
PRIFYSGOL
BANGOR
UNIVERSITY

School of Computer Science
College of Physical & Applied Sciences

# Unsupervised Change Detection in Multivariate Streaming Data

---

William J. Faithfull

Submitted in partial satisfaction of the requirements for the
Degree of Doctor of Philosophy
in Computer Science

*Supervisor* Prof. Ludmila I. Kuncheva

# Acknowledgements

> *An approximate answer to the right problem is worth a good deal more than an exact answer to an approximate problem.*
>
> — **John Tukey**

It is a great privilege to have reached the point where I am writing acknowledgements, a privilege I did not always think I would achieve. Being here, I am profoundly grateful to Bangor University for my fee waiver, without which I would not have even been able to consider such a commitment. Taking on this endeavour on a part time basis has been a challenging and at times, isolating experience. It is to the great credit of the people around me that my spirit remained unbroken.

There are too many acknowledgements to list individually. My thanks go out to so many people. My parents for their support. My colleagues in the Computer Science department, with whom I taught and worked for 4 years. My friends who have been there for me throughout – especially Francis, Joe, Marc and Tud. My supervisor, Lucy. An incredible researcher, teacher and a relentless force for self-belief and high standards, who I count as a very dear friend. Finally, my partner, Liz. She has been unflappable, when I was not. She's endured this black hole into which I throw my time, and kept me on the rails. I couldn't have done it without you.

# Declaration and Consent

**Details of the Work**

I hereby agree to deposit the following item in the digital repository maintained by Bangor University and/or in any other repository authorized for use by Bangor University.

**Author Name:** ……..

**Title:** ……..

**Supervisor/Department:** ……..

**Funding body (if any):** ……..

**Qualification/Degree obtained:** ……..

This item is a product of my own research endeavours and is covered by the agreement below in which the item is referred to as "the Work". It is identical in content to that deposited in the Library, subject to point 4 below.

**Non-exclusive Rights**

Rights granted to the digital repository through this agreement are entirely non-exclusive. I am free to publish the Work in its present version or future versions elsewhere.

I agree that Bangor University may electronically store, copy or translate the Work to any approved medium or format for the purpose of future preservation and accessibility. Bangor University is not under any obligation to reproduce or display the Work in the same formats or resolutions in which it was originally deposited.

**Bangor University Digital Repository**

I understand that work deposited in the digital repository will be accessible to a wide variety of people and institutions, including automated agents and search engines via the World Wide Web.

I understand that once the Work is deposited, the item and its metadata may be incorporated into public access catalogues or services, national databases of electronic theses and dissertations such as the British Library's EThOS or any service provided by the National Library of Wales.

I understand that the Work may be made available via the National Library of Wales Online Electronic Theses Service under the declared terms and conditions of use (http://www.llgc.org.uk/index.php?id=4676). I agree that as part of this service the National Library of Wales may electronically store, copy or convert the Work to any approved medium or format for the purpose of future preservation and accessibility. The National Library of Wales is not under any obligation to reproduce or display the Work in the same formats or resolutions in which it was originally deposited.

**Statement 1:**

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree unless as agreed by the University for approved dual awards.

Signed …….. (candidate)

Date ……..

**Statement 2:**

This thesis is the result of my own investigations, except where otherwise stated. Where correction services have been used, the extent and nature of the correction is clearly marked in a footnote(s).

All other sources are acknowledged by footnotes and/or a bibliography.

Signed …….. (candidate)

Date ……..

**Statement 3:**

I hereby give consent for my thesis, if accepted, to be available for photocopying, for inter-library loan and for electronic storage (subject to any constraints as defined in statement 4), and for the title and summary to be made available to outside organisations.

Signed …….. (candidate)

Date ……..

**NB:** Candidates on whose behalf a bar on access has been approved by the Academic Registry should use the following version of **Statement 3:**

**Statement 3 (bar):**

I hereby give consent for my thesis, if accepted, to be available for photocopying, for inter-library loans and for electronic storage (subject to any constraints as defined in statement 4), after expiry of a bar on access.

Signed …….. (candidate)

Date ……..

