A Pdf-free Change Detection Test for Data Streams Monitoring

Li Bu, Dongbin Zhao
The State Key Laboratory of Management and Control for Complex Systems
Institute of Automation, Chinese Academy of Sciences, Beijing, China
Email: bulipolly@gmail.com, dongbin.zhao@ia.ac.cn

Cesare Alippi
Politecnico di Milano
Milano, Italy
Email: cesare.alippi@polimi.it

***Abstract*—We experience changes in stationarity/time variance in many practical applications. Since changes modify the operational framework the application is working with, its accuracy performance is in turn affected. When changes can occur, we need to detect them as soon as possible, in general by inspecting features extracted from data, and afterwards intervene to mitigate their effects. In this paper, we propose a novel change detection test based on the least squares density difference estimation. Neither assumptions about the distribution of features are needed, nor the change types are made (the method is pdf-free and can handle arbitrary changes.). What here proposed requires limited data to become operational and thresholds needed to assess the change can be set met to predefined false positive rates. We show through comprehensive experiments the effectiveness of the detection method and point out how it outperforms other related methods.**

I. Introduction

In traditional learning, the stationary hypothesis is always assumed implying that the data generating process does never change. However aging phenomena affecting sensors, time variance of the environment and faults lead to changes in the process generating the data and, as a consequence, deviations from nominal, error-free values in turn, changes affect the normal operation of the application and its performance.

After having extracted features, we should look at changes in the probability density function (pdf) to detect the occurrence of a change. The literature presents many methods monitoring some features of the pdf, e.g., mean or variance. For instance, Ross et al. [1] [2] propose several methods to allow for nonparametric change detection in non-Gaussian sequences: nonparametric hypothesis tests are combined to create a sequential change-point model (CPM) [1], while [2] presents two control charts to detect arbitrary changes when the distribution is unknown. Raza et al. propose an EWMA-based method [3] to monitor auto-correlated observations without delay which is suitable for real-time adaptive classification problems [4]. Alippi et al. put forward a Just-in-time framework in [5] [6] to adapt models only when needed (changes are detected). The proposed CI-CUSUM test [5] and ICI-based change detection test(CDT) [7] can detect trends and drifts, with due the thresholds learned from a training set. Schilling proposes a KNN-based test [8] to measure the proportion of all k nearest neighbor comparisons in which a point and its neighbor are members from the same set. The statistic explicitly satisfies a limiting normal distribution under some mild requirements, and is compared with a predefined threshold.

However, the above methods either were not designed for multi-dimensional features, or cannot control the rate of false positives (FPs). Clearly, having the possibility to compare the pdfs directly at the data streams(or features) level would be the preferable method. However, it would suffer from the fact that mostly, we have a limited dataset, and it is hard to obtain accurate estimates for the pdf. A different solution is proposed by Sugiyama et al., which uses a linear model to directly estimate the density-ratio [9] [10] or the density-difference [11] [12] of two subsets. These methods overcome the drawback of traditional two-step procedures requesting to estimate two densities separately and then computing their difference in order to detect a change, a situation that propagates the estimation error. The values of density-ratio and density-difference represent the dissimilarity of the two pdfs with high values implying a large difference. However, neither a reasonable threshold is proposed there to detect a change, nor the FP rate is considered in their work. Another drawback is that the estimated values change a lot, in particular in correspondence with limited data sets, which makes it impossible in those papers to derive an effective threshold.

By investigating the least squares density-difference estimation (LSDD) [12] method, we discovered some interesting properties. In particular, the distribution of LSDD values of a stationary dataset is associated with an asymmetric distribution, and the values of nonstationary dataset will exceed the distribution limits. Although the subset size has a strong impact on LSDD values, the difference between LSDD values from pre and post-change samples with a small size is still significant. This implies that we can detect changes even with a small dataset.

In this paper, we propose a pdf-free change detection test to monitor data streams based on LSDD and investigate resulting properties. The method can be used in multi-dimensional datasets. A bootstrapping procedure has been considered to extract the LSDD values in scenarios encompassing small datasets. By fitting the collected values with a Gamma distribution that shows to be appropriate in modeling the LSDD values, thresholds can be easily obtained according to

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.86

predefined FP rates. The carried out experiments validate the effectiveness of our method.

