Hierarchical Change-Detection Tests

Cesare Alippi, *Fellow, IEEE*, Giacomo Boracchi, Manuel Roveri

***Abstract*—We present hierarchical change-detection tests (HCDTs), as effective online algorithms for detecting changes in datastreams. HCDTs are characterized by a hierarchical architecture composed of a *detection* and a *validation* layer. The detection layer steadily analyzes the input datastream by means of an online, sequential, change-detection test (CDT), which operates as a low-complexity trigger that promptly detects possible changes in the process generating the data. The validation layer is activated when the detection one reveals a change, and performs an offline, more sophisticated, analysis on recently acquired data to reduce false alarms.**

Our experiments show that, when the process generating the datastream is unknown, as it is mostly the case in the real world, HCDTs achieve a far more advantageous trade-off between false-positive rate and detection delay than their single-layered, more traditional, counterpart. Moreover, the successful interplay between the two layers permits HCDTs to automatically reconfigure after having detected and validated a change. Thus, HCDTs are able to reveal further departures from the new, post-change, state of the data-generating process.

***Index Terms*—Change-Detection Tests, Stream Data Analytics, Hierarchical Architectures, Cognition-Inspired Systems, Cognitive Fault Detection, Big Data Analytics.**

I. Introduction

One of the most important challenges in datastream analysis is the online detection of changes affecting the data-generating process. Changes might reveal critical situations, such as a fault affecting a sensing apparatus, an anomalous event, or an unforeseen evolution of the surrounding environment, to name a few examples. As such, a prompt detection of these situations is essential for undertaking suitable countermeasures like repairing/replacing a sensor, raising an alarm, or activating adaptation mechanisms [1].

Datastreams can be conveniently monitored by online and sequential techniques characterized by a low computational-burden; this is particularly true in *big data* scenarios, where massive amounts of data steadily arrive over time. Methods designed to detect changes in datastreams are typically referred to as change-detection tests (CDTs) [2]; in the classification literature [1] changes in the data-generating process are referred to as *concept drift* [3].

Due to their statistical nature, CDTs intrinsically introduce false-positives, which might prompt costly and unnecessary reactions to the detected -not existing- change. For instance, in contaminant-detection systems [4], alarms activate emergency procedures or disruptive interventions, and frequent false alarms induce cry-wolf effect. Even in less critical scenarios, where false alarms are harmless, they might unnecessarily activate adaptation procedures or request human intervention. Therefore, CDTs need to be suitably configured to operate at low false-positive rate (FPR). Unfortunately, a reduction of the FPR comes at the expenses of an increased detection delay (DD), since each CDT is characterized by an intrinsic FPR *vs* DD trade-off controlled by its own parameters.

To achieve more advantageous FPR vs DD trade-offs it is necessary to design a new change-detection algorithm. To this purpose, we propose hierarchical change-detection tests (HCDTs), powerful algorithms that combine different techniques to detect and validate changes. More specifically, each HCDT features a two-layered architecture consisting in a *detection layer* and *validation layer*, and implements an automatic *reconfiguration* mechanism. The detection layer is designed to steadily analyze the datastream at a low computational cost, by means of an *online* and sequential CDT. Once the detection layer reveals a change, it activates the validation layer that performs an *offline* analysis based on an hypothesis test (HT) to determine whether the detection corresponds to an actual change in the data-generating process or not (false-positive detection). When the change is actually confirmed, the validation layer automatically identifies a sequence of data generated in the post-change conditions to be used to reconfigure the HCDT. Differently, when the change is not confirmed, the HCDT restarts in its initial conditions. This self-reconfiguration phase makes HCDTs able to autonomously track data-generating processes that evolve through stationary states. In turn, HCDTs reconfiguration can be also used to pilot other adaptation mechanisms like those governing adaptive classifiers [1], [5]–[7].

Here we show that the change-detection performance can be often improved by introducing a validation layer. In fact, our experiments reveal that HCDTs often achieve a far more convenient FPR *vs* DD trade-off than their single-layered counterpart, namely the solely CDT operating at the detection layer. In particular, HCDTs outperform traditional (single-layered) CDTs in monitoring scenarios where the pre/post change states of the data-generating process are unknown, as it is the case in most of real-world applications. Thus, provided a suitable configuration of the detection layer, HCDTs achieve substantially lower FPR than their single-layered counterparts, while preserving similar DD values. Such performance improvement comes at a marginal computational overhead. In fact, when the HCDT is properly configured, the validation layer is rarely activated and its computational load is largely due to the detection layer, thus HCDTs represent viable solutions for online monitoring datastreams.

A. Related Works

Hierarchies of mechanisms/algorithms have been considered in many research fields, where hierarchical architectures are

The authors are with the Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy e-mail: firstname.lastname@polimi.it

often presented as solutions to trade-off effectiveness and efficacy. Without intending to be exhaustive, we recall the hierarchy of memories in computer architectures [8], hierarchical algorithms in computer vision (e.g., [9], [10]) and hierarchical routing algorithms in wireless sensor networks (e.g., [11]).

Recent studies [12], [13] have shown that also some mechanisms in the human brain can be modeled as a hierarchy of sub-systems, characterized by different activation times and response accuracies [14]. Sub-systems in lower levels (e.g., the amygdala) are characterized by automatic processes that continuously monitor the external stimuli and promptly trigger potential threats to activate quick reactions (e.g., increase the heartbeat or the respiration rates or the release of stress-hormones). Differently, sub-systems in upper levels (e.g., the prefrontal cortex) are generally activated in response to detections raised by lower levels and are characterized by more complex conscious processes, aiming at improving, integrating or even correcting the actions/decisions made by the lower levels. A detailed description of automatic and conscious mechanisms for emotional processes can be found in [13].

Interestingly, hierarchical approaches received a very little attention in the change-detection literature. In fact, even though the idea of combining online and offline change-detection techniques to increase the reliability of the decisions was originally discussed in [2], to the best of our knowledge, a general hierarchical-architecture for change-detection purposes has never been investigated.

It is worth mentioning the CDT presented in [5], which implements a two-thresholds mechanism to analyze the average error of a classifier to detect concept drift. As soon as the error exceeds the lower threshold, a warning-level alert is raised; when the alert state persists and the error exceeds the higher threshold, a drift-level detection is triggered. Drift-level detections force the classifier to update, whereas warning-level alerts that do not trigger a drift-level detection are considered to be false alarms. In [15] a similar mechanism is used to detect concept drift by analyzing the average distance between misclassified data. While these algorithms are able to perform automatic reconfiguration, they do not feature a hierarchical architecture. A two-layered CDT was presented in [16], where a validation procedure was employed to reduce the FPR. This CDT has been also used for contaminant-detection purposes in smart-building scenarios [4].

Our work extends [16] and provides the following original contributions. At first, we present HCDTs from an high-level perspective, detailing a general methodology for designing HCDTs based on different change-detection and validation techniques; second, we develop the automatic reconfiguration (Algorithm 2) of HCDTs; third, we perform a larger experimental analysis to investigate the improvements that HCDTs provide in terms of FPR *vs* DD trade-off. To this purpose, we designed and assessed the performance of three different HCDTs, while [16] presented a single HCDT implementing the ICI-based CDT [7].

B. Paper Structure

The rest of the paper is organized as follows. Section II formalizes the change-detection problem, while Section III describes the general methodology for designing HCDTs. Section IV overviews suitable techniques for designing specific HCDTs, and meaningful examples of HCDTs are provided in Section V. Experiments are presented and discussed in Section VI, while Section VII provides concluding remarks.

II. Problem Statement

Denote by $\{s(t), t = 1, \dots\}$ the datastream to be inspected for changes, $s(t) \in \mathbb{R}^p$ being the data acquired at time instant t. We assume that, provided some suitable preprocessing function $\mathcal{P}$ (e.g., those mentioned in Section IV-A), it is possible to extract a stream $X = \{x(t), t = 1, \dots\}$, $x(t) \in \mathbb{R}^d$, of *change indicators* that, in stationary conditions, are independent and identically distributed (i.i.d.) realizations of a random variable $\mathcal{X}$ having probability density function (pdf) ϕ_0. In the sequel, we address the problem of monitoring the stream X to identify changes in stationarity of $\mathcal{X}$ which, in turn, indicate a change in the process generating the datastream $\{s(t), t = 1, \dots, \}$.

A common model for changes in stationarity of $\mathcal{X}$ is

$$x(t) \sim \begin{cases} \phi_0 & t < T^* \\ \phi_1 & t \geq T^* \end{cases} , \tag{1}$$

where T^* is the unknown change point (or change-time instant) and $\phi_1 \neq \phi_0$ is the pdf characterizing the postchange distribution of the data. Model (1) corresponds to an abrupt and permanent change affecting ϕ_0.

Define the detection time $\widehat{T}$ as the earliest time instant when the CDT claims that the sequence $X_{\widehat{T}} = \{x(t), t = 1, \dots, \widehat{T}\}$ contains a change point. Following (1), the CDT promptness can be computed by the detection delay:

$$DD = \underset{X}{E}[\widehat{T} - T^* | \widehat{T} \geq T^*, \phi_1], \tag{2}$$

namely, the expectation of the detection latency $\widehat{T} - T^*$ conditioned by the fact that the change was successfully detected, (i.e., $\widehat{T} \geq T^*$). Expectation in (2) is computed over realizations of sequences X generated by model (1). When the sequence $X_{\widehat{T}}$ contains i.i.d. data (thus no change-point), the CDT had a false-positive detection at $\widehat{T}$. As mentioned in Section I, the FPR, namely, the probability of a CDT to yield a false-positive detection on a given i.i.d. sequence, has to be also considered when assessing the CDT effectiveness[1].