**<u>Statement 4</u>:**

Choose **<u>one</u>** of the following options

| | | |
|---|---|---|
| a) | I agree to deposit an electronic copy of my thesis (the Work) in the Bangor University (BU) Institutional Digital Repository, the British Library ETHOS system, and/or in any other repository authorized for use by Bangor University and where necessary have gained the required permissions for the use of third party material. | |
| b) | I agree to deposit an electronic copy of my thesis (the Work) in the Bangor University (BU) Institutional Digital Repository, the British Library ETHOS system, and/or in any other repository authorized for use by Bangor University when the approved **bar on access** has been lifted. | |
| c) | I agree to submit my thesis (the Work) electronically via Bangor University's e-submission system, however I **opt-out** of the electronic deposit to the Bangor University (BU) Institutional Digital Repository, the British Library ETHOS system, and/or in any other repository authorized for use by Bangor University, due to lack of permissions for use of third party material. | |

*Options B should only be used if a bar on access has been approved by the University.*

**In addition to the above I also agree to the following:**

1. That I am the author or have the authority of the author(s) to make this agreement and do hereby give Bangor University the right to make available the Work in the way described above.

2. That the electronic copy of the Work deposited in the digital repository and covered by this agreement, is identical in content to the paper copy of the Work deposited in the Bangor University Library, subject to point 4 below.

3. That I have exercised reasonable care to ensure that the Work is original and, to the best of my knowledge, does not breach any laws – including those relating to defamation, libel and copyright.

4. That I have, in instances where the intellectual property of other authors or copyright holders is included in the Work, and where appropriate, gained explicit permission for the inclusion of that material in the Work, and in the electronic form of the Work as accessed through the open access digital repository, *or* that I have identified and removed that material for which adequate and appropriate permission has not been obtained and which will be inaccessible via the digital repository.

5. That Bangor University does not hold any obligation to take legal action on behalf of the Depositor, or other rights holders, in the event of a breach of intellectual property rights, or any other right, in the material deposited.

6. That I will indemnify and keep indemnified Bangor University and the National Library of Wales from and against any loss, liability, claim or damage, including without limitation any related legal fees and court costs (on a full indemnity bases), related to any breach by myself of any term of this agreement.

Signature: …… Date : ……..

# Abstract

Change Detection and its closely associated sister fields provide fundamental components for many vital applications such as quality control, data mining, power distribution, network intrusion detection and adaptive classification.

There is a tremendous body of research in statistics, quality control, data mining and applied areas that has contributed to a diverse arsenal of change detectors. Whilst there has been a greater focus on the univariate problem, there are many approaches to the more challenging problem of multivariate change detection. Novel change detection methods continue to be actively developed.

Supervised change detection methods have a clear pathway to improvement, by training on labelled data. However, there are a number of problems for which abundant labelled data is scarce or unavailable. For these problems, an unsupervised approach must be taken using incoming data.

It is proposed here to develop general, composable modules to improve on the existing methods for unsupervised multivariate change detection. The modules should be composable such that they can all be applied together without interfering with each other.

This thesis proposes three such modules. Firstly, Principal Components Analysis (PCA) is assessed as a general purpose feature extraction and selection step. Secondly, it is proposed to chain univariate change detection methods to multivariate criteria, such that they act as adaptive thresholds. Finally, univariate change detectors are built into subspace ensembles where each detector

monitors a single feature of the input space, allowing them to function as a multivariate change detector. These three modules are jointly assessed against a challenging problem of unsupervised endogenous eye blink detection.

# Contents

# List of Figures

# List of Tables

# List of Symbols

| | |
|---|---|
| $\mathbf{X} = \{\vec{x}_1, \vec{x}_2, ...\}$ | Infinite data stream. |
| $\mathbf{Y} = \{y_1, y_2, ...\}$ | Class labels for data stream. |
| $\mathbf{Z} = \{z_1, z_2, ...\}$ | Data stream with class labels. |
| $t$ | Time index in data stream. |
| $p$ | Dimensionality of data stream. |
| $\vec{x}_t$ | Example vector at time $t$. |
| $x_t$ | Univariate example at time $t$. |
| $y$ | Class label. |
| $z_t = (x_t, y)$ | Univariate example and class label at time $t$. |
| $\mu$ | Mean. |
| $\sigma$ | Standard Deviation. |
| $\vec{\mu}$ | Multivariate mean. |
| $\mathbf{\Sigma}$ | Covariance matrix. |
| $\mu_t$ | Mean at time $t$. |
| $\sigma_t$ | Standard Deviation at time $t$. |
| $S_i$ | Data Source $i$. |
| $\mathcal{P}$ | Underlying generative process. |
| $p(\vec{x})$ | Prior probabilities of $\vec{x}$. |
| $p(y)$ | Prior probabilities of classes. |
| $p(\vec{x}\|y)$ | Class conditional *pdf*. |
| $p(y\|\vec{x})$ | Posterior probabilities. |
| $p(\vec{x}, t)$ | Prior probabilities of $\vec{x}$ at time $t$. |
| $p(y, t)$ | Prior probabilities of classes at time $t$. |
| $p(\vec{x}\|y, t)$ | Class conditional *pdf* at time $t$. |
| $p(y\|\vec{x}, t)$ | Posterior probabilities at time $t$. |