The structure of the paper is as follows. Section II briefly introduces the LSDD method. Section III shows a detailed description of our proposed LSDD-based CDT (LSDD-CDT) method. The experimental section IV validates the effectiveness of our method, and gives a comprehensive comparison with other related methods on benchmarks. The conclusion is given in Section V.

II. LSDD METHOD

Density-difference estimation between two probability densities has been proposed in [12] to measure the least squares density-difference:

$$D^2(p,q) = \int (p(x) - q(x))^2 dx, \tag{1}$$

where $x \in R^d$ is a real vector, and $p(x), q(x)$ are two probability density functions. Kernel density estimation [13] with Gaussian kernel is used to model $p(x) - q(x)$ without strong assumptions:

$$g(x,\Theta) = \sum_{i=1}^{2n} \theta_i \exp(-\frac{||x - c_i||_2^2}{2\sigma^2}) \tag{2}$$

$$\Theta=(\theta_1,...\theta_i,...\theta_{2n})$$

where $(c_1, ..., c_n, c_{n+1}, ..., c_{2n}) = (x_{p,1}, .., x_{p,n}, x_{q,1}, ..., x_{q,n})$ are d-dimensional kernel centers, n is the subset size, Θ is a parameter vector, and σ is the kernel width, e.g, empirically determined as the median distance between points in the aggregate sample [14] [15]: $\sigma = median(||x_i - x_j||_2, 0 < i < j \leq 2n)$. The optimal parameter Θ^* is achieved by minimizing the squared loss $J(\Theta)$:

$$J(\Theta) = \int (g(x,\Theta) - (p(x) - q(x)))^2 dx \tag{3}$$

Adding a $L2$-regularizer λ to the objective function (3) to avoid over-fitting, the optimization problem transformes to:

$$\begin{aligned} \hat{\Theta} &= \arg\min_{\Theta} (J(\Theta) + \lambda \Theta^T \Theta) \\ &= \arg\min_{\Theta} (\Theta^T H \Theta - 2h^T \Theta + \lambda \Theta^T \Theta) \end{aligned} \tag{4}$$

where $\lambda \geq 0$, H is a $2n \times 2n$ matrix, and h is a $2n \times 1$ vector defined as:

$$\begin{aligned} H_{i,j} &= \int \exp(-\frac{||x - c_i||_2^2}{2\sigma^2}) \exp(-\frac{||x - c_j||_2^2}{2\sigma^2}) dx \\ &= (\pi\sigma^2)^{d/2} \exp(-\frac{||c_i - c_j||_2^2}{4\sigma^2}) \end{aligned} \tag{5}$$

$$\begin{aligned} h_i = &\int \exp(-\frac{||x - c_i||_2^2}{2\sigma^2}) p(x) dx \\ &- \int \exp(-\frac{||x - c_i||_2^2}{2\sigma^2}) q(x) dx \end{aligned} \tag{6}$$

$i, j = 1, ..., n, n+1, ...2n$. Since $p(x), q(x)$ are unknown, an estimate $\hat{h}_i$ is used:

$$\begin{aligned} \hat{h}_i = &\frac{1}{n} \sum_{j=1}^{n} \exp(-\frac{||x_{p,j} - c_i||_2^2}{2\sigma^2}) \\ &- \frac{1}{n} \sum_{j=1}^{n} \exp(-\frac{||x_{q,j} - c_i||_2^2}{2\sigma^2}) \end{aligned} \tag{7}$$

Thus, $\hat{\Theta}$ can be expressed as:

$$\hat{\Theta} = (H + \lambda I)^{-1} \hat{h} \tag{8}$$

By replacing $p(x) - q(x)$ with $g(x, \hat{\Theta})$, the D^2 -distance can be estimated by two equivalent expressions $\hat{D}^2(p,q) \approx \hat{h}^T \hat{\Theta}$ and $\hat{D}^2(p,q) \approx \hat{\Theta}^T H \hat{\Theta}$. [11] combines them together to reduce the bias brought by λ:

$$\hat{D}^2(p,q) = 2\hat{h}^T \hat{\Theta} - \hat{\Theta}^T H \hat{\Theta}$$

The quality of over-fitting depends on the regularization parameter λ. [11] [12] suggest to find the optimal parameters by cross validation when comparing two different distributions. Whenever we compare two subsets drawn from a stationary distribution, the estimated densities are similar, and the density-difference is so small that a larger λ corresponding to smoother fitting is always preferred. However in this case, the LSDD values associated with the transition stationary to non-stationary will not be close to the real ones. In this paper, we propose a method to select the parameter λ by controlling the relative difference defined as:

$$RD = \frac{\hat{h}^T \hat{\Theta} - \hat{\Theta}^T H \hat{\Theta}}{\hat{h}^T \hat{\Theta}} = 1 - \frac{\hat{\Theta}^T H \hat{\Theta}}{\hat{h}^T \hat{\Theta}}$$

Expanding $(H + \lambda I)^{-1}$ at $\lambda = 0$ by Taylor expansion, $\hat{h}^T \hat{\Theta}$ and $\hat{\Theta}^T H \hat{\Theta}$ can be expressed as:

$$\hat{h}^T \hat{\Theta} = \hat{h}^T H^{-1} \hat{h} + f_1(\lambda),$$

$$\hat{\Theta}^T H^{-1} \hat{\Theta} = \hat{h}^T H^{-1} \hat{h} + f_2(\lambda),$$

where $f_1(\lambda)$, $f_2(\lambda)$ are the higher order terms. When $\lambda = 0$, $\hat{D}^2 = \hat{h}^T \hat{\Theta} = \hat{\Theta}^T H^{-1} \hat{\Theta} = \hat{h}^T H^{-1} \hat{h}$; when $\lambda > 0$, the ratio between the two expressions $\frac{\hat{h}^T \hat{\Theta}}{\hat{\Theta}^T H^{-1} \hat{\Theta}}$ changes with λ, and is above 1, i.e. the relative difference RD changes. Thus, we offer some alternatives of λ during training, and choose the one which controls the relative difference RD smaller than a given constant.

III. LSDD-CDT

By inspecting changes in the LSDD we can immediately detect a change in the pdfs. However, $\hat{D}^2$ values are strongly dependent on the particular realization of samples as well as the particular distribution. That is, we can't give a fitting function or a fixed constant associated with thresholds suitable for any situation.

Obviously, the distribution of $\hat{D}^2$ in the stationary case satisfies an unknown but fixed distribution, whereas the values associated with a change in stationarity will not satisfy this distribution. Since there is no direct and theoretical basis

indicating which distribution should be considered to fit the L^2-norm dissimilarity of two densities, in this paper Gamma distribution is considered because of its proven effectiveness in extensive experiments.

A. Generating LSDD Values with Bootstrapping

Having a small training set allows the method for becoming immediately operational even with few available data, and at the same time, the computational cost is kept under control. We consider using bootstrap to generate a bootstrap-based distribution as suggested in [16] from the small training set. Two subsets are sampled repeatedly with bootstrap from the training set with size N_t, and each of them with n samples works as a reference Z_p and a testing set Z_q respectively. Then, the LSDD value y_i is calculated by comparing Z_p and Z_q as a sample from $\hat{D}^2$ distribution.

In order to verify the feasibility of using bootstrap, we also extracted a very large dataset with values extracted according to the same distribution. In this case, the computed $\hat{D}^2$ values are treated as being extracted from the "real" distribution of LSDD values. The estimated distribution of $\hat{D}^2$ values with bootstrap will be compared with this "real" distribution in subsection IV-A.

B. Estimating Thresholds with Predefined FP Rates

In [17], a practical method based on the Wilson-Hilferty normal approximation is provided to estimate the upper tolerance limits of a Gamma distribution directly on collected samples.