III. Hierarchical Change-Detection Tests

Figure 1 illustrates the architecture of HCDTs, indicating the interplay between the detection and the validation layers. The general formulation of HCDTs is detailed in Algorithm 1, while the reference methodology for deriving specific HCDTs is presented in the sequel.

[1] In the sequential monitoring literature, CDT performance is also assessed as the expected time between false positives, namely, the average run length $ARL_0 = \underset{X}{E}[\widehat{T}|\phi_0]$, and the mean delay $ARL_1 = \underset{X}{E}[\widehat{T}|\phi_1]$.

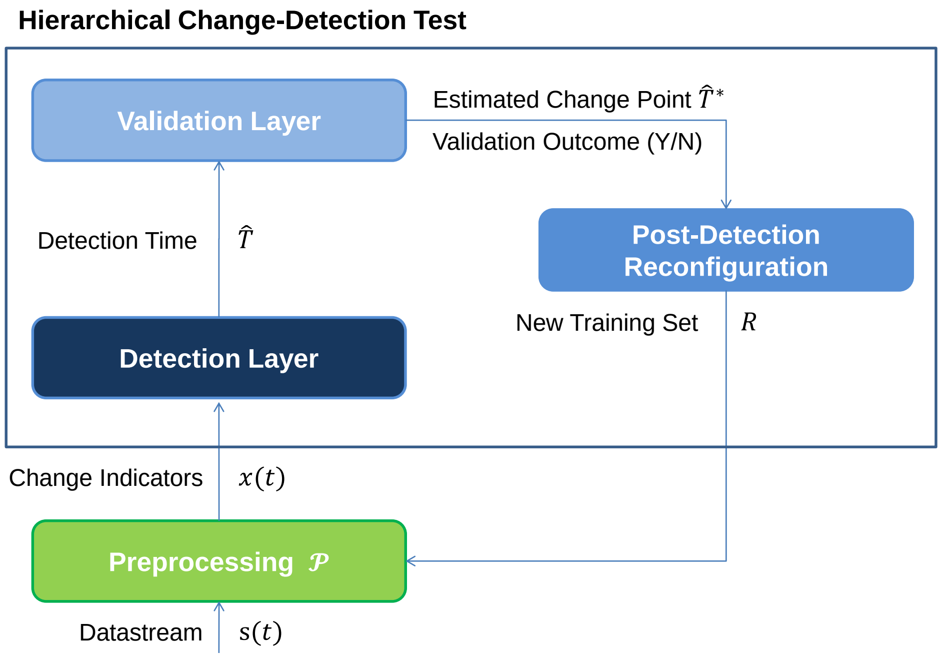

Hierarchical Change-Detection Test
Validation Layer
Estimated Change Point $\widehat{T}^*$
Validation Outcome (Y/N)
Detection Time $\widehat{T}$
Post-Detection Reconfiguration
New Training Set R
Detection Layer
Change Indicators $x(t)$
Preprocessing $\mathcal{P}$
Datastream $s(t)$

Fig. 1. A scheme illustrating the HCDT architecture. When the datastream $\{s(t), t = 1, \dots, \}$ can not be modeled as a sequence of i.i.d. random values, it is necessary to perform a preliminary processing $\mathcal{P}$ yielding the stream of i.i.d. change indicators $X = \{x(t), t = 1, \dots, \}$, otherwise $s(t) = x(t)$. The detection layer runs a CDT on the input stream X, and activates the validation layer as soon as it detects a change at time $\widehat{T}$. Then, the validation layer identifies a suitable validation sequence $V \subset X_{\widehat{T}}$ and runs an HT to assess whether V contains a change point or not. When the change point $\widehat{T}^*$ is found, the detection is confirmed, and a subsequence of the datastream $R = \{s(t), t = \widehat{T}^*, \dots, \widehat{T}\}$ is identified to reconfigure the HCDT and possibly the preprocessing. Differently, when the change is not validated, R remains the original training sequence and the HCDT is restarted in its previous conditions. This automatic reconfiguration makes the HCDT able to continue the monitoring activity after each detected change, thus being able to detect any further departure from the post-change conditions.

A. *Preprocessing*

When the datastream follows trends or, more generally, exhibits specific structures or dynamics (as it often happens in the real world), the HCDT can not be directly applied, since the datastream $\{s(t), t = 1, \dots\}$ violates the i.i.d. assumption. Thus, suitable preprocessing-techniques, like those mentioned in Section IV-A, are typically envisioned to compute a stream of change indicators X that follows (1). In these cases, preprocessing $\mathcal{P}$ is the first operation to be performed (Algorithm 1, line 3) and the stream of change indicators X becomes the input of the HCDT. The preprocessing phase has to be designed so that changes in the datastream modify the distribution of change indicators; examples of preprocessing techniques are given in Section IV-A. When preprocessing is not needed, $x(t) = s(t)$, and $\mathcal{P}$ in Algorithm 1 becomes the identity function.

B. *Detection Layer*

The detection layer steadily analyzes the input stream X to detect changes that might occur at anytime. In particular, the detection layer answers, at each new input arrival, the following question: "*are all data received so far generated by a stationary process?*". As far as the answer to this question is positive, no detection/alarm is raised, and the monitoring activity continues. In contrast, a negative answer implies the detection of a change in the data-generating process.

ALGORITHM 1: General formulation of the HCDT.

```
input: $R$, the training sequence from the datastream
1.  Configure the HCDT on $R$ (Algorithm 2)
2.  while ($s(t)$ is available) do
3.      Apply preprocessing $\mathcal{P}$ yielding $x(t)$
4.      Run the CDT at the detection layer
5.      if (CDT detects a change at time $t$) then
6.          Set $\widehat{T} = t$
7.          Activate the validation layer
8.          Identify the validation sequence $V \subset X_{\widehat{T}}$
9.          Run the HT to find a change point in $V$
10.         if (a change-point is found) then
11.             Confirm the detection $\widehat{T}$ and change point $\widehat{T}^*$
12.             Define $R = \{s(t), t = \widehat{T}^*, \dots, \widehat{T}\}$
13.             Reconfigure the HCDT on $R$ (Algorithm 2)
            else
14.             Restart the HCDT as configured over $R$
            end
        end
    end
```

Streaming data have to be monitored in an *online* and *sequential* manner, by means of suitable techniques that potentially consider the whole sequence $X_T = \{x(t), t = 1, \dots, T\}$ before claiming a change at time T. The best candidates for addressing change detection on streaming data are the online and sequential CDTs, namely statistical techniques

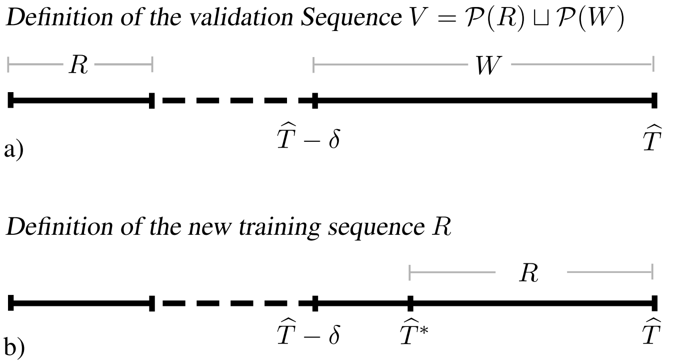

Definition of the validation Sequence $V = \mathcal{P}(R) \sqcup \mathcal{P}(W)$
R
W
a)
$\widehat{T} - \delta$
$\widehat{T}$
Definition of the new training sequence R
R
b)
$\widehat{T} - \delta$
$\widehat{T}^*$
$\widehat{T}$

Fig. 2. a) The validation sequence $V \subset X_{\widehat{T}}$ is defined as $V = \mathcal{P}(R) \sqcup \mathcal{P}(W)$, being $\mathcal{P}(R)$, the sequence of change indicators that was used to configure the CDT at the detection layer and $\mathcal{P}(W)$ the output of the preprocessing over the window $W = \{s(t), t = \widehat{T} - \delta, \dots, \widehat{T}\}$ that contains the most recent data. b) $R = \{s(t), t = \widehat{T}^*, \dots, \widehat{T}\}$ is a subsequence of the datastream that is used to reconfigure the whole HCDT after the change, according to Algorithm 2.

able to detect changes in data sequences characterized by an increasing length. Sequential CDTs can detect subtle changes more effectively than *one shot* techniques applied to the last recent data.

In HCDTs, the CDT at the detection layer monitors each incoming $x(t)$ (Algorithm 1, line 4). At time $\widehat{T}$, when the detection layer reveals a change in $\mathcal{X}$ (line 5), the validation layer is activated (line 7).

Given its online processing modality, the CDT must be characterized by a low DD and a low computational burden. Having low DD is fundamental because any delay at the detection layer reflects on the overall latency and reaction time of the HCDT. The CDT at the detection layer has to be computationally light since it is executed at each new sample arrival and, to support online operations in big data scenarios, the inner statistics of the CDT have to be computed incrementally over-time. Examples of CDTs that can be employed at the detection layer are given in Section IV-B.