| | |
|---|---|
| $P$ | Distribution before proposed change point. |
| $Q$ | Distribution after proposed change point. |
| $\hat{P}$ | Empirical distribution before proposed change point. |
| $\hat{Q}$ | Empirical distribution after proposed change point. |
| $W_1$ | Trailing window in a pair of sliding windows. |
| $W_2$ | Leading window in a pair of sliding windows. |
| $\Delta$ | Change detector statistic. |
| $\lambda$ | Change detector threshold. |

# Chapter 1

# Introduction

## 1.1 Overview

Change detection in streaming data is a research area which borrows from many fields. Data mining and stream processing allow the maintenance of finite-sized windows from an infinite stream [176, 85, 74, 17], or streaming statistics [53]. Elements of machine learning and statistics are used to maintain probabilistic [187] and empirical models [162]. Information-theoretic measures [95] and distances and statistical tests [2, 135] offer definitions of change. Combinations of these components can be and have been applied to dozens of problem areas, resulting in numerous novel approaches.

We may first draw a distinction between *supervised* and *unsupervised* stream change detection. In the former, after processing each example we are afforded the true class label, and we can adapt our approach based on the error. In the latter situation, we have no class labels and must operate blindly on only the observed data. In this wholly unsupervised formula, change detection is used to identify unforseen events for which we do not have training examples, such as equipment failure or malicious network intrusion. This is also applicable to adaptive learning. Many stream classifiers make the assumption that data observations are random examples of a fixed statistical process. In practice, these algorithms are deployed in non-stationary environments where

the statistical process generating the examples evolves over time, so they must detect this evolution and adapt the deployed algorithm accordingly.

Unsupervised multivariate change detectors tend to be complex pipelines of operations, often involving feature extraction, distribution modelling and statistical tests among other components. This work is intended to develop general, composable alternatives which perform better than the current methods for unsupervised multivariate change detection. Their desiderata are (1) that they should make as few assumptions as possible in order to be widely applicable and (2) that the application of one should not necessitate or preclude the application of the others.

## 1.2 Research Hypothesis

The hypothesis of this thesis consists of five parts. It is proposed that (1) It is possible to detect change with sufficient accuracy from the distribution of unlabelled numerical data that is *i.i.d* under the null hypothesis. (2) Change detection approaches can be broken down into pipelines of operations. With these in mind, the remaining parts are formulated as follows. (3) Principal Component Analysis (PCA) is a beneficial and context-free feature extraction method for multivariate change detection. (4) A chain of one multivariate and one univariate change detector will perform better than the multivariate change detector alone. (5) An ensemble of univariate change detectors over multivariate data will perform better than a single multivariate change detector.

## 1.3 Thesis Structure

The aim of this work is to lend empirical support to all five parts of the hypothesis. The chapters are structured around the thesis objectives.

The objective of Chapter 2 is firstly to introduce the reader to the problem of change detection, provide the necessary background, and show how change can be detected in the absence of labels. Secondly, it demonstrates how change detectors can be deconstructed into pipelines of operations. Chapter 3 experimentally investigates the application of Principal Component Analysis. Chapter 4 experimentally investigates chains of detectors as an alternative to thresholding for multivariate change detectors. Chapter 5 experimentally investigates ensembles of univariate change detectors applied to multivariate data. Chapter 6 applies the techniques introduced in the previous chapters to the problem of endogenous eye-blink detection, demonstrating both their viability and composability. Finally, Chapter 7 introduces an open source Java library developed in the course of this thesis, which encapsulates the work from Chapters 2–5.

## 1.4 Contributions

The contributions offered by this thesis are as follows.