Assuming a series of $\hat{D}^2$ values $y_1, ..., y_m$ extracted from the training set as per sample of a Gamma distribution, let

$$\bar{Y} = \frac{1}{m}\sum_{i=1}^{m} y_i^{1/3},$$

$$S^2 = \frac{1}{m-1}\sum_{i=1}^{m} (y_i^{1/3} - \bar{Y})^2.$$

The upper tolerance limit U_{tl} [17] is given as:

$$U_{tl} = (\bar{Y} + tl \cdot S)^3, tl = \frac{1}{\sqrt{m}} t_{m-1,\gamma,z_p\sqrt{m}} \tag{9}$$

where z_p is the p-th quantile of the standard normal distribution and $t_{m-1,\gamma,z_p\sqrt{m}}$ denotes the γ-th quantile of a noncentral t-distribution with $m-1$ degrees of freedom and noncentrality parameter $z_p\sqrt{m}$. With the preset confidence level γ, U_{tl} is only determined by p.

As recommended in [18], the selected threshold should better ensure that the FP rate $\mu = 1 - p$ is controlled which satisfies

$$P(\hat{D}^2 > U_{tl}) = \mu. \tag{10}$$

The corresponding value of ARL0 (average run length, which denotes the average number of observations between false detections whereas there is no change) is $1/\mu$. Thus given a FP rate or ARL0, the upper tolerance limit U_{tl} is only determined by (10) as a threshold T.

C. The LSDD-CDT Algorithm

To detect changes in data streams where samples are continuously produced, LSDD can be applied sequentially to compare the density difference between two fixed sliding windows. During the testing phase, the left window Z_p with pdf $p(x)$ includes samples arrived earlier and confirmed without any changes as a reference, and the right window Z_q with pdf $q(x)$ contains the new samples treated as a testing set. As time passes, both Z_p and Z_q slide away to collect new samples; the oldest ones are discarded. Then a series of $\hat{D}^2$ values $y_1, ..., y_i, ...$ is generated by estimating the least squares density difference of continuously updated Z_p and Z_q. A change is detected only when a $\hat{D}^2$ value is greater than the threshold T associated with a predefined FP rate μ.

Algorithm 1 LSDD-CDT

Input: training set with N_t samples, window size $n(n < N_t/2)$, FP rates $\mu_c < \mu_w < \mu_s$, confidence level γ, sample time m;
Output: A series of $\hat{D}^2$ values $y_1, ..., y_i, ...$, detection results: no changes / (changes, location).
Generate LSDD values $\hat{D}^2$ on the training set with bootstrap and calculate the parameters σ and λ;
Fit $\hat{D}^2$ values with a Gamma distribution, and calculate the thresholds T_W, T_C, T_S according to predefined FP rates μ_w, μ_c, μ_s respectively based on (9)(10);
Prepare samples for right and left windows; $i = 1$;
while (1) **do**
  Calculate the LSDD value $y_i$of two subsets Z_p, Z_q;
  if $y_i > T_W$ or during a "warning" state **then**
    Set/keep the warning alarm; Stop updating Z_p;
    if $y_i > T_C$ **then**
      Detection results: change, point P_C;
      Collect new samples, retrain models by steps 2-4;
    end if
    if $y_i < T_S$ or the "warning" state lasts exceeding n samples **then**
      Clear the warning alarm;
      Update Z_p .
    end if
  else
    Update Z_p, Z_q with slide strategy;
    $i = i + 1$;
  end if
end while

In order to be more sensitive to small changes under the same FP rate, a three-level threshold mechanism is adopted to reduce the FN rate. It contains a warning threshold (T_W), a change threshold (T_C) and a safe threshold (T_S) which are determined by different FP rates respectively as in (10) to give a warning, confirm a change or clear a warning. When small perturbations happen, i.e., the $\hat{D}^2$ value exceeds T_W which corresponds to a high FP rate μ_w, a "warning" state is initiated at point P_W. The right window Z_q slides to collect new samples to further determine whether there is a change,

or it is a false alarm. Meanwhile, the left widow Z_p stops updating to avoid the influence of possible changes. If the $\hat{D}^2$ value goes over T_C further which corresponds to a low FP rate μ_c, change is confirmed at point P_C. If the $\hat{D}^2$ distance falls back below T_S which is related to a higher FP rate μ_s than T_W, or the "warning" state continues exceeding n samples, the warning alarm is cleared and considered as a false alarm. The two windows continue to slide or update as normal.

The detailed procedure is summarized in Algorithm 1. We take a simple reaction (step 12) as an example to show how it deals with datastreams, and it can be replaced by any effective strategy in real applications [22]. Moreover, the computational complexity is $O(N \cdot n^2 \cdot d)$ where N is the size of dataset.