C. Validation Layer

The validation layer gets activated every time the CDT at the first layer detects a change, to assess whether $\mathcal{X}$ has actually changed or the CDT provided a false-positive detection. To this purpose, a suitable validation sequence is extracted from $X_{\widehat{T}}$ and a statistical test is formulated to answer the following question: "*does the validation sequence contain a change-point?*". Therefore, change validation consists in a *retrospective* and *offline* analysis over a given sequence having a *fixed length* and, to this purpose, the validation layer makes use of HTs.

Identification of the Validation Sequence: the validation sequence $V \subset X_{\widehat{T}}$ is isolated from X after each detected change (Algorithm 1, line 8). To properly perform change validation, V has to include –when $\mathcal{X}$ actually underwent a change– both data generated before and after the unknown change-point. Therefore, we select a window containing the last $\delta > 0$ arrivals of the datastream, i.e. $W = \{s(t), t = \widehat{T} - \delta, \dots, \widehat{T}\}$, which is expected to refer to the post-change conditions, and the training sequence of the HCDT, which certainly refers to the initial stationary conditions. As shown in Figure 2.a, V is defined as $V = \mathcal{P}(R) \sqcup \mathcal{P}(W)$, where $\mathcal{P}(R)$ is the sequence of change indicators that was used as a training set for the CDT at the detection layer, $\mathcal{P}(W)$ is the sequence of change indicators extracted from W, and $\sqcup$ denotes the juxtaposition operator. This option enables change validation without having to store all the input data, which is often unfeasible in datastream analysis.

Change Validation via Hypothesis Testing: two strategies can be pursued to determine whether V contains a change-point or not. The first one consists in analyzing V to compute $\widehat{T}^*$, a refined estimate[2] of the change point T^*, and then defining

$$\begin{aligned} V_0 = \quad & \{x(t) \in V,\ t < \widehat{T}^*\} \\ V_1 = \quad & \{x(t) \in V,\ t \geq \widehat{T}^*\}\,. \end{aligned} \tag{3}$$

These sequences yield a partition of V (i.e., $V = V_0 \sqcup V_1$) that, when the change has actually occurred at $\widehat{T}^*$, is expected to provide the largest evidence for claiming V contains a change point. The detection can then be validated by formulating an hypothesis test like the following:

$$\begin{aligned} \mathcal{H}_0 : \quad & \text{"data in } V_0 \text{ and } V_1 \text{ are identically distributed"} \\ \mathcal{H}_1 : \quad & \text{"data in } V_0 \text{ and } V_1 \text{ are from two different pdfs"}\,. \end{aligned} \tag{4}$$

When the test statistic provides enough statistical evidence to conclude that V contains a change point, the detection at $\widehat{T}$ is confirmed together with its estimated change-point $\widehat{T}^*$ (Algorithm 1, line 11). Conversely, when there is not enough statistical evidence, the detection raised by the CDT is discarded and considered to be a false-positive detection. Several test statistics could be employed to design such hypothesis test, the most relevant ones are reviewed in Section IV-C, together with techniques to estimate $\widehat{T}^*$.

The second strategy consists in change-point methods (CPM) [17], namely HTs that simultaneously validate the change and estimate $\widehat{T}^*$. In particular, each point in V is tested to be a change point, thus the HT in (4) is executed for all the possible partitions of V, to determine whether any of these yields enough statistical evidence for rejecting the null hypothesis. Once the change has been validated, the partition yielding the largest evidence of the change identifies $\widehat{T}^*$.

Detection and validation layers are characterized by different requirements. Since the validation layer is only sporadically activated, its computational load is not a critical issue as for the detection layer. Differently, the HT employed at the validation layer has to be powerful (in statistical terms), namely should reject with high probability the stationary hypothesis when a change has actually occurred in $\mathcal{X}$. In particular, the validation layer has to be able to confirm the decisions of the detection layer even when V contains few data after the change.

[2] CDTs are typically characterized by an intrinsic delay since they require enough certainty before claim the presence of a change in the input stream, thus $\widehat{T}^* \neq \widehat{T}$.

ALGORITHM 2: Reconfiguration of the HCDT.

```
    input: training sequence $R$
1.  if ($\mathcal{P}$ has to be configured from training data) then
2.      Split $R = R_0 \sqcup R_1$
3.      Configure $\mathcal{P}$ on $R_0$
4.      Compute $R_X = \mathcal{P}(R_1)$
    else
5.      Compute $R_X = \mathcal{P}(R)$
    end
6.  if (not enough samples in $R_X$ for reconfiguring HCDTs) then
7.      Gather more recent data $S$ from the datastream
8.      Set $R_X = R_X \sqcup \mathcal{P}(S)$
9.      Run the validation layer on $R_X$
        if a change point is found in $R_X$ then
10.         Set $R = \{x(t) \in S, t > \widehat{T}^*\}$
11.         Restart from line 1
        end
    end
12. Configure the CDT at the detection layer from $R_X$
```

D. HCDT Reconfiguration

At each confirmed change, the validation layer provides a new sequence $R = \{s(t),\ t = \widehat{T}^*, \dots, \widehat{T}\}$ that is supposed to contain data generated after $\mathcal{X}$ has changed (see Figure 2.b, and Algorithm 1 line 12). This sequence can be used to automatically reconfigure the HCDT to the post-change conditions. Such automatic reconfiguration can be performed at each detection, making HCDTs able to autonomously track data-generating processes evolving through a sequence of stationary states.

HCDT reconfiguration involves the CDT at the detection layer and possibly the preprocessing phase $\mathcal{P}$, but not the validation layer. Both the initial configuration (Algorithm 1 line 1), and the post-detection reconfiguration (Algorithm 1 line 13) of the HCDT can be performed by the general procedure reported in Algorithm 2 and described in what follows.

First of all, the training sequence R is conveniently split into two parts (Algorithm 2 line 2), the former (R_0) is used for reconfiguring the preprocessing (line 3), the latter for computing the change indicators $R_X = \mathcal{P}(R_1)$ (line 4) that are used for configuring/reconfiguring the CDT at the detection layer. This step is necessary to prevent overfitting [18] when the preprocessing is configured from training data – e.g. when $\mathcal{P}$ involves an approximation model that is fitted to the training data. In fact, change indicators computed from the same data used for configuring $\mathcal{P}$ might be poorly representative of the change indicators computed during operational life, thus they should not be used for CDT configuration. When $\mathcal{P}$ does not need to be configured, $\mathcal{P}$ can be directly applied to the whole training sequence $R_X = \mathcal{P}(R)$ (line 5); similarly, when no preprocessing phase is needed, $\mathcal{P}$ becomes the identity and $R_X = R$.

When R_X does not contain enough training samples to allow the reconfiguration of the CDT, the activation of the HCDT is postponed to gather additional training data (line 7). However, data received after $\widehat{T}$ have not been tested by the HCDT and, hence, we cannot guarantee they are stationary. Therefore, after having computed the change indicators by means of $\mathcal{P}$ (line 8), it is safer to perform an additional validation step to make sure that the new training sequence is stationary (line 9). To this purpose, R_X is further tested by the validation layer, and used for reconfiguring the CDT when no change points are found (line 12). Otherwise, R is defined as the part of the datastream generated after the change, and the whole reconfiguration is repeated (lines 10 and 11). This additional validation is meant to prevent the HCDT reconfiguration on nonstationary data, e.g., sequences containing gradual drifts or an abrupt change.

It is important to remark that, when the validation layer does not confirm the detection at $\widehat{T}$, the CDT is reset to its original conditions. In the practice, all data acquired before the false-positive detection are ignored and the CDT is restarted to detect any departure from R_X after $\widehat{T}$ (Algorithm 1 line 14).

E. Pairing the Detection and Validation Layers

To guarantee the successful interplay of the detection and validation layers, the CDT and HT that constitute a HCDT should be selected with special care. This is particularly important in nonparametric monitoring, where, as we will show in Section IV, both the CDTs and HTs typically rely on test statistics that are meant to detect/validate specific types of changes (e.g., changes in the mean or in the variance). Then, for the HCDT to be successful, the HT at the validation layer should be able to validate each type of change that the CDT at the detection layer is able to detect. The three HCDTs in Section V have been designed with this constraint in mind.

Another important remark is that the detection and validation layers have clearly different roles, and that the validation layer cannot be used to online monitor the datastream (even though this leverages a powerful HT). In fact, activating the validation layer at each new input $x(t)$ might yield, beside computational issues (as we show the validation layer is often much more computationally demanding than the detection one), unacceptable FPR [19] since HTs are typically one-shot techniques.

IV. Techniques for Implementing HCDTs

Here, we provide an overview of some consolidated techniques that can be used to design specific HCDTs.

A. Preprocessing Techniques

The literature presents several data-processing solutions to compute change indicators that are distributed as in (1). Most often, preprocessing is performed by computing the residuals with respect to approximation/predictive models [20]–[22] or decorrelating/detrending the datastream [23], [24]. Feature extraction is another viable option to compute change-indicators: examples are the sample moments of the datastream (computed over non-overlapping windows to guarantee the temporal independence of the change indicators over time), a measure of datastream self-similarity [25], and the Hellinger distance of the empirical distributions [26] computed with respect to a reference training sequence. In the classification literature, concept drift is typically detected by monitoring the classification error on supervised samples [1], [3], [5], [15], [27], thus ϕ_0 and ϕ_1 are two Bernoulli distributions whose parameters are the average classification error before and after the change, respectively.

