mathcal{D}_t}(h) = \mathbb{E}_{z\sim\mathcal{D}_t}\left[\ell(h, z)\right]$, where $\ell(h, z) : \mathcal{Y} \times \mathcal{Y} \to [0, 1]$ is a given bounded loss function. We denote the average risk over an index set I, not necessarily an ordinal one, by $R_I(h) = \frac{1}{|I|}\sum_{t\in I} R_{\mathcal{D}_t}(h)$, and its empirical estimate by $\hat{R}_I(h) = \frac{1}{|I|}\sum_{t\in I} \ell(h, z_t)$.

We would like to detect different types of concept changes. In the following, we formulate different kinds of concept changes, from a sharp abrupt change, to a more realistic noisy change, and finally, a slowly evolving noisy gradual change. For each type of change, we provide a suitable detection algorithm and a corresponding statistical guaranty. The presentation begins with the simplest and progresses to the most difficult one. Note that the algorithm for noisy gradual change also detects simpler drift types, and therefore may be applied as an all-purpose detection scheme.

We follow a hypothesis testing paradigm. Our null hypothesis $\mathcal{H}_0$ is of equality or proximity of the average true risk on sequential training and test segments. The alternative $\mathcal{H}_1$ asserts that there is a substantial difference between the two. Variations of this hypothesis, which are suited to different types of concept changes, are presented below.

2.1. Abrupt (noisy) change

The first, and most simplistic concept change is an *abrupt change,* where we assume that within each concept the risk is constant, and between the concepts, due to the concept switch, there is a change in the risk's value. The model includes a parameter $\Delta \geq 0$ that defines the rate of change to be tested. The threshold Δ serves as a sensitivity valve, which provides the means to tradeoff small changes in the concept with a growing training size. The top image in Figure 1a shows an abrupt change, while the bottom image presents a deviation that is under the threshold.

In the first step of our basic detection scheme the observed sub-sequence Z_n is divided into a training window $S_{ord} = Z_1^k$, $1 < k < n$, and a test window $S'_{ord} = Z_{k+1}^n$. If a change is not detected, the test window is added to the training window and a new test window is introduced. Detection is done by comparing the loss on the test window with the loss of multiple test sets of the same size obtained by random shuffling of the examples. This idea is formulated in Algorithm 1. The shuffled train-test partitions of the data are denoted by (S, S'), such that S consists of k elements of Z_n, and S' is $Z_n \setminus S$. The predictor $A_S \in \mathcal{H}$ is obtained by training algorithm A on a set S. We denote by $f \doteq A_{S_{ord}}$ the predictor obtained when A is trained on the ordered training set.

In each iteration of Algorithm 1 we compare the risk on the ordered train-test split, $\hat{R}_{ord} = \hat{R}_{S'_{ord}}(A_{S_{ord}})$, with the permutation loss $\hat{R}_{perm} = \mathbb{E}_{S\sim\mathcal{U}_n}[\hat{R}_{S'}(A_S)]$, where $\mathcal{U}_n$ denotes the uniform distribution over all possible $\binom{n}{k}$ training sets of size k from the sample Z_n. The permutation loss may be estimated over $P \leq \binom{n}{k}$ random partitions. The comparison between the losses obtained on the ordered split and the shuffled splits is done by an hypothesis test, denoted by "TEST" in Algorithm 1. The TEST may be done by thresholding the difference between the losses by applying the bound provided next in Theorem 3, or by applying a permutation test detailed in Section 3.

The intuition behind this scheme is that if no concept change has occurred, the prediction on the ordered split should not deviate too much from that of the shuffled splits, especially if the learning algorithm has algorithmic stability. An advantage of using shuffled train-test splits over other error based schemes, as in (Gama et al., 2004), is that the training and test sets are disjoint. Therefore, the test error is a valid estimation of the risk.[2] In addition, by using

[2]In other methods, the online error is computed, in which case the test errors are no longer independent, thus the classification

multiple train-test splits we gain a good estimation of the error under the null hypothesis, as well as valuable information regarding its variation.

Note that the model presented in this paper assumes temporal independence of the samples. In cases of dependency, means to maintain exchangeability of the samples, such as block-wise permutations along with removing the initial part of each block, may be applied.

A generalization of abrupt change is *noisy abrupt change*, where within each concept the risk may vary up to some rate. These variations are common in most real-life scenarios, where the process generating the stream has small changes, which affect the risk but should be disregarded. We define these permitted variations as follows.