IV. EXPERIMENTS

To validate the effectiveness of the proposed LSDD-CDT method, we provide a comprehensive comparison with three other related methods on six different applications.

The three methods are KNN-based test [8] [19], H-ICI CDT [20] and CPM tests [1] [2]. KNN-based test aims at monitoring the statistic $T_{k,n}$, which indicates how close the two distributions are. For a fair comparison, it uses the same training and detection strategy as the LSDD-CDT. Since a larger parameter k will bring a higher FP rate and a lower FN rate, we set $k = 7$ to obtain a similar FN rate as our method. H-ICI CDT is a two-level hierarchical CDT, whose first level uses the ICI-based CDT [7], and the second one uses the Hotellings T-square statistic[21]. CPM-LP [1] and CPM-CvM [2] are two CPM methods designed to detect arbitrary changes.

There are six applications which are simulated and coupled with different distributions or different changes. 2000 samples are contained in each application and a different change happens at point 1000. Applications D3-6 are well-known multidimensional benchmarks for assessing detection performance.

- Applications D1-2 refer to a Normal distribution. Parameters change from (0, 0.5) to (0.2 0.5) and (0.3, 0.8), where the two values in each bracket are the mean and the standard variance of the distribution.
- Application D3 is a 10-dimension Normal distribution suggested in [23] which aims at evaluating the detection performance dealing with multi-dimension applications.
- Application D4 is STAGGER (used in [24]) which is 3-dimensional. Since we do not focus on the "reaction" aspect here, we change this classification problem into a detection problem by taking only one class of samples. 1000 samples respectively with Concept1 and then Concept2 are generated.
- Application D5 is a circle problem [25] with dataset satisfying $(x-a)^2 + (y-b)^2 \leq r^2$; and changes affect the radius r. In this paper, $a = b = 0.5, r = 0.2 \rightarrow 0.3$, 10% noise is added and the ranges of x, y are [0,1].
- Application D6 is a moving hyperplane problem $y \leq -a_0 + \sum_{i=1}^{d} a_i x_i$, and changes happen by changing a_0. $a_1 = a_2 = 0.1$, $a_0 = -1 \rightarrow -3.2$, 10% noises are added, and the ranges of x_i, y are [0, 1] and [0, 5] respectively as suggested in [25].

The setting parameters are fixed as follows. The size of training set N_t is 400, confidence level γ is 0.99, and sample time m is 2000. The FP rates μ_s, μ_w, μ_c corresponding to T_S, T_W and T_C respectively are set to 2%, 1% and 0.1%, i.e. the respective values of ARL0 are 50, 100 and 1000. The relative difference RD of two $\hat{D}^2$ expressions is set to 0.25. 500 trials are operated for each application.

These settings are appropriate for LSDD-CDT and KNN-based test since they follow the same detection procedure. The H-ICI CDT retains the same settings as in [20]. The ARL0 values of CPMs are set to 1000, i.e., the predefined FP rates are 0.1% as μ_c.

The following four indices are introduced to evaluate the performance of our algorithm:

- False positives (FPs): it counts the times that a change is detected while there is no change. In D1-6, a FP means there is a "change" detected before point 1000. At each trial, the "change" times will be recorded, and then we sum the times on the 500 trials.
- False negatives (FNs): it counts the times that no changes are detected while there are. In D1-6, a FN means no changes are detected after point 1000. At each trial, if no change is detected, a FN is recorded, and we sum the FNs on the 500 trials.
- Delay (sample): it measures the promptness by considering the detection delay. The value of $P_C - 1000$ is recorded when $P_C > 1000$ at each trial, and both the mean and the standard deviation of the delay values are calculated on the 500 trials.
- Computational time (CT (s)): it shows the execution time taken to perform the tests (reference platform: ThinkCentre M4300t Intel i5 core running @ 3.1GHz, 4G RAM). Both the mean and the standard deviation of the delay values are calculated on the 500 trials.