B. Change-Detection Tests

CDTs in the literature can be divided into *parametric*, which assume that ϕ_0 and ϕ_1 are a priori known, and in *nonparametric* which do not make such assumption. Among parametric CDTs we mention the sequential probability ratio test (SPRT) [2] and the cumulative sums (CUSUM) test [28], while for a survey on recent developments in the quickest change-detection literature, we invite the reader to refer to [29].

We are mainly interested in nonparametric CDTs since, in the real world, the post-change distribution ϕ_1 is rarely known because of the change unpredictability or lack of training data. Only few nonparametric CDTs have been proposed in the literature, like the NP-CUSUM test [30], which is a nonparametric extension of the CUSUM designed to detect shifts in the expectation of an unknown random variable. The CI-CUSUM test [31] aims at detecting more general distribution changes by extracting features like the sample moments, projections over the principal components and the Mann-Kendall statistics from a sliding window. Thus, the problem of detecting arbitrary distribution changes of $\mathcal{X}$ is transformed into the problem of detecting changes in the magnitude of the feature vector components (by means of a cumulative-sum mechanism), which are also very likely to change when $\mathcal{X}$ changes. This idea is further developed in the ICI-based CDTs [7], where features are carefully designed to follow a Gaussian distribution. CDTs of this family leverage the Intersection of Confidence Intervals (ICI) rule [32], [33], a statistical technique to define adaptive supports for polynomial regression, to detect changes in the monitored features. The CDT in [34] detects distribution changes by directly estimating the likelihood ratio between ϕ_0 and ϕ_1, without explicitly estimating the two distributions. Differently from the other CDTs discussed here, this latter CDT does not refer to a unique (initial) stationary state, and detects changes by comparing data over two different sliding windows. A batch-wise nonparametric CDT that monitors the Hellinger distance between the empirical distributions computed from the current batch and the training set is presented in [26]. Simple thresholding-based CDTs as well as other change-detection solutions designed for fixed-length sequences have not been considered here.

C. Validation Techniques

In what follows we describe the main techniques for performing change validation on a given sequence V (defined as in Section III-C), namely the techniques to estimate $\widehat{T}^*$ and the HTs.

Estimating $\widehat{T}^*$: To formulate the hypothesis testing problem (4) the validation layer has first to compute $\widehat{T}^*$ from V. In the parametric scenario (where both ϕ_0 and ϕ_1 are known), $\widehat{T}^*$ is obtained as in [2] by maximizing the likelihood of the change hypothesis over V, i.e.,

$$\widehat{T}^* = \underset{t\in[\widehat{T}-\delta,\widehat{T}]}{\text{argmax}} \log \left(\prod_{x(i)\in V, i<t} \phi_0(x(i)) \prod_{x(i)\in V, i\geq t} \phi_1(x(i)) \right). \quad (5)$$

In contrast, in the nonparametric case, $\widehat{T}^*$ has to be estimated through heuristic approaches, such as running the CDT at the detection layer over V, after having adjusted its parameter to provide prompter detections. Then, $\widehat{T}^*$ is defined as the detection time from this latter execution over V. ICI-based CDTs [7] naturally increase their detection promptness when operating on shorter data sequences: this has motivated the design of the refinement procedure (see [7], Algorithm 3), which can be directly used on V to estimate $\widehat{T}^*$.

Change Validation via Hypothesis Testing: Both the HTs (4) and CPMs can use nonparametric statistics such as the Kolmogorov Smirnov [35] or the Cramer-Von Mises [36] ones, which compare the empirical cumulative density functions of V_0 and V_1. Since it is very likely that a change in $\mathcal{X}$ would also affect its moments, it is often more convenient to look for changes in the sample moments of $\mathcal{X}$ or other meaningful statistics. In fact, tests based on statistics that detect changes in the distribution are typically less powerful [37] than tests based on statistics meant to assess specific sort of changes (e.g., changes in the sample moments). Examples of statistics typically employed in HTs like (4) are the Hotelling T-square [38], which detects changes in the mean, the Bartlett [39], which detects changes in the variance, the Mann Withney [40], which detects changes in the location of the distribution (thus also in the mean), the Mood [41], which detects changes in the scale of the distribution (thus also in the variance) or the Lepage [42], which detects changes in both the location and scale. In [43], change points are located by estimating the density ratio before and after the change.

The change-point formulation has been recently extended to perform online and sequential-change detection, by iterating the CPM at each new data arrival. Thus, at each new input, the test statistics have to be computed for all the possible partitions of X like (3). The computational load of these algorithms depends on the length of X_t, thus increases over time. This issue becomes a serious problem when the test statistics cannot be computed incrementally. Approximated expressions of the test statistics have been recently proposed in [23], [44], to bound the computational and memory requirements. However, these algorithms are far more computationally demanding than the CDTs in Section IV-B, since the test statistics have still to be computed on all possible partitions of a sliding window opened over the datastream. This is the main motivation why not havnig online CPMs at the detection layer.

V. Examples of HCDTs

To substantiate the general methodology described in Section III we present three examples of HCDTs. For each HCDT, we compare the computational load of the detection and validation layers.

The fist algorithm we report is a parametric HCDT built over the CUSUM test, the second one is a feature-based (nonparametric) HCDT that implements the ICI-based CDT, while the third one is another example of nonparametric HCDTs that is based on the NP-CUSUM test. The corresponding validation layers have been selected according to the criteria in Section III-E, while the reconfiguration is performed following Algorithm 2. We have made available for download MATLAB implementations of these HCDTs[3].

A. Hierarchical CUSUM Test

Detection Layer: The CUSUM test [28] is a parametric CDT that monitors the behavior of the log-likelihood ratio

$$s(t) = \log\left(\frac{\phi_1(x(t))}{\phi_0(x(t))}\right). \tag{6}$$

When $x(\cdot) \sim \phi_0$, (6) is expected to be negative, thus its cumulative sum

$$\mathtt{S}(t) = \sum_{i=1}^{t} s(i), \tag{7}$$

follows a decreasing trend. The stopping time of the CUSUM test is defined as

$$\widehat{T} = \min\left\{t : (\,\mathtt{S}(t))^+ > \kappa\right\}, \tag{8}$$

where $(\cdot)^+ = \max\{0, \cdot\}$ and κ is the CUSUM threshold parameter that has to be properly tuned.

Validation Layer: After each detection, $\widehat{T}^*$ is estimated by maximizing the posterior probability of the change hypothesis [2] over V as in (5). Given $\widehat{T}^*$, V is partitioned as in (3) to perform change validation by means of a suitable parametric HT (4), since both ϕ_0 and ϕ_1 are known. For instance, in case of Gaussian distributions, changes in the mean can be validated with a one-sided z-test, while changes in the variance with the chi-square test. Since the CUSUM test does not require any training sequence, the hierarchical CUSUM uses R_X only to build the validation sequence V.

The computational load of the Hierarchical CUSUM test is mainly determined by the cost in computing the likelihood with respect to ϕ_0 (i.e., assessing $\phi_0(x(i))$) and ϕ_1. The CUSUM at the detection layer requires only 2 likelihood computations at each sample arrival. Differently, an activation of the validation layer involves (5), which requires approximatively $(\delta + 1)n/2$ likelihood computations, where n is the cardinality of the validation sequence ($n = \#V$). Thus, the validation layer is substantially more computationally demanding than the detection layer.

B. Hierarchical ICI-based CDT

Detection Layer: The ICI-based CDT presented in [45] monitors two features computed over non-overlapping windows of $\nu > 0$ change indicators. In what follows, we use j to index sequence of feature values. The first feature is the sample mean:

$$\mathtt{M}(j) = \frac{1}{\nu}\sum_{t=j\nu}^{(j+1)\nu-1} x(t), \tag{9}$$

while the second one is a power-law transform of the sample variance

$$\mathtt{V}(j) = \left(\sum_{t=j\nu}^{(j+1)\nu-1} \frac{\big(x(t) - \mathtt{M}(j)\big)^2}{\nu - 1}\right)^{h_0}, \tag{10}$$

which approximately follows a Gaussian distribution [46]. The exponent h_0 is computed as described in [46].

During operational life, both features are independently monitored by the ICI rule [32], which determines when the sequence of feature $\mathtt{M}$ or $\mathtt{V}$ cannot be suitably fit by a zero-order polynomial. In particular, let μ_j be the polynomial fit of $\{\,\mathtt{M}(1), \dots, \mathtt{M}(j)\}$ (the same applies to $\{\,\mathtt{V}(1), \dots, \mathtt{V}(j)\}$), and let σ_j be the standard deviation of the corresponding estimator, such that the confidence interval around the j-th estimate is

$$\mathcal{I}_j = [\mu_j - \Gamma\sigma_j; \mu_j + \Gamma\sigma_j]\,, \tag{11}$$

where $\Gamma > 0$ is a tuning parameter that in practice rules the FPR *vs* DD trade-off in the the CDT. Then, the CDT detects change within the j-th window as soon as $\mathcal{I}_j$ does not intersect all the previous confidence intervals, i.e., when $\bigcap\limits_{k=1,\dots,j} \mathcal{I}_k = \emptyset$. The intersection of confidence intervals (and often also μ_j) can be incrementally computed, while σ_j is given by analytical expressions.