Definition 1 (η-permitted variations)**.** *A stream segment $[t_1, t_2]$ is said to have η-permitted variations, for some $\eta \geq 0$, with respect to $h \in \mathcal{H}$, if $\max_{i,j \in [t_1,t_2]} \left| R_{\mathcal{D}_i}(h) - R_{\mathcal{D}_j}(h) \right| \leq \eta$.*

Figure 1b illustrates an η-permitted variation and a noisy abrupt change. A concept change is defined by:

Definition 2 (Concept change w.r.t. $h \in \mathcal{H}$)**.** *Let I, J be two sequential stream segments with η-permitted variations w.r.t. h. There is a concept change of size $\Delta > \eta$ between I and J if $|R_J(h) - R_I(h)| \geq \Delta$.*

The null and alternative hypotheses for a (noisy) abrupt change are defined as follows. Let $I = [1, k]$ and $J = [k+1, n]$ define the ranges of two consecutive stream segments with η-permitted variations, corresponding to the training and test set windows. Then,

$$\begin{aligned} \mathcal{H}_0 &: \ |R_I(f) - R_J(f)| \leq \Delta, \\ \mathcal{H}_1 &: \ |R_I(f) - R_J(f)| \geq \Delta + \gamma(\eta). \end{aligned} \tag{1}$$

The function $\gamma(\eta) \geq 0$ depends on the permitted variation and other elements of the algorithm (see Theorem 3 below for further details). Notice that the hypotheses are parameterized by the prediction function f, which is trained on the training window I.

Our analysis relies on algorithmic stability (Bousquet & Elisseeff, 2002), which is reviewed in the supplementary material. The following theorem bounds the difference between the risks under the null and alternative hypotheses. The proof is provided in the supplementary material. The slack γ quantifies an area where a change cannot be detected, as defined in (1); it shrinks as the algorithm is more stable and the training and test sets are larger.

Theorem 3 (Detection of (noisy) abrupt change)**.** *Define $\mathcal{H}_0$ and $\mathcal{H}_1$ as in (1). For an algorithm with $\beta_n = \mathcal{O}(\frac{1}{n})$, a slack $\gamma = \frac{12np\beta_{n(1-p)} + 2\sqrt{2\log(4/\delta)/np} + 2\eta}{(1-p)(1+2p)}$, where $p = \frac{|J|}{n}$,*

error is not a random variable of a Bernoulli process.

Algorithm 1: Concept Drift Detection Scheme

```
Input: Algorithm $A$, window size $m$, number of
       permutations $P$, $\Delta$, significance rate $\delta$.
Output: A set of change indices $D$.
Initialization: $t_1 \leftarrow 1$, $k \leftarrow m$, $D \leftarrow \varnothing$
while not end of stream do
    $S_{ord} \leftarrow Z_{t_1}^{k}$, $S'_{ord} \leftarrow Z_{k+1}^{k+m}$
    if detect($A, S_{ord}, S'_{ord}, P, \Delta, \delta$) is True then
        $t_1 \leftarrow k$, $D \leftarrow D \cup \{k\}$.
    $k \leftarrow k + m$
return $D$
Procedure detect($A, S_{ord}, S'_{ord}, P, \Delta, \delta$)
    for $i \leftarrow 1$ to $P$ do
        $(S_i, S_i^{'}) \leftarrow$ random split of $S_{ord} \cup S'_{ord}$
    return TEST($\hat{R}_{ord}, \{\hat{R}_{S_i'}(A_{S_i})\}_{i=1}^{P}, \Delta, \delta$)
```

and any $\delta \in (0,1)$, we have that under $\mathcal{H}_0$, with probability of at least $1-\delta$,

$$\begin{aligned} \left| \hat{R}_{ord} - \hat{R}_{perm} \right| \quad &\leq \quad 6np\beta_{n(1-p)} + \sqrt{2\log(4/\delta)/np} \\ &\qquad + \Delta(1-p)(1+2p) + \eta. \end{aligned}$$

Under $\mathcal{H}_1$, the inverse inequality holds with probability of at least $1-\delta$.