A. Comparison of Distribution Fitting

In order to verify the feasibility of using bootstrap, several experiments are taken on applications D1 and D3-5. Two windows repeatedly collect independent n samples respectively from a certain application, and then a series of $\hat{D}^2$ values are calculated to build the "real" distribution $Dist$. $\hat{Dist}$ is built on the training set with bootstrap, which is an estimate of the real one when training samples are limited. Since with the increase of the number m of extractions to generate $\hat{D}^2$ values, the distribution $Dist$ is closer to the real one, $m = 2000$ is large enough to set the reference case.

We discretize the probabilities into 60 intervals, and calculate the mean squared errors (MSE) between $Dist$ and the estimated one. The sample size n is set to 100 and 200, and "Bs" means we use bootstrap to extract subsets from the training set. Each comparison runs 100 times, and the mean and standard deviation of MSE ($mean(std)$) are recorded in Table II. The fitting performance with using bootstrap is

TABLE I
SIMULATION RESULTS WITH SIX METHODS ON DIFFERENT APPLICATIONS

		LSDD-CDT	KNN	H-ICI	CPM-LP	CPM-CvM
D1	FPs	49	306	0	481	462
	FNs	271	94	179	0	0
	Delay(sample)	206.63(259.97)	382.29(308.58)	574.39(233.12)	125.09(169.33)	92.75(144.63)
	CT(s)	12.44(1.94)	17.51(1.13)	0.044(0.0087)	-	-
D2	FPs	49	547	0	480	457
	FNs	42	9	0	0	0
	Delay(sample)	115.62(156.97)	108.56(146.9)	266.08(45.74)	38.83(118.97)	44.86(87.76)
	CT(s)	12.85(1.14)	16.7(2.38)	0.037(0.004)	-	-
D3	FPs	162	452	1	13361	14641
	FNs	5	3	1	0	0
	Delay(sample)	32.81(53.03)	46.03(56.59)	167.78(18.3)	22.7(20.92)	20.84(16.97)
	CT(s)	18.1(1.67)	32.1(3.65)	0.089(0.015)	-	-
D4	FPs	33	0	2500	24500	17672
	FNs	2	0	0	0	0
	Delay(sample)	35.88(5.75)	3.52(0.74)	412.3(388.88)	0.96(1.03)	15.76(9.7)
	CT(s)	13.92(2.42)	25.7(3.3)	0.086(0.028)	-	-
D5	FPs	57	459	0	972	519
	FNs	18	10	242	59	169
	Delay(sample)	83.19(31.24)	74.13(34.12)	656.05(251.85)	314.11(269.45)	298.91(312.82)
	CT(s)	13.55(0.21)	25.76(1.23)	0.066(0.009)	-	-
D6	FPs	57	475	0	382	461
	FNs	1	5	0	250	182
	Delay(sample)	34.05(10.74)	53.01(52.96)	203.92(31.14)	345.18(289.11)	266.68(281.44)
	CT(s)	15.69(0.76)	26.97(2.12)	0.067(0.012)	-	-

similar with that trained by enough samples in D1, D4 and D5. However, when $n = 200$ in D3, the fitting errors with bootstrap are much larger than the others, due to the presence of high dimensional data. For instance, with $d = 10$, only 400 training samples coupled with a large window size (200) are not enough to be close to the "real" distribution. That is, high dimension and large window size demand large training set.

TABLE II
FITTING COMPARISON WITH GAMMA DISTRIBUTION

	Sample size n	D1	D3	D4	D5
Gamma	100	1.63(0.56)	1.35(0.38)	5.042.03)	1.17(0.32)
$(*10^{-5})$	200	1.63(0.53)	1.75(0.41)	2.79(0.98)	1.11(0.29)
Gamma(Bs)	100	2(0.89)	4.24(0.78)	5.13(2.79)	1.26(0.4)
$(*10^{-5})$	200	1.98(0.89)	33.77(5.57)	3.31(1.37)	1.32(0.39)