Validation Layer: We present two different validation layers that are able to assess changes both in the mean and in the variance of $\mathcal{X}$, since these are the changes that the above ICI-based CDT is designed to detect. The first option consists in estimating $\widehat{T}^*$ inside V by means of the refinement procedure [7] of the ICI-based CDT applied over the feature detecting the change, and then running an Hotelling T-square test [38] on two-dimensional vectors $[\,\mathtt{M}(\cdot),\, \mathtt{V}(\cdot)]$ – which are treated as realizations from a multivariate Gaussian random variable[4]. The second option consists in running a CPM based on the Lepage statistic $\mathcal{L} = \mathcal{U} + \mathcal{M}$ [42], which is the sum of the Mann-Whitney statistic $\mathcal{U}$ [40] and Mood statistic $\mathcal{M}$ [41]. As mentioned in Section IV-C, $\mathcal{L}$ is a nonparametric statistic able to assess both changes in the location (thus, also the mean) and/or the scale (thus, also the variance) of $\mathcal{X}$. The training sequence R_X is used to configure the ICI-based CDT and build the validation sequence V.

The ICI-based CDT is computationally light, since it requires around 4ν operations every ν data arrived. The validation layer that estimates $\widehat{T}^*$ at first and then runs the HT based on Hotelling T-square statistic (4) is also computationally light, since the refinement procedure requires few iterations of the ICI-based CDT over V, and the HT involves few operations on the feature vectors in V (that are n/ν and that have been previously computed). Differently, the validation layer

[3] `http://home.deib.polimi.it/boracchi/Projects/`

[4] Alternatively, changes can be validated on the sole feature detecting the change by means of a t-test.

implementing the Lepage CPM on V is computationally more bulky, since the statistics $\mathcal{M}$ and $\mathcal{U}$ require to sort V_0 and V_1, thus $\mathcal{O}\left(n\ log\left(n\right)\right)$ operations, and have to be computed $(n-1)$-times (namely, for all the possible partitions of V). The CPM is more computationally demanding than the CDT also because typically $n >> n/\nu$.

C. Hierarchical NP-CUSUM Test

Detection Layer: The NP-CUSUM test was introduced in [30] to detect shifts in the mean of a datastream drawn from an unknown distribution. This CDT monitors the score function

$$\mathtt{S}_u(t) = \big(\mathtt{S}_u(t-1) + x(t) - \mu_0 + c\big)^+,\ t > 1, \tag{12}$$

where μ_0 is defined as the mean over R_X, $c > 0$ is a tuning parameter, and $\mathtt{S}_u(0) = 0$. The score function $\mathtt{S}_u$ detects positive shifts in the mean of the stream; to detect shifts in both directions, a second score function $\mathtt{S}_d(t) = \big(\mathtt{S}_d(t-1) - x(t) + \mu_0 + c\big)^+$ has to be simultaneously monitored. The stopping time of the NP-CUSUM test is defined as

$$\widehat{T} = \min\left\{t : \mathtt{S}_u(t) > \kappa \text{ or } \mathtt{S}_d(t) > \kappa\right\}, \tag{13}$$

being κ another tuning parameter.

Validation Layer: The validation layer implements a CPM based on the Mann-Whitney statistics $\mathcal{U}$ to assess location changes in the distribution of data inside V, thus also those changes in the mean that the NP-CUSUM is designed to detect. Here, the training sequence R_X is used to compute μ_0 and build V.

The computational burden of the detection layer is rather negligible, as it requires only 8 operations per input data. The bulky component of this HCDT is the CPM at the detection layer, which has to compute $(n-1)$-times the statistic $\mathcal{U}$ that, as previously discussed, requires sorting values in V_0 and V_1.

VI. Experiments

To quantitatively assess the advantages of HCDTs over their single-layered counterparts (in what follows simply referred to as CDTs), we designed two experiments[5]. The first one is meant to illustrate the superior performance of HCDTs when detecting a subtle abrupt change in a dataset of synthetically generated sequences. The second one refers to a dataset acquired from an industrial monitoring application, where each sequence exhibits six consecutive abrupt changes that has been prepared to investigate the effectiveness of HCDTs reconfiguration. Experiments are performed by using the HCDTs presented in Section V.

A. Test on Synthetically Generated DataSet

This dataset contains 10000 sequences defined as

$$x(t) \sim \begin{cases} \mathcal{N}(1,1) & t \leq 30000 \\ \mathcal{N}(1.5,1) & 30000 < t \leq 60000 \end{cases}, \tag{14}$$

to test the detection of a subtle change in the mean of ϕ_0 (namely 0.5 times the standard deviation of ϕ_0). We generate sequences that are long-enough after T^* to make sure that the change is –in practice– always detected. All the HCDTs in Section V have been configured on a training sequence containing the first 400 samples, and are able to detect changes in the expectation of a random variable. Since the stationary data are i.i.d. as in (1), there is no need of preprocessing.

The figures of merit to assess detection performance are the FPR and DD, defined in Section II. When comparing HCDTs and CDTs it has to be considered that the performance of both algorithms depend on specific tuning parameters regulating the FPR *vs* DD trade-off (namely, κ for the CUSUM and NP-CUSUM tests, Γ for the ICI-based CDT). Therefore, the performance of HCDTs and CDTs are assessed by comparing the FPR *vs* DD curves obtained by considering a suitable range of values for the tuning-parameters. In particular, we tested the CUSUM using $\kappa \in \{7, \ldots, 13\}$, the ICI-based CDT using $\Gamma \in \{1.5, 1.75, \ldots, 3\}$ and the NP-CUSUM using $\kappa \in \{50, 100, 150, 250, 350, 450, 550, 750, 950\}$, since these values covered a suitable region of the (FPR, DD) plot. We manually tuned c in the NP-CUSUM test and set $c = 0.1$ to guarantee suitable detection performance for the range of change magnitudes considered in [16]. All the hypothesis tests (i.e, the z-test, the Hotelling T-square, the Lepage CPM and the Mann-Withney CPM) were configured by setting the confidence $\alpha = 0.05$, namely the probability of type I errors in hypothesis testing. To enable a fair comparison, the DD of HCDTs is computed only in those sequences where the CDT had no false positives.

To illustrate the advantages of using HCDTs, we report the number of false-positive detections discarded by the validation layer. In particular we compute, for each value of CDT parameter, the average number of validation-layer activations by false-positive detections (averages are computed over 1000 samples). Even though this value heavily depends on the FPR of the detection layer (the larger the number of detections, the larger the number of validation-layer activations), this figure of merit provides: *i)* a qualitative assessment of how often false-positive detections are discarded when monitoring long sequences, and *ii)* an indication of the computational overhead introduced by the HCDTs. These values, together with comments reported in Section V about the computational load of the considered HCDTs, should be taken into account when configuring the HCDTs.

Figure 3 indicates that the curve of hierarchical CUSUM coincides with that of the CUSUM test at the detection layer, and that the validation layer is never activated for discarding a detection raised by the CUSUM test. This is because the likelihood ratio (6) at the detection layer is a very powerful statistics, which makes the validation introduced by the z-test useless[6].

In contrast, introducing a separate validation layer is clearly beneficial in nonparametric scenarios, as demonstrated by the distance between the FPR *vs* DD curves in Figures 4 and 5. The curves of the 25th and 75th percentiles of the empirical distribution of $\widehat{T}$ (for a given value of FPR) indicate that the

[5] The comparison among different HCDTs is not in the scope of this paper.

[6] The effectiveness of the CUSUM can be also appreciated by comparing the DD of this parametric solution against the nonparametric HCDTs in Figures 4 and 5.

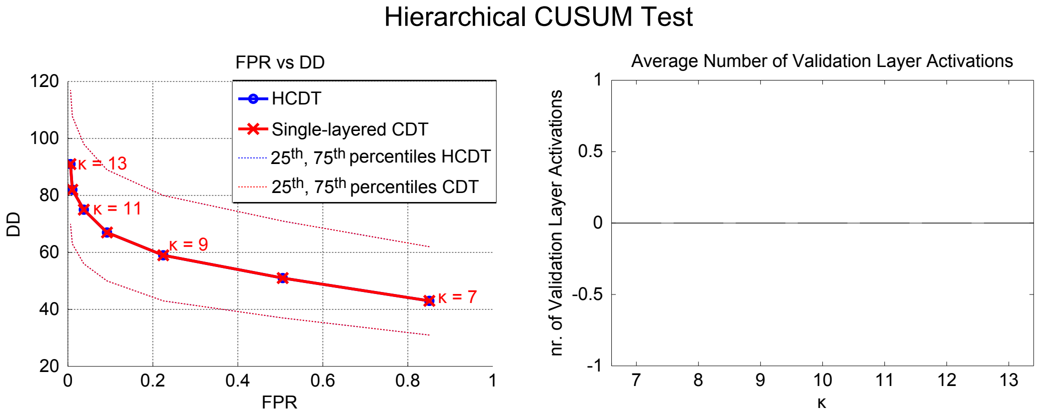


Fig. 3. The FPR *vs* DD curve computed for the CUSUM test and the hierarchical CUSUM test when $\kappa \in \{7, \ldots, 13\}$ (left); the average number of activations of the validation layer over 1000 samples (right). The overlap between the FPR *vs* DD curves and the lack of validation layer activations before detection indicate that, in this parametric scenario, introducing a separate validation layer yields no improvements: the z-test always confirm the detection raised by the CUSUM test at the detection layer, because the likelihood ratio (6) is a more powerful statistic.