Theorem 3 states a threshold which differentiates between the two hypotheses with high probability, and may be used for the TEST routine in Algorithm 1. The role of each element in the bound is as follows:

1. Test window size $p = \frac{|J|}{n}$: the larger the window is the better concentration of the empirical risk around the risk and consequently, a tighter bound is obtained. In practice, the window size corresponds to the maximum detection delay and therefore one should choose it as large as possible while considering the tolerance to detection delay.
2. The significance rate δ bounds the miss detection and false detection rates, usually set in the range $0.01 - 0.05$.
3. The hyperparameter Δ denotes the rate of change one wishes to ignore. The output of the test for different values of Δ may be computed after training the predictors once, therefore, detection as a function of Δ can be given instead of a single binary value with no further computational cost.
4. The stability rate $\beta_n = \mathcal{O}(\frac{1}{n})$ of the algorithm controls the variability of the loss. For a stable algorithm $np\beta_{n(1-p)} \to 0$ as $n \to \infty$.[3]
5. The rate η bounds the permitted variations of the risk within an interval, and can be empirically estimated in advance. As can be expected, in the simplified scenario of an abrupt change ($\eta = 0$), the bound is tighter.

[3]Examples of stable algorithms are k-Nearest Neighbors, soft margin SVM, SVM regression (SVR), and Regularized Least Squares (Bousquet & Elisseeff, 2002).

2.2. Gradual drift

Another concept drift type is a *gradual drift*. In many real life problems, the change between two concepts happens gradually. For example, in fraud detection, the characteristics of fraudulent actions may have a slow transition phase. Gradual drifts may be modeled by a mixing stage, in which an increasing weight is given to the next concept and a decreasing one to the prior concept.

We treat two cases of gradual drift: one which is faster, where the change length is within the test window, and another, corresponding to a gradual drift evolving very slowly; see Figure 1c for an illustration. In the first case, there is sufficient difference between the average risk in the test window and the previous concept, thus, the abrupt change algorithm may be used. In the second case, the drift is slower, and therefore, the difference between the risks may be insufficient to cross the threshold. When the drift is slow, it is most challenging to detect, however, when it is not detected, it can slowly degrade the quality of the predictor. To treat this drift type we provide a lookahead procedure presented in Algorithm 2. In this procedure, if the hypothesis test determines that there may be a change, we retain the same training window and move to test the next test window. This process is continued until a drift is detected, the test determines a false-alarm, or the maximum number of lookahead iterations have been reached.

We define the null and alternative hypotheses for detecting slow gradual change as follows. The hypotheses are directly defined for the noisy setting, so all stream segments have η-permitted variations w.r.t. f (Definition 1). Let m be the size of the test window, let $I = [1, k]$ be the training window and let $J_q = [k+(q-1)m+1, k+qm]$, for $q = 1, \dots, M$, be the ranges of M test stream segments. The hypotheses at the $M' \leq M$ lookahead iteration are:

$$\begin{aligned} &\mathcal{H}_0 : \forall q=1\dots M' && \left|R_I(f) - R_{J_q}(f)\right| \leq \Delta \qquad (2)\\ &\mathcal{H}_1 : \forall q=1\dots M'-1 && \left|R_I(f) - R_{J_q}(f)\right| \leq \Delta \\ &\quad\text{and} && \left|R_I(f) - R_{J_{M'}}(f)\right| \geq \Delta + \gamma(\eta). \end{aligned}$$

Recall that S_{ord} denotes the training set. Denote the q^{th} test set by $S'_{ord,q} = \{z_t : t \in J_q\}$. Let $\hat{R}_{perm,q}$ be the average empirical risk obtained on permutations of $[S_{ord}, S'_{ord,q}]$. The following theorem bounds the miss detection and false alarm rates of a gradual drift at iteration $M' \leq M$.

Theorem 4 (Detection of slow gradual change). *Define $\mathcal{H}_0$ and $\mathcal{H}_1$ as in (2). For an algorithm with $\beta_n = \mathcal{O}(\frac{1}{n})$, a slack $\gamma = \frac{12np\beta_{n(1-p)} + 2\sqrt{2\log(4M'/\delta)/np} + 2\eta}{(1-p)(1+2p)}$, where $p = \frac{m}{n}$, we have that for any $\delta \in (0,1)$ and $T = 6np\beta_{n(1-p)} + \sqrt{2\log(4M'/\delta)/np} + \Delta(1-p)(1+2p) + \eta$, under $\mathcal{H}_0$, with probability of at least $1-\delta$,*

$$\forall q = 1, ..., M' \quad \left|\hat{R}_{S'_{ord,q}}(f) - \hat{R}_{perm,q}\right| \leq T.$$