1. An overview of the field of change detection and associated problems.

2. A conceptual model of change detector modules, with taxonomies for each.

3. An experimental study on the use of Principal Component Analysis as a general-purpose feature extraction step for multivariate change detec-

tion problems. It is demonstrated that an unconventional use of PCA as an unsupervised feature extraction technique was strongly correlated with better change detection performance.

4. An experimental study on the hierarchical application of change detectors as an alternative to the specially crafted thresholds of each change detection technique. It is demonstrated that using a control chart threshold as opposed to a bootstrapping procedure was related to a lower false positive rate.

5. An experimental study on the use of a novel subspace ensemble to combine existing univariate change detectors over multivariate data. The novel ensembles considerably outperformed the pure multivariate change detectors they were compared with.

6. An experimental study of the previous three contributions when applied to unsupervised endogenous blink detection.

7. A Java library for the composition of change detectors from components, evaluation of change detectors and artificial change generation.

# Chapter 2

# Change Detection in Streaming Data

## 2.1 Introduction

We can define the change detection problem as follows. A data stream can be described as a potentially infinite sequence of vectors, $\mathbf{X} = \{\vec{x}_1, \vec{x}_2, ...\}$ where $\vec{x}_t$ is a vector of size $p$ arriving at time $t$. Suppose that these vectors are produced by a data source (or alternatively belong to a *Concept*), $S_1$, but that at some time, this data source is replaced with another, $S_2$. The objective is to identify from the data that the source has changed from $S_1$ to $S_2$. A univariate problem is illustrated in Figure 2.1, where observing a significant change in the mean would be sufficient to identify the change.

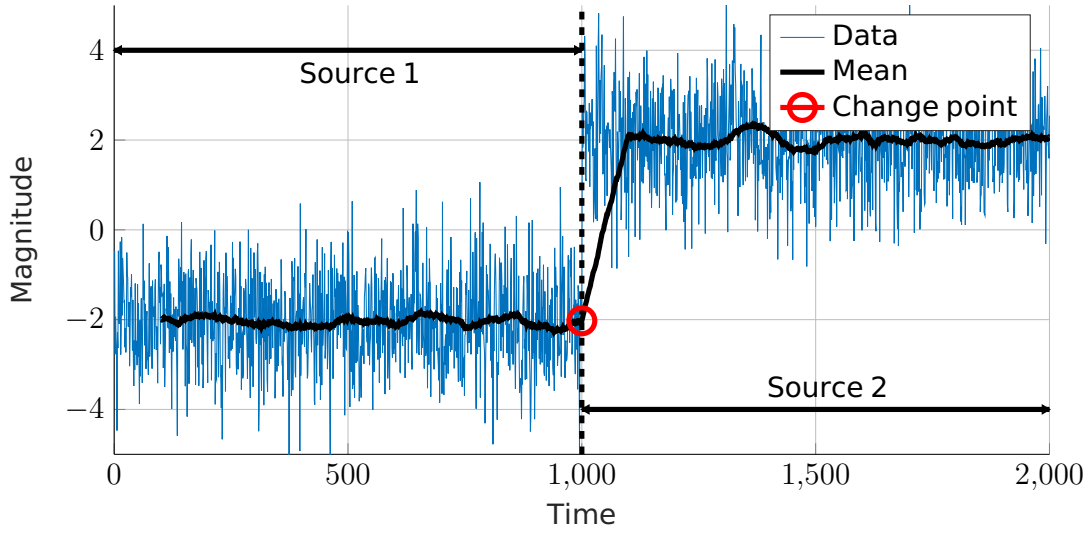



**Figure 2.1:** Sample streaming data with a change point at $t = 1000$ from Data Source 1 to Data Source 2.

Change Detection is used across a multitude of disciplines, especially those involved with the analysis of sequential data where the source of that data is impossible to model accurately, or is subject to unexpected change over time. With the ubiquity of smartphones, internet-attached devices, remote sensors and computer networks it is now extremely common for sources of data to be continuous, and arrive at applications in a streaming fashion [54, 51, 184]. By 'streaming', we refer to an ordered sequence of instances which can only be processed either individually or in relatively small batches. Data processing has traditionally involved a phase of data collection, and then operation on the resultant static dataset. Increasingly, these two phases have become interconnected, as much out of necessity of scale as out of convenience and automation.