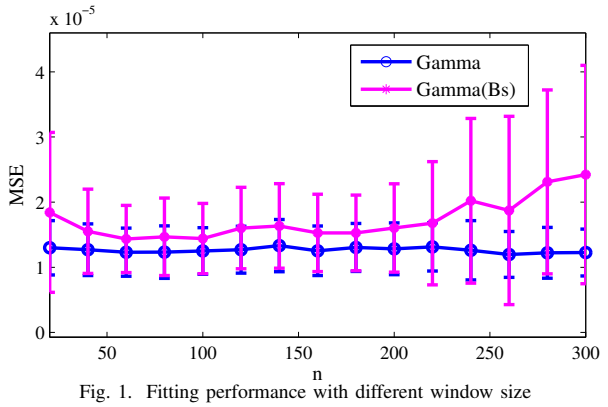

Fig. 1. Fitting performance with different window size

Furthermore, in order to discover how the subset size n impacts, the fitting MSEs along with different n are calculated shown in fig.1. The samples are generated from D1, and it runs 100 times at each value of n. With the increase of n when $n \leq 100$, the performance of bootstrap improves, whereas when $n > 100$, the errors increase because the subset size is closer to the training size. We can draw the same conclusion that fitting the distribution of $\hat{D}^2$ values with bootstrap is appropriate when choosing a suitable n, such as $n = 100$.

B. Performance Analysis

A comprehensive comparison is taken on D1-D6 between our LSDD-CDT and three other detection methods. The window size n is set to 100, and the other settings are consistent as defined in the beginning of the section. At each trial, we record FPs before change location 1000 and FNs if changes are not detected after 1000. There are at most 600 (1000-400) FPs which means the method detects a false positive at each sample, whereas there is 0 or 1 FN which indicates a right detection or a failure. For each application, we sum the FPs and FNs, and calculate the average delay and computational time over 500 trials. The first three methods are implemented in MATLAB, while CPMs operate using the R package cpm. In this case, we didn't record the execution time of the CPMs to keep the comparison fair. The detection performance is shown in Table I.

H-ICI has the least FPs in most applications and the shortest computational time. However, it does not provide an acceptable delay in detecting both small and large changes. It has too many FPs and FNs in D4 and D5 respectively, i.e., it can't offer an accurate detection when data are discrete, and always fails to detect changes in multidimensional problems with small changes.

CPMs do well in D1-2 with the shortest detection delay, whereas the performance of accuracy and promptness in D3-6 is pretty bad. The reasons being the methods are rank-based and cannot deal with multidimensional problems directly without transforming them into several single-dimension problems. Furthermore, in most applications, FPs are too high to give a reliable result, where they almost report a FP every 20 samples in D4. In D6, the high FNs mean that they can't detect these obvious changes.

The KNN-based test does the best in D4 whose observations are discrete with the shortest delay and perfect accuracy. It also shows similar performance in promptness and computational time with our LSDD-CDT. However, it has a bit high FPs in most applications, which means it always reports a false detection before the real change location.

Based on the analysis above, we can get the following conclusions: H-ICI shows an obvious advantage in computational time, CPMs do well in detection promptness on single-dimensional problems with normal distribution, but they both have their intolerable shortcomings and can't handle the most applications well; LSDD-CDT does a good job in promptness under acceptable FPs and FNs in all these applications, that is, it has an excellent integrated and consistent performance. In real applications, methods with low FP rate and small detection delay are always preferred.

V. CONCLUSION

In this paper, we propose a novel pdf-free change detection algorithm for data streams monitoring. It can deal with observations without knowing any priors, especially those which are multi-dimensional. In order to make it applicable with limited samples, a bootstrapping procedure works to extract the sufficient LSDD values during training. By fitting the collected values with a Gamma distribution, thresholds are easily obtained by estimating the upper tolerance limits with predefined FP rates. The comprehensive experiments also show that the proposed LSDD-CDT has a good integrated and consistent performance in promptness and accuracy.

However, the LSDD-CDT takes a long time to train models and operates a test, which limits the application. What's more, the choice strategy of kernel width is also a restriction that the only fixed width might not offer a good fitting performance when dealing with multimodal data. We will be dedicated to solving these problems, and make it more practical.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China under Grant No. 61273136.

REFERENCES

[1] G. J. Ross, D. K. Tasoulis, and N. M. Adams, "Nonparametric monitoring of data streams for changes in location and scale," *Technometrics*, vol. 53, no. 4, pp. 379–389, 2011.