Algorithm 2: Slow Gradual Drift Detection Scheme. See Algorithm 1 for `detect` procedure.

```
Input: Algorithm $A$, window size $m$, $P$ permutations,
       maximal lookahead iterations $M$, $\Delta$, warning and
       detection significance rates: $\delta_w$, $\delta_d$.
Output: A set of change indices $D$.
Initialization: $t_1 \leftarrow 1$, $k \leftarrow m$, $D \leftarrow \varnothing$
while not end of stream do
    exit ← False, $q \leftarrow 1$
    while exit is False and $q \leq M$ do
        $S_{ord} \leftarrow Z_{t_1}^{k}$, $S'_{ord} \leftarrow Z_{k+(q-1)m+1}^{k+qm}$
        if detect($A, S_{ord}, S'_{ord}, P, \Delta, \delta_d$) is True
        then
            $t_1 \leftarrow k+1$, $k \leftarrow k+m$, $D \leftarrow D \cup \{k\}$
            exit ← True
        else if detect($A, S_{ord}, S'_{ord}, P, \Delta, \delta_w$) is
        False then
            $k \leftarrow k+m$, exit ← True
        $q \leftarrow q+1$
return $D$
```

Under $\mathcal{H}_1$, with probability of at least $1-\delta$,

$$\max_{q=1\dots M'-1} \left|\hat{R}_{S'_{ord,q}}(f) - \hat{R}_{perm,q}\right| \leq T,$$

and

$$\left|\hat{R}_{S'_{ord,M'}}(f) - \hat{R}_{perm,M'}\right| \geq T.$$

The proof of the theorem relies on the bounds in Theorem 3 and the union bound.

3. Test Procedure

Using Theorems 3 and 4 themselves as testing procedures, with type-1 and type-2 error guarantees provided by the bounds, may result in conservative tests, as the bounds are based on large deviations results. In this section we propose another hypothesis testing procedure for the "TEST" module in Algorithms 1 and 2. The theorems imply that variability of the loss, encapsulated in the error stability of the algorithm, should be taken into account in the test procedure. In this section, we present a different perspective on these variations using *permutation tests* (Good, 2005). Permutation tests are a well known methodology in statistics to obtain most powerful statistical procedures, and will provide tighter thresholds for the testing procedure.

Our test statistic is $\hat{R}_{ord}$, the loss over the ordered train-test split. To test if a change has taken place, we use a permutation test. The heart of the method is that under the null hypothesis the samples are exchangeable and therefore may be randomly shuffled. According to the null hypothesis, the average risk before and after the tested change index k are Δ-close. Under this hypothesis, the samples are

Algorithm 3: TEST $(\hat{R}_{ord}, \left\{\hat{R}_{S_i'}(A_{S_i})\right\}_{i=1}^{P}, \Delta, \delta)$ Procedure for detection scheme

```
Input: $\hat{R}_{ord}$, $\left\{\hat{R}_{S_i'}(A_{S_i})\right\}_{i=1}^{P}$, $\Delta$, significance rate $\delta$
if $\frac{1+\sum_{i=1}^{P} 1[\hat{R}_{ord}-\hat{R}_{S_i'}(A_{S_i})\leq\Delta]}{P+1} \leq \delta$ then
    detection ← True
else detection ← False
return detection
```

approximately exchangeable. Specifically, the more stable the algorithm is and the smaller Δ is, the more the examples are exchangeable. As the size of the training set grows, which emanates when a change is not detected, stability of the learner strengthens and ensures exchangeability.

By shuffling the data multiple times we can obtain an estimate of the non-parametric distribution of the test statistic $\hat{R}_{ord}$ under the null hypothesis. Recall that $\hat{R}_{S_i'}(A_{S_i})$ is the empirical error obtained for the i^{th} shuffled split. If there was a concept drift at time index k, we expect $\hat{R}_{ord}$ to be larger than if $\mathcal{H}_0$ was true. The larger value we observe, the stronger is the evidence against $\mathcal{H}_0$. Therefore, the achieved significance level of the procedure is $\mathbb{P}_{S_i,S_i'}\left[\hat{R}_{ord} - \hat{R}_{S_i'}(A_{S_i}) \leq \Delta \middle| \mathcal{H}_0\right]$. Algorithm 3 defines this testing procedure. Optimally, to get the exact distribution of the statistic, we would need to examine all possible train-test shuffles, but in practice much less are sufficient. Also note that the train-test procedure of the multiple splits can be straightforwardly parallelized.

In the gradual drift detection scheme, at iteration M', M' base hypotheses are combined, and therefore, there is a need to correct for multiple comparisons. In Theorem 4, this is done by the union bound, which is equivalent to the Bonferroni correction in the statistical literature. In the permutation test, we use the Benjamini-Hochberg correction (Benjamini & Hochberg, 1995), which is less conservative.