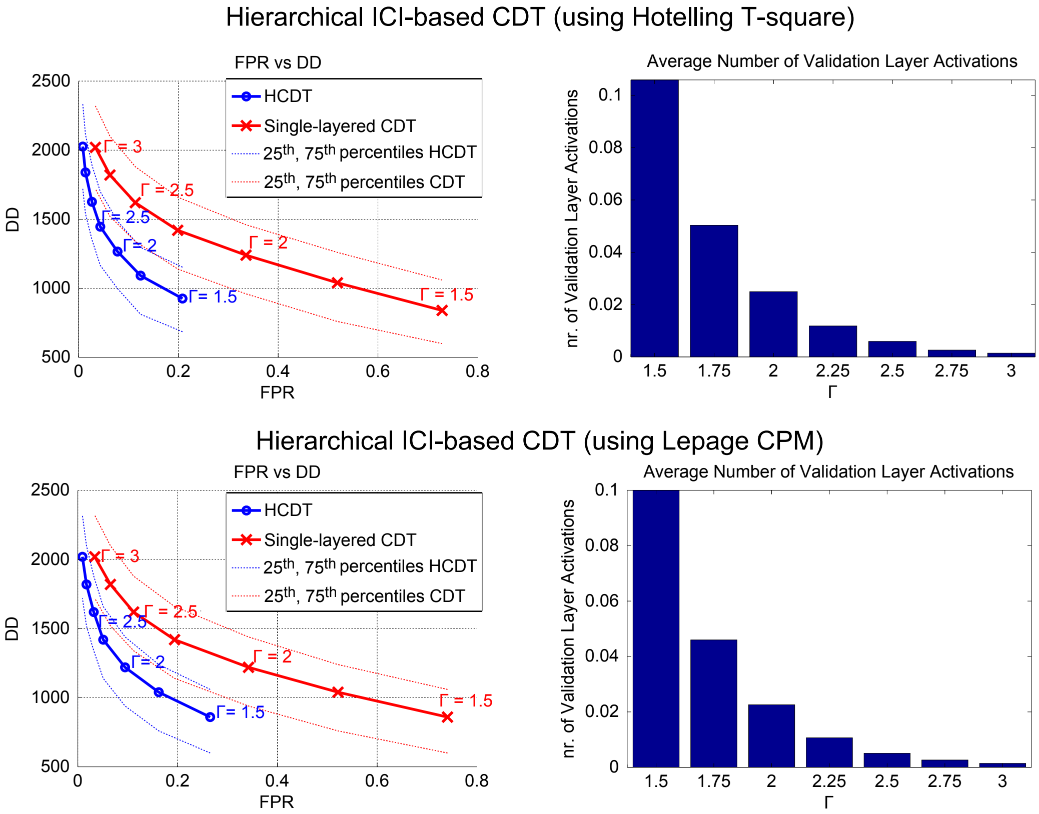


Fig. 4. The FPR *vs* DD curve computed for the ICI-based CDT and the hierarchical ICI-based CDT when $\Gamma \in \{1.5, 1.75, \ldots, 3\}$ (left); the average number of activations of the validation layer over 1000 samples (right). In this nonparametric scenario, the HCDT provides a marked improvement over its single-layer counterpart. We report the performance of both solutions using the HT based on the Hotelling T-square statistic and the Lepage CPM at the validation layer.

performance gap between the two solutions is substantial, and cannot be simply achieved by adjusting the tuning parameters of the CDT at the detection layer. In these cases, the validation layer is often activated to discard false-positive detections: this is particularly evident at low values of Γ or κ, where the CDT at the detection layer operates at large FPR values. Figure 4 reports both the HCDTs based on the Lepage CPM and the Hotelling T-square test at the validation layer. The FPR *vs* DD curves of the two HCDTs are very similar, with a lower number of validation layer activations required by the Lepage CPM, which is probably more powerful for validating changes in V. However, as remarked in Section V-B, the validation layer implementing the Lepage CPM is far more computationally demanding than the validation layer based on the HT based on the Hotelling T-square statistic. The comparison between NP-CUSUM and hierarchical NP-CUSUM tests in Figure 5 confirms the advantages provided the additional validation layer when performing nonparametric

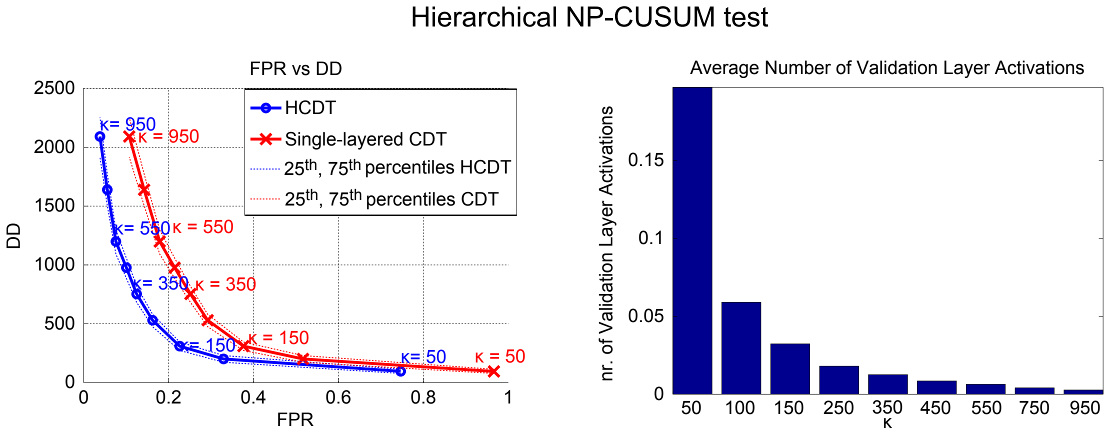


Fig. 5. The FPR *vs* DD curve computed for the NP-CUSUM and the hierarchical NP-CUSUM tests when $\kappa \in \{50, 100, 150, 250, 350, 450, 550, 750, 950\}$ (left); the average number of activations of the validation layer over 1000 samples (right). As in Figure 4, the HCDT achieves a marked improvement over its single-layer counterpart.

monitoring.

B. Test on an Industrial Dataset

This dataset contains 1000 sequences of photodiodes measurements acquired by an X-ray machine for industrial monitoring and safety inspection. Each sequence includes 21000 samples and has an abrupt change every 3000 samples. These sequences were prepared to yield changes having similar magnitude[7]. In these sequences, changes typically affect also the shape of the data distribution (e.g., the skewness might also change and some peaks appear in the pdf after the change) as shown in the illustrative example in the first two rows of Figure 6. Note that these histograms refer to the sequence provided as example in the second row, and that other sequences of the dataset are generated from different distributions. No preprocessing on these sequences is needed since, thanks to the specific acquisition process, data can be properly described by (1).

In this experiment we considered, as a reference example, the HCDT exploiting the ICI-based CDT and the Hotelling T-square statistic (see Section V-B), where we set $\Gamma = 2.5$ and $\alpha = 0.05$. The initial training sequence contains 160 samples and this is also the minimum number of samples required in R to reconfigure the HCDT after each validated change.

The empirical distributions of $\widehat{T}$ and $\widehat{T}^*$ for the hierarchical CDT are reported in the third and fourth rows of Figure 6, and indicate that the reconfiguration phase of this HCDT was very successful. In fact, the change-detection performance is stable and does not degrade when several changes arrive, since the distribution of detection times and of change-point estimates are very similar for all changes. Table I confirms that the DD and the FPR are in practice constant for all the six changes for the HCDT.

TABLE I
CHANGE-DETECTION PERFORMANCE ON THE INDUSTRIAL DATASET

	HCDT			**HCDT (reconfiguration only)**		
change nr	**DD**	**FPR**	**FNR**	**DD**	**FPR**	**FNR**
1	246.9	2.2%	0.2%	246.6	11.0%	0.2%
2	230.0	2.9%	0.3%	227.8	11.5%	0.1%
3	238.1	2.6%	0.3%	237.5	13.0%	0.2%
4	227.1	1.6%	0.4%	227.0	11.8%	0.4%
5	234.0	2.6%	0.1%	233.8	10.7%	0.1%
6	239.9	2.3%	0.3%	239.3	12.7%	0.3%

To remark the importance of performing change validation, we report the performance of the same hierarchical CDT where the validation layer always confirms a detection. In practice, this HCDT leverages only the reconfiguration mechanism, and for this reason we refer to *HCDT reconfiguration only* in Figure 6 and Table I. The values of FPR reported Table I shows that the validation layer at the HCDT has discarded several several false-positive detections, and the same emerges when comparing the empirical distribution of HCDT and HCDT reconfiguration only (third and fifth rows of Figure 6, respectively).