[2] G. J. Ross and N. M. Adams, "Two nonparametric control charts for detecting arbitrary distribution changes," *Journal of Quality Technology*, vol. 44, no. 2, p. 102, 2012.

[3] H. Raza, G. Prasad, and Y. Li, "Dataset shift detection in non-stationary environments using ewma charts," in *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3151–3156.

[4] H. Raza, G. Prasad, and Y. Li, "Adaptive learning with covariate shift-detection for non-stationary environments," in *Computational Intelligence (UKCI), 2014 14th UK Workshop on*. IEEE, 2014, pp. 1–8.

[5] C. Alippi and M. Roveri, "Just-in-time adaptive classifiers-Part I: Detecting nonstationary changes," *Neural Networks, IEEE Transactions on*, vol. 19, no. 7, pp. 1145–1153, 2008.

[6] C. Alippi and M. Roveri, "Just-in-time adaptive classifiers-Part II: Designing the classifier," *Neural Networks, IEEE Transactions on*, vol. 19, no. 12, pp. 2053–2064, 2008.

[7] C. Alippi, G. Boracchi, and M. Roveri, "A just-in-time adaptive classification system based on the intersection of confidence intervals rule," *Neural Networks*, vol. 24, no. 8, pp. 791–800, 2011.

[8] M. F. Schilling, "Multivariate two-sample tests based on nearest neighbors," *Journal of the American Statistical Association*, vol. 81, no. 395, pp. 799–806, 1986.

[9] 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.

[10] S. Liu, J. A. Quinn, M. U. Gutmann, and M. Sugiyama, "Direct learning of sparse changes in markov networks by density ratio estimation," in *Machine Learning and Knowledge Discovery in Databases*. Springer, 2013, pp. 596–611.

[11] M. Sugiyama, T. Kanamori, T. Suzuki, M. Plessis, S. Liu, and I. Takeuchi, "Density-difference estimation," *Neural Computation*, vol. 25, no. 10, pp. 2734–2775, 2013.

[12] T. D. Nguyen, M. C. Du Plessis, T. Kanamori, and M. Sugiyama, "Constrained least-squares density-difference estimation," *IEICE TRANSACTIONS on Information and Systems*, vol. 97, no. 7, pp. 1822–1829, 2014.

[13] E. Parzen, "On estimation of a probability density function and mode," *The annals of mathematical statistics*, pp. 1065–1076, 1962.

[14] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 723–773, 2012.

[15] A. Gretton, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, K. Fukumizu, and B. K. Sriperumbudur, "Optimal kernel choice for large-scale two-sample tests," in *Advances in Neural Information Processing Systems*, 2012, pp. 1205–1213.

[16] K. De Brabanter, S. Sahhaf, P. Karsmakers, J. De Brabanter, J. Suykens, and B. De Moor, "Nonparametric comparison of densities based on statistical bootstrap," in *Proc. of the 4th European Conference on the Use of Modern Information and Communication Technologies (ECUMICT 2010)*, 2010, pp. 179–190.

[17] K. Krishnamoorthy, T. Mathew, and S. Mukherjee, "Normal-based methods for a gamma distribution," *Technometrics*, vol. 50, no. 1, 2008.

[18] D. M. Hawkins and K. Zamba, "Statistical process control for shifts in mean or variance using a changepoint formulation," *Technometrics*, vol. 47, no. 2, 2005.

[19] N. Henze, "A multivariate two-sample test based on the number of nearest neighbor type coincidences," *The Annals of Statistics*, pp. 772–783, 1988.

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

[21] J. A. Johnson and D. W. Wichern, *Applied multivariate statistical analysis 4th ed*, Prentice Hall, 1998.

[22] C. Alippi, D. Liu, D. Zhao, and L. Bu, "Detecting and reacting to changes in sensing units: the active classifier case," *Systems, Man, and Cybernetics: Systems, IEEE Transactions on*, vol. 44, no. 3, pp. 353–362, 2014.

[23] H. Raza, G. Prasad, and Y. Li, "Ewma model based shift-detection methods for detecting covariate shifts in non-stationary environments," *Pattern Recognition*, vol. 48, no. 3, pp. 659–669, 2015.

[24] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Machine learning*, vol. 23, no. 1, pp. 69–101, 1996.

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