4. Experiments

We compare the performance of our detection algorithm in classification and regression settings with other concept-drift detection methods, and baseline methods. We use synthetic data and drifts synthesized in real data, thus controlling drift points and allowing precise performance analysis.

Detection Algorithms: We denote by *PERM* and *grad-PERM* the application of Algorithm 1 and 2, respectively, applied with the testing procedure in Algorithm 3. In classification problems, we compare PERM's detection to three known methods for concept drift detection: *DDM* (Gama et al., 2004), early drift detection (*EDDM*) (Baena-García et al., 2006), and *STEPD* (Nishida & Yamauchi, 2007); see Section 1.1. We also compare the performance with learners with a fixed memory window (*Window #*, where # is the window size), a full memory that trains on all prior examples (*No detection*), and an "optimal" detector in hindsight that detects a drift at its onset (*Exact detection*). We set the sensitivity level of PERM and grad-PERM to $\delta = 0.01, \Delta = 0$, the warning and detection thresholds of STEPD to $w = 0.05, d = 0.01$, and the parameters of EDDM to $\alpha = 0.95$ and $\beta = 0.90$.[4]

Performance Measures: Performance comparisons are done by evaluating detection quality and error rate. A True Positive (TP) detection is defined as a detection within a fixed delay range after the precise concept change time. This range is taken to be the size of the window in PERM and STEPD. 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 measured both by the Recall $= TP/(TP+FN)$ and Precision $= TP/(TP+FP)$ of the detector. The prediction on a test example is done by a predictor trained on the previous examples.

4.1. Synthetic Data

In the following synthetic classification tasks the base algorithm was K-Nearest Neighbors ($k = 3$), each stream was randomly repeated 100 times, and $P = 100$ reshuffling splits were used in PERM. The first stream, denoted "Mixed", represents abrupt drift with label noise. The features are two boolean (v, w) and two numeric (x, y) attributes; the examples are labeled as positive if at least two conditions are met: $v, w = 0, y < 0.5 + 0.3\sin(3\pi x)$. The classification is reversed at each concept change. The second dataset, denoted "Circles", presents a more challenging concept drift with label noise. We sample data uniformly from the unit square, and label an example as positive/negative if it resides inside/outside a moving and dynamically changing circle. We set the initial center and direction of movement at random, and at each concept drift gradually move the center and change the radius.[5] The third, most challenging synthetic dataset, is the rotating "Checkerboard"(Elwell & Polikar, 2011), 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 $\pi/20$ radians.

[4]These hyper parameters were not optimized. The parameters δ and d correspond 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.

[5]When the radius is maximal (0.45), the circle shrinks (until it reaches 0.15) and its direction changes.

Table 1. Checkerboard dataset - gradual drift (100 repetitions).

Algorithm	detection delay	#runs/100 no detection
PERM	241.0±113.2	0
grad-PERM	**137.2±115.7**	0
DDM	569.6±54.7	0
EDDM	494.0±77.7	4
STEPD	455.3±106.8	15

The results on the different streams, each consisting of 10 drifts and concept lengths of 1000 samples, are presented in Figure 2: rows correspond to datasets and columns to performance measures. Both recall and precision of the PERM algorithm are higher than the other methods. The performance difference of the error-noise curves is smaller on the Mixed and Circle datasets. These two problems are not hard, and therefore, while DDM and EDDM encounter many miss detections, their error rate remains low. In the checkerboard dataset, the performance of PERM improves as the window range grows. This can be anticipated because as the test size grows the variations of the error decreases, which reduces the false detection rate.

Our last synthetic drift was generated using the Checkerboard dataset. In this experiment, a single gradual drift is initiated by increasing the label noise at time-indices $[1400, 1600, 1800, 2000]$ with corresponding noise-rates $[0.05, 0.1, 0.2, 0.3]$. The growing label noise serves as a gradual drift which slowly evolves. The window size in PERM, grad-PERM and STEPD was 100 samples. Table 1 shows that grad-PERM obtained the best detection rate.

4.2. User Preference Prediction

We compare detection performance on a user preference prediction task defined using the 20-news groups text dataset[6], consisting of $18,846$ documents and over $75,000$ features. We apply the TF-IDF weighting scheme on the documents and partition the 20 groups to six subgroups according to subject as presented in the repository. We define two prediction problems: a binary classification problem of identifying whether a user likes/dislikes the subject, and a regression problem of rating user preference. In all our experiments the time-series is generated by randomly reordering all the documents. We use $P = 500$, and SVM and SVR with linear kernel as the learning algorithms.[7]

We ran two experiments on the classification problem. In the first, after 1000 documents (∼50 documents from each newsgroup) a concept drift is activated, in which a random

[6]http://qwone.com/~jason/20Newsgroups

[7]We used scikit-learn: Machine Learning in Python toolbox. Cross-validation on a single random concept showed low sensitivity to the choice of C on this dataset and therefore the default value $C = 1$ was chosen.