C. Remarks

It is worth mentioning that, given a specific configuration for a CDT, any HCDT implementing the same CDT at the detection layer cannot achieve lower DD than its single-layered counterpart. This clearly emerges in Table I, where the DD of the HCDT are larger HCDT reconfiguration only, which does not perform change validation. In fact, introducing the validation layer might eventually increase (while surely not decrease) the DD, due to false negatives of the HT or the request of additional samples in V (see Algorithm 2). False negatives of the HT might increase the false negative rate (FNR) of the HCDT that is also possibly larger than its single-layered counterpart (and in general larger than solutions not performing change-validation, as shown in Table I[8]).

[7] In particular, if we denote by μ_0 and σ_0 the mean and the standard deviation of the empirical distributions before the change (and by μ_1 and σ_1 for the post-change mean and standard deviation), we have selected only changes that satisfy the following conditions: $2 < \frac{(\mu_0 - \mu_1)^2}{\sigma_0^2} < 4$ and $2 < \frac{\sigma_0^2}{\sigma_1^2} < 4$.

[8] In the industrial dataset sequences are not long enough to avoid false-negative detections and FNR is sometimes different from zero.

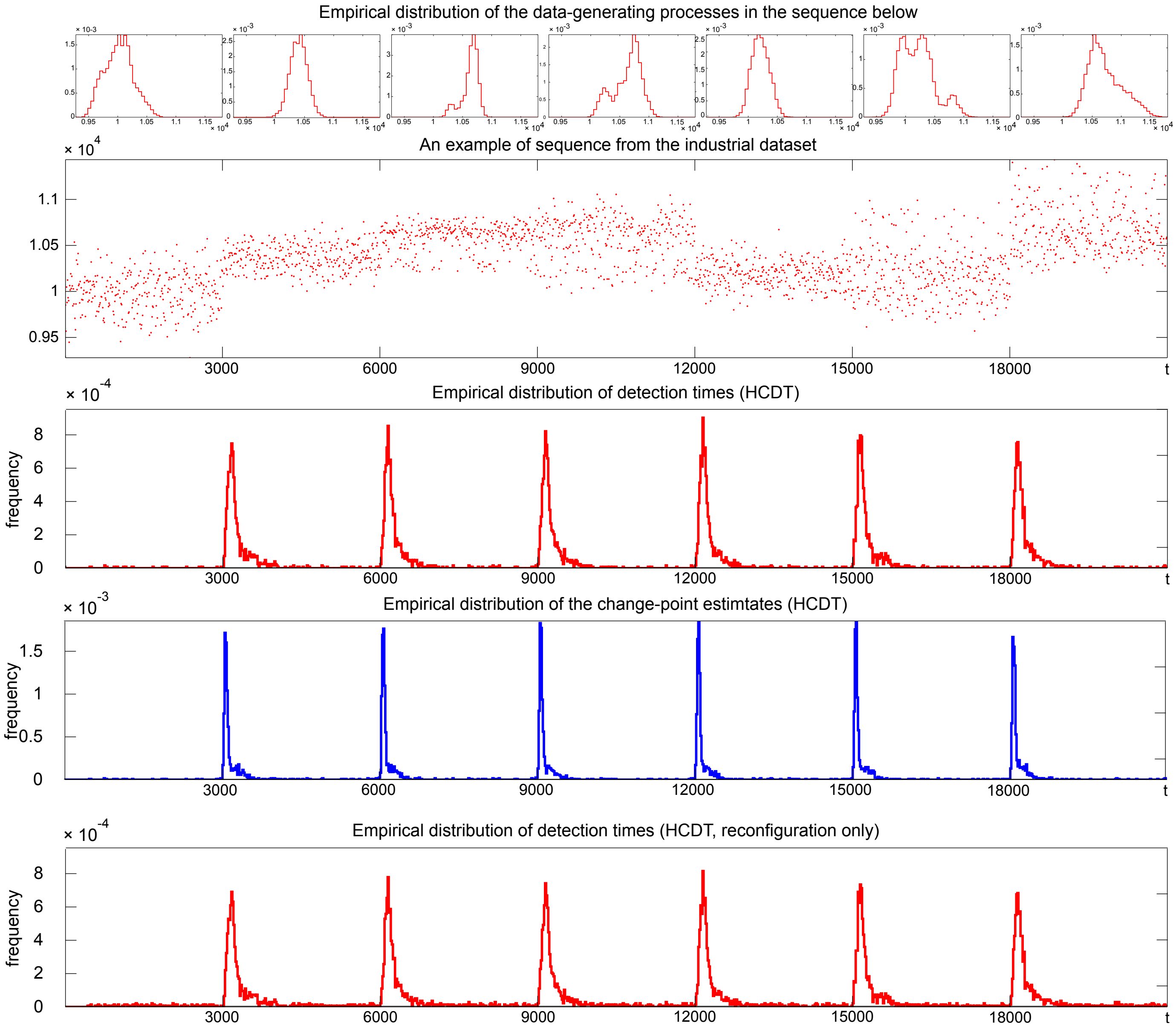


Fig. 6. First row: empirical distributions of the data-generating process in a sequence from the industrial monitoring dataset (reported in the second row). Third row: the empirical distribution of the detections $\widehat{T}$ of the hierarchical ICI-based CDT using the HT based on the Hotelling T-square statistic, and the estimated change-points $\widehat{T}*$ (fourth row). Both these histograms show that the reconfiguration of the HCDT is always successful, since all the changes are detected with similar performance (check also Table I). The fifth row refers to the same HCDT where the detection layer always confirm the detections raised by the CDT at the first layer (HCDT, reconfiguration only). In this case, the peaks of the histograms are lower and detections are more spread in stationary regions, indicating a large number of false-positive detections (as shown in the FPR columns in Table I).

This increase in DD (and FNR) is not in contrast with the results shown in Figures 3 - 5, because the dramatic reduction of FPR provided by the validation layer makes it possible to configure the CDT at the detection layer to yield very prompt detections, while still guaranteeing acceptable values of the overall FPR of the HCDT. This clearly emerges in the comparison of the FPR *vs* DD curves in Figures 3 - 5.

Another important remark is that, although HTs operate at a predefined percentage α of false positives (type I errors), their use at the validation layer typically results in a percentage of false positives larger than α. This is due to the fact that, in HCDTs, the hypothesis test is activated on sequences V that have been previously *selected* by the detection layer, thus cannot be considered as drawn from the distribution generating i.i.d. sequences of i.i.d. samples (where the control over type I errors applies). Nevertheless, even though α does not exactly correspond to the probability of type I errors, it still can be used to tune the HT.

VII. Conclusions

We have presented a general methodology for designing hierarchical change-detection tests, powerful change-detection algorithms characterized by a two-layered architecture that enables the validation of each detected change. Our experiments demonstrate that introducing such validation phase is often beneficial and that HCDTs achieve a marked improvement over traditional, nonparametric, CDTs. Furthermore, HCDTs can track evolving processes since they are naturally able to reconfigure after each change to detect further departures from the new, post-change conditions. In a broad sense, HCDTs provide an abstract processing level to make intelligent systems adaptive in dynamic and evolving environments.

The combination of a prompt and computationally-light CDT at the detection layer with a more sophisticated HT at the validation layer makes HCDTs suitable for operating on datastreams, addressing the emerging big data scenarios. Remarkably, the peculiar architecture of HCDTs recalls the emotional processes in the human brain, where different regions of the brain, characterized by different activation times and response accuracies, interact.

References

[1] C. Alippi, G. Boracchi, and M. Roveri, “Just-in-time classifiers for recurrent concepts,” *Neural Networks and Learning Systems, IEEE Transactions on*, vol. 24, no. 4, pp. 620–634, April.

[2] M. Basseville and I. V. Nikiforov, *Detection of abrupt changes: theory and application*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993.

[3] J. a. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, “A survey on concept drift adaptation,” *ACM Comput. Surv.*, vol. 46, no. 4, pp. 44:1–44:37, Mar. 2014. [Online]. Available: http://doi.acm.org/10.1145/2523813

[4] G. Boracchi, M. Michaelides, and M. Roveri, “A cognitive monitoring system for contaminant detection in intelligent buildings,” in *International Joint Conference on Neural Networks (IJCNN 2014)*, 6 - 11 July 2014, pp. 1–8.

[5] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, “Learning with drift detection,” in *Advances in Artificial Intelligence SBIA 2004*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2004, vol. 3171, pp. 66–112.

[6] C. Alippi and M. Roveri, “Just-In-Time adaptive classifiers – part II: Designing the classifier,” *IEEE Transactions on Neural Networks*, vol. 19, no. 12, pp. 2053 –2064, dec. 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] C. Hamacher, Z. Vranesic, and S. Zaky, *Computer organization*. McGraw-Hill, 2002.

[9] K. Toyama and G. D. Hager, “Incremental focus of attention for robust vision-based tracking,” *International Journal of Computer Vision*, vol. 35, no. 1, pp. 45–63, 1999.

[10] J. Šochman and J. Matas, “Waldboost-learning for time constrained sequential detection,” in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2. IEEE, 2005, pp. 150–156.

[11] C. Alippi, R. Camplani, and M. Roveri, “An adaptive llc-based and hierarchical power-aware routing algorithm,” *Instrumentation and Measurement, IEEE Transactions on*, vol. 58, no. 9, pp. 3347–3357, 2009.