Table 2. 20 Newsgroup - Binary classification (50 repetitions). Detection of 10 drift points in a range of 100 samples after drift onset. Detection error for exact windowing is $\mathbf{.19 \pm .01}$.

Algorithm	Recall	Precision	Error
PERM	**0.95±.06**	**0.92±.075**	**.19±.01**
DDM	0.67±.18	0.24±.06	.21±.01
EDDM	0.70±.14	0.25±.08	.23±.02
STEPD	0.51±0.19	0.51±0.19	.21±.01
Window 500	-	-	.20±.01
Window 1000	-	-	.23±.02
No Detection	-	-	.38±.04

Table 3. 20 Newsgroup - Binary classification with gradual drift (100 repetitions).

Algorithm	detection delay	#runs/100 no detection	FA before drift
PERM	245±172	14	**0.15±0.3**
grad-PERM	**169±160**	**5**	**0.16±0.3**
DDM	321±140	42	0.75±0.7
EDDM	**188±155**	26	1.6±1.3
STEPD	358±199	62	0.6±0.5

third of the binary ratings are switched and the rest remain as in the previous concept.[8] In the second, we use the same concepts but with a gradual drift: after 500 documents from the first concept, sets of 100 documents are added with mixing proportions $[0.8|0.2, 0.6|0.4, 0.4|0.6, 0.2|0.8]$ from the first and second concepts respectively. Another 500 documents from the second concept follow.

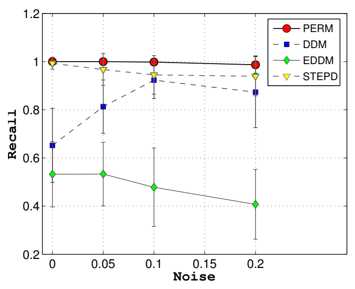
In the regression problem, the label is initiated randomly to $1 - 10$ and a concept drift is activated after 2000 documents. Two types of drifts are considered: (I) the label of all six groups undergo a random walk step ± 3, (II) four groups randomly selected change rating of ± 3. Scenario (II) is more challenging as the drift is smaller. Performance of PERM is compared only to that of “Window”, “No detection”, and “Exact detection”, since DDM, EDDM and STEPD are limited to zero-one classification problems.

The results are presented in Tables 2 to 4. The performance of PERM for regular drift and grad-PERM for gradual drift, are the highest in both classification and regression.

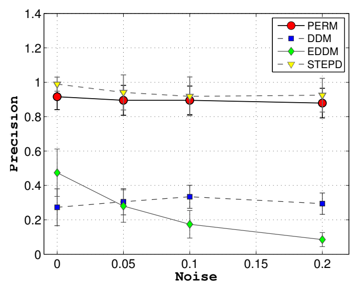
5. Discussion

We presented a resampling scheme for concept-drift detection with respect to the risk of a learning algorithm. One of the advantages of the method is its applicability to any stable learning algorithm and any bounded loss function chosen as the most suitable for the task at hand. For example, weighted loss may be used in a data imbalance scenario.

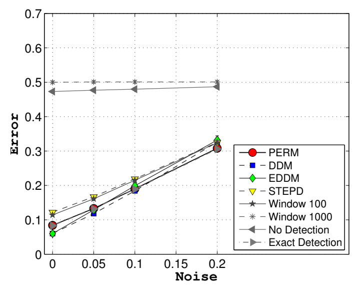
[8]If at some point, a concept has all positive or all negative labeling, the concept is reinitialized randomly.