[12] R. Gupta, T. R. Koscik, A. Bechara, and D. Tranel, “The amygdala and decision-making,” *Neuropsychologia*, vol. 49, no. 4, pp. 760–766, 2011.

[13] K. N. Ochsner and L. Feldman Barrett, “A multiprocess perspective on the neuroscience of emotion,” *Emotion: Current issues and future directions*, pp. 38–81, 2001.

[14] C. Alippi, “Intelligence for embedded systems, a methodological approach,” p. 283, 2014.

[15] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldá, and R. Morales-Bueno, “Early drift detection method,” in *In Fourth International Workshop on Knowledge Discovery from Data Streams*, 2006.

[16] C. Alippi, G. Boracchi, and M. Roveri, “A hierarchical, nonparametric, sequential change-detection test,” in *International Joint Conference on Neural Networks (IJCNN 2011)*, 31 2011-aug. 5 2011, pp. 2889 –2896.

[17] D. M. Hawkins, P. Qiu, and C. W. Kang, “The changepoint model for statistical process control,” *Journal of Quality Technology*, vol. Vol. 35, No. 4, pp. 355–366, 2003.

[18] T. Hastie, R. Tibshirani, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations*. Springer-Verlag, New York, February 2009.

[19] C.-S. J. Chu, M. Stinchcombe, and H. White, “Monitoring structural change,” *Econometrica*, vol. 64, no. 5, pp. 1045–65, September 1996.

[20] F. Gustafsson, *Adaptive Filtering and Change Detection*. Wiley, Oct. 2000. [Online]. Available: http://www.wiley.com/WileyCDA/WileyTitle/productCd-0471492876,descCd-description.html

[21] L. Ljung, *System identification: theory for the user*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1986.

[22] C. Alippi, G. Boracchi, and B. Wohlberg, “Change detection in streams of signals with sparse representations,” in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2014, pp. 5252 – 5256.

[23] 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.

[24] C. Alippi, G. Boracchi, and M. Roveri, “An effective Just-In-Time adaptive classifier for gradual concept drifts,” in *International Joint Conference on Neural Networks (IJCNN 2011)*, 31 2011-aug. 5 2011, pp. 1675 –1682.

[25] G. Boracchi and M. Roveri, “Exploiting self-similarity for change detection,” in *Neural Networks (IJCNN), 2014 International Joint Conference on*, July 2014, pp. 3339–3346.

[26] G. Ditzler and R. Polikar, “Hellinger distance based drift detection for nonstationary environments,” in *Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2011 IEEE Symposium on*, April 2011, pp. 41–48.

[27] R. Elwell and R. Polikar, “Incremental learning in nonstationary environments with controlled forgetting,” in *International Joint Conference on Neural Networks (IJCNN 2009)*, june 2009, pp. 771 –778.

[28] E. S. Page, “Continuous inspection schemes,” *Biometrika*, vol. 41, no. 1/2, pp. pp. 100–115, 1954. [Online]. Available: http://www.jstor.org/stable/2333009

[29] A. Polunchenko and A. Tartakovsky, “State-of-the-art in sequential change-point detection,” *Methodology and Computing in Applied Probability*, vol. 14, no. 3, pp. 649–684, 2012. [Online]. Available: http://dx.doi.org/10.1007/s11009-011-9256-5

[30] A. G. Tartakovsky, B. L. Rozovskii, R. B. Blaek, and H. Kim, “Detection of intrusions in information systems by sequential change-point methods,” *Statistical Methodology*, vol. 3, no. 3, pp. 252 – 293, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1572312705000493

[31] 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, july 2008.

[32] A. Goldenshluger and A. Nemirovski, “On spatial adaptive estimation of nonparametric regression,” *Math. Meth. Statistics*, vol. 6, pp. 135–170, 1997.

[33] V. Katkovnik, K. Egiazarian, and J. Astola, “Adaptive window size image de-noising based on intersection of confidence intervals (ICI) rule,” *J. of Math. Imaging and Vision*, vol. 16, pp. 223–235, 2002.

[34] Y. Kawahara and M. Sugiyama, “Sequential change-point detection based on direct density-ratio estimation,” *Statistical Analysis and Data Mining*, vol. 5, no. 2, pp. 114–127, 2012. [Online]. Available: http://dx.doi.org/10.1002/sam.10124

[35] F. J. J. Massey, “The kolmogorov-smirnov test for goodness of fit,” *Journal of the American Statistical Association*, vol. 46, no. 253, pp. 68–78, 1951. [Online]. Available: http://www.jstor.org/stable/2280095

[36] T. W. Anderson, “On the distribution of the two-sample Cramer-von Mises criterion,” *The Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 1148–1159, 1962.

[37] J. D. Gibbons and S. Chakraborti, *Nonparametric Statistical Inference (Statistics: a Series of Textbooks and Monogrphs)*, 4th ed. CRC, May 2003. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0824740521

[38] R. Johnson and D. Wichern, *Applied multivariate statistical analysis*. Prentice Hall, 2002, no. v. 1.

[39] M. S. Bartlett and D. G. Kendall, “The statistical analysis of variance-heterogeneity and the logarithmic transformation,” *Supplement to the Journal of the Royal Statistical Society*, vol. 8, no. 1, pp. pp. 128–138, 1946. [Online]. Available: http://www.jstor.org/stable/2983618

[40] H. B. Mann and D. R. Whitney, “On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other,” *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, 1947. [Online]. Available: http://dx.doi.org/10.2307/2236101

[41] A. M. Mood, “On the asymptotic efficiency of certain nonparametric two-sample tests,” *The Annals of Mathematical Statistics*, vol. Vol. 25, No. 3, pp. 514–522, September 1954.

[42] Y. Lepage, “A combination of Wilcoxon’s and Ansari-Bradley’s statistics,” *Biometrika*, vol. Vol. 58, No. 1, pp. 213–217, April 1974.

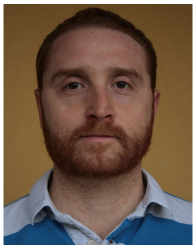
[43] S. Liu, M. Yamada, N. Collier, and M. Sugiyama, “Change-point detection in time-series data by relative density-ratio estimation,” *Neural Networks*, vol. 43, no. 0, pp. 72 – 83, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608013000270

[44] G. Ross and N. M. Adams, “Two nonparametric control charts for detecting arbitrary distribution changes,” *Journal of Quality Technology*, vol. Vol 44, No. 22, pp. 102–116, 2012.

[45] C. Alippi, G. Boracchi, and M. Roveri, “Change detection tests using the ICI rule,” in *International Joint Conference on Neural Networks (IJCNN 2010)*, 2010, pp. 1 –7.

[46] G. S. Mudholkar and M. C. Trivedi, “A Gaussian approximation to the distribution of the sample variance for nonnormal populations,” *Journal of the American Statistical Association*, vol. 76, no. 374, pp. pp. 479–485, 1981.

Cesare Alippi received the degree in electronic engineering cum laude in 1990 and the PhD degree in 1995 from the Politecnico di Milano, Italy. He is currently a full professor of information processing systems at the Politecnico di Milano. He has been a visiting researcher at UCL (United Kingdom), MIT, ESPCI (F), CASIA (RC), and USI (CH). In 2004, he received the IEEE Instrumentation and Measure- ment Society Young Engineer Award. In 2011, he received the Knight of the Order of Merit of the Italian Republic. In 2013, he received the IBM Faculty Award. In 2016, he received the Outstanding paper award of the IEEE Transactions on Neural Networks and Learning Systems Current research activity addresses adaptation and learning in nonstationary environments and intelligent embedded systems. He is a fellow of the IEEE, Board of Governors member of the International Neural Network Society, Vice-President education of the IEEE Computational Intelligence Society (CIS), past AE of the IEEE Transactions on Instrumentation and Measurements, IEEE Transactions on Neural Networks, IEEE Computational Intelligence Magazine, and member and chair of other IEEE com- mittees including the IEEE Rosenblatt award.



Giacomo Boracchi received the M.S. degree in Mathematics from the Universitaà Statale degli Studi di Milano, Italy, and the Ph.D. degree in Information Technology at Politecnico di Milano, Italy, in 2004 and 2008, respectively. He was researcher at Tampere International Center for Signal Processing, Finland, during 2004-2005.

Currently, he is an assistant professor at the Dipartimento di Elettronica, Informazione e Bioingegneria of the Politecnico di Milano. His main research interests are learning methods for nonstationary environments, as well as mathematical and statistical methods for signal and image processing. In 2015 he received the IBM Faculty Award.

Manuel Roveri received the Dr.Eng. degree in Computer Science Engineering from the Politecnico di Milano (Milano, Italy) in June 2003, the MS in Computer Science from the University of Illinois at Chicago (Chicago, Illinois, U.S.A.) in December 2003 and the Ph.D. degree in Computer Engineering from Politecnico di Milano (Milano, Italy) in May 2007.

Currently, he is an Associate Professor at the Dipartimento di Elettronica, Informazione e Bioingegneria of the Politecnico di Milano. His research interests include intelligent embedded and cyber-physical systems, learning in nonstationary environments, adaptive algorithms, model-free fault detection and diagnosis systems and exploratory computing. Manuel Roveri is an Associate Editor of the IEEE Transactions on Neural Networks and Learning Systems and served as chair and member in many IEEE subcommittees.