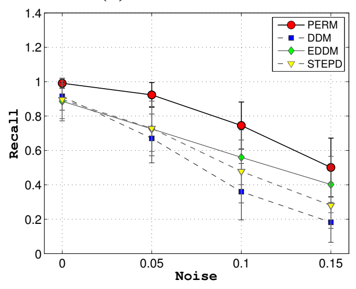
(a) Mixed: Recall-Noise



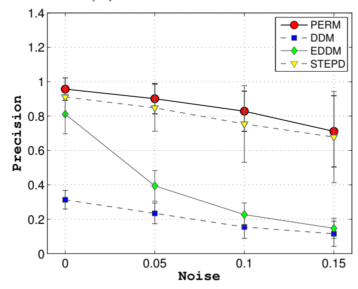
(b) Mixed: Precision-Noise



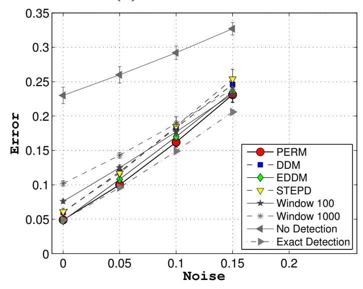
(c) Mixed: Error-Noise



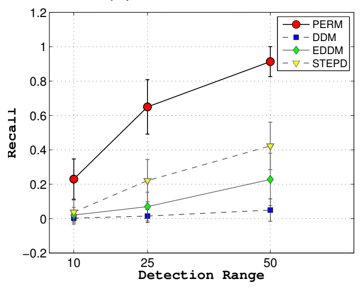
(d) Circle: Recall-Noise



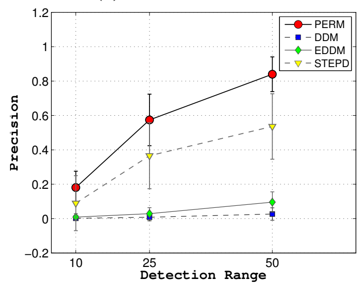
(e) Circle: Precision-Noise



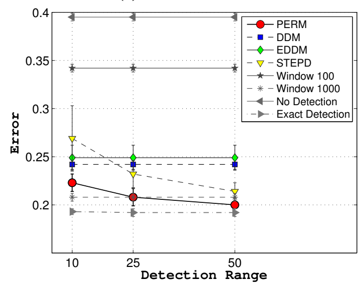
(f) Circle: Error-Noise



(g) Checkerboard:Recall-Range



(h) Checkerboard:Precision-Range



(i) Checkerboard:Error-Range

Figure 2. Synthetic datasets. Figures 2a to 2c - Mixed dataset with detection range of 50 samples. Figures 2d to 2f - Circle dataset with detection range of 100 samples. A larger range was chosen to compensate for the class imbalance in this problem. Figures 2g to 2i - Checkerboard dataset with different detection ranges.

Table 4. 20 Newsgroup-Regression (50 repetitions). Detection of 5 concept drifts, with detection range of 200 samples.

(a) Detection Rate of PERM

	Recall	Precision
(I)	0.93±.12	0.93±.10
(II)	0.87±.17	0.89±.11

(b) Error Rate. Exact detection: (I) $\mathbf{1.46}\pm\mathbf{.16}$, (II) $\mathbf{1.48}\pm\mathbf{.19}$.

	PERM	Window 500|1000	No Detection
(I)	$\mathbf{1.50}\pm\mathbf{.17}$	1.60±.17, 1.58±.13	2.30±.13
(II)	$\mathbf{1.47}\pm\mathbf{.19}$	1.59±.22, 1.50±.19	2.14±.21

Our detection algorithm outputs time indices of concept changes that form windows of adaptive size, each comprising of examples from a single concept. The detection times may be used for initiating a new training phase, as done in our experimental evaluation, but can also be used as a basis for ensemble learners, by providing an informed choice of the training windows for the ensemble's members.

Our experiments show that the proposed scheme is more robust to noise and has better precision and recall rates than existing schemes. There is a bias-variance tradeoff in the choice of window size in PERM. A larger window reduces the variability which increases the accuracy of the test, but causes a detection delay which may increase the error.

In future work we plan to implement an online setting for the scheme, in which the learners of the reshuffled samples are preserved and updated. While the current algorithm is readily parallelized, this regime should enjoy a favorable computation time.

Acknowledgments

This research was partially supported by the ISF under contract 890015 and by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI).

References

Baena-García, M., Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavaldá, R., and Morales-Bueno, R. Early drift detection method. In *StreamKDD*, pp. 77–86, 2006.

Basseville, M. and Nikiforov, I. V. *Detection of Abrupt Changes: Theory and Application*. Prentice-Hall, 1993.

Benjamini, Y. and Hochberg, Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *JRSS*, pp. 289–300, 1995.

Bifet, A. and Gavalda, R. Learning from time-changing data with adaptive windowing. In *ICDM*, 2007.

Black, M. and Hickey, R.J. Maintaining the performance of a learned classifier under concept drift. *IDA*, 1999.

Bousquet, O. and Elisseeff, A. Stability and generalization. *JMLR*, 2:499–526, 2002.

Brodsky, B.E. and Darkhovsky, B.S. *Nonparametric methods in change-point problems*. Springer, 1993.

Chen, Jie and Gupta, A Arjun K. *Parametric Statistical Change Point Analysis: With Applications to Genetics, Medicine, and Finance*. Springer, 2012.

Dasu, T., Krishnan, S., Venkatasubramanian, S., and Yi, K. An information-theoretic approach to detecting changes in multi-dimensional data streams. In *ISCSA*, 2006.

Desobry, F., Davy, M., and Doncarli, C. An online kernel change detection algorithm. *IEEE Trans. Signal Process*, 53(8):2961–2974, 2005.

Elwell, R. and Polikar, R. Incremental learning of concept drift in nonstationary environments. *IEEE Trans. Neural Netw.*, 22(10):1517–1531, 2011.

Gama, J., Medas, P., Castillo, G., and Rodrigues, P. Learning with drift detection. In *Advances in Artificial Intelligence–SBIA 2004*, pp. 286–295. 2004.

Good, P.I. *Permutation, parametric and bootstrap tests of hypotheses*. Springer, 2005.

Gustafsson, F. The marginalized likelihood ratio test for detecting abrupt changes. *IEEE Trans. Autom. Control*, 41(1):66–78, 1996.

Hulten, G., Spencer, L., and Domingos, P. Mining time-changing data streams. In *KDD*, 2001.

Kawahara, Y. and Sugiyama, M. Sequential change-point detection based on direct density-ratio estimation. *Statistical Analysis and Data Mining*, 5(2):114–127, 2012.

Kawahara, Y., Yairi, T., and Machida, K. Change-point detection in time-series data based on subspace identification. In *ICDM*, pp. 559–564, 2007.

Kearns, M. and Ron, D. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Neural Computation*, 11(6):1427–1453, 1999.

Kifer, D., Ben-David, S., and Gehrke, J. Detecting change in data streams. In *VLDB*, pp. 180–191, 2004.

Klinkenberg, R. Concept drift and the importance of examples. In *Text Mining TAA*, pp. 55–77, 2003.

Klinkenberg, R. Learning drifting concepts: Example selection vs. example weighting. *IDA*, 8:281–300, 2004.

Klinkenberg, R. and Joachims, T. Detecting concept drift with support vector machines. In *ICML*, 2000.

Kolter, J.Z. and Maloof, M.A. Dynamic weighted majority: An ensemble method for drifting concepts. *JMLR*, 8: 2755–2790, 2007.

Lavielle, M. and Teyssiere, G. Detection of multiple change-points in multivariate time series. *Lithuanian Mathematical Journal*, 46(3):287–306, 2006.

Liu, S., Yamada, M., Collier, N., and Sugiyama, M. Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks*, 2013.

Lung-Yut-Fong, A., Lévy-Leduc, C., and Cappé, O. Homogeneity and change-point detection tests for multivariate data using rank statistics. *arXiv:1107.1971*, 2011.

Masud, M., Gao, J., Khan, L., Han, J., and Thuraisingham, B. A multi-partition multi-chunk ensemble technique to classify concept-drifting data streams. In *KDD*. 2009.

Moskvina, V. and Zhigljavsky, A. An algorithm based on singular spectrum analysis for change-point detection. *COMMUN STAT SIMULAT*, 32(2):319–352, 2003.

Nishida, K. and Yamauchi, K. Detecting concept drift using statistical testing. In *DS*, pp. 264–269, 2007.

Sebastião, R. and Gama, J. Change detection in learning histograms from data streams. In *PAI*. Springer, 2007.

Stanley, K.O. Learning concept drift with a committee of decision trees. *Informe técnico*, 2003.

Street, W. and Kim, Y. A streaming ensemble algorithm (sea) for large-scale classification. In *KDD*, 2001.

Takeuchi, J. and Yamanishi, K. A unifying framework for detecting outliers and change points from time series. *IEEE Trans. Knowl. Data Eng*, 18(4):482–492, 2006.

Tsymbal, A. The problem of concept drift: definitions and related work. Technical report, 2004.

Vapnik, V.N. *Statistical learning theory*. Wiley, 1998.

Vovk, V., Nouretdinov, I., and Gammerman, A. Testing exchangeability on-line. In *ICML*, pp. 768–775, 2003.

Wang, H., Fan, W. Yu, P., and Han, J. Mining concept drifting streams using ensemble classifiers. In *KDD*, 2003.