The purpose of this chapter is to give the reader an in-depth introduction to the field of change detection and an understanding of the nature of the problem that it applies to, especially in the realm of streaming data. It will discuss general frameworks and assumptions that underpin many approaches. Section 2.2 discusses types of change and establishes how change is context-dependent. Section 2.3 overviews applications and related fields to change detection, and provides a historical perspective. Section 2.4 reviews taxonomies for change detection and adjacent fields, to show how approaches are organised in the literature. Section 2.5 builds on the existing taxonomies to show how change detection can be broken down into a pipeline of modules. Section 2.6 details building blocks and methods that are used in the course of this thesis. Section 2.7 is an overview of how change detection techniques can be evaluated. The chapter is summarised in Section 2.8.

## 2.2 The Characterisation and Context of Change

'Change' is a broad term, difficult to define and quantify, and highly dependent on the application at hand. Since change is context-dependent, it is useful

to characterise it so that we can reason about the *type* of change we wish to detect. This section will discuss how to quantify and reason about changes and offer a demonstration of how change varies on the context.

### 2.2.1 Types of Change

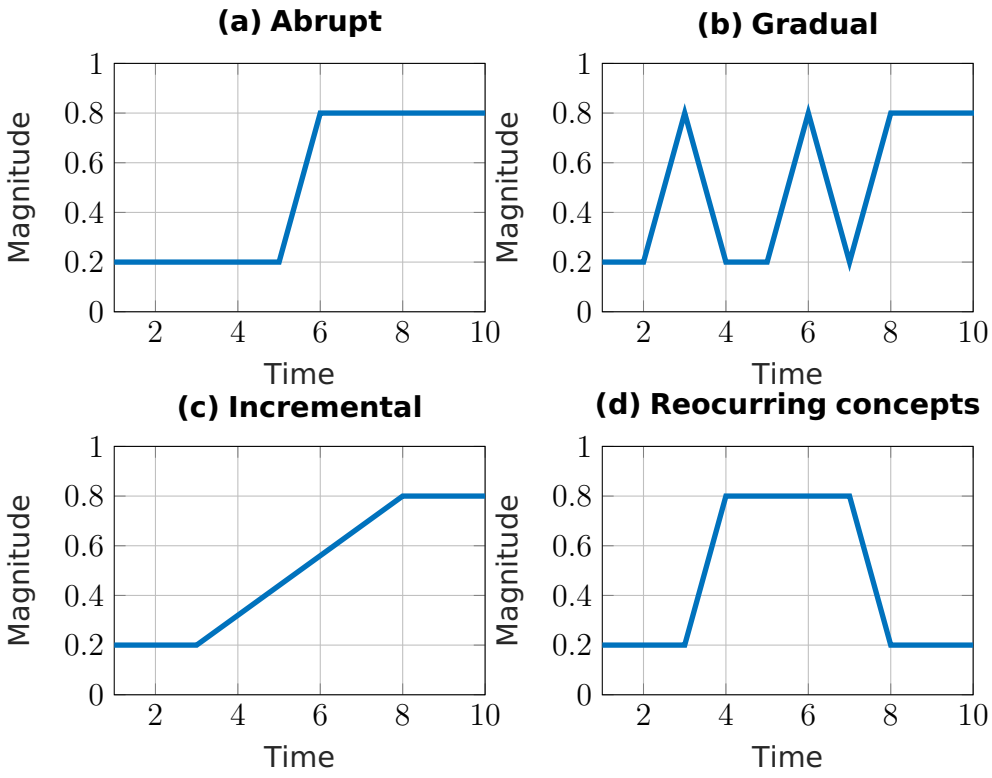



**Figure 2.2:** Illustration of types of changes over time between two sources, $S_1$ and $S_2$. [55, 186].

Gama [53] identifies two dimensions of analysis; the causes of change, and the rate of change. Consider first how a single variable in a sequence might change over time. There are four common patterns of change over time which are regularly discussed in the literature [55, 186], illustrated in a single variable in Figure 2.2.

An abrupt [176, 9, 84, 55] (*sudden* [186, 168, 41, 160, 55, 87], *revolutionary* [66], *substitution* [134]) change occurs where $S_1$ is replaced with $S_2$ at a single time point $t$, illustrated in Figure 2.2a at $t = 5$. An example of such a change might be a catastrophic sensor failure.

When a change occurs over a range of time points, it is referred to in the literature as a *gradual* change, although this term can imply one of two types of change. In the first type, points are sampled with increasing probability over time from $S_2$ and with decreasing probability from $S_1$, until the concept has changed completely to $S_2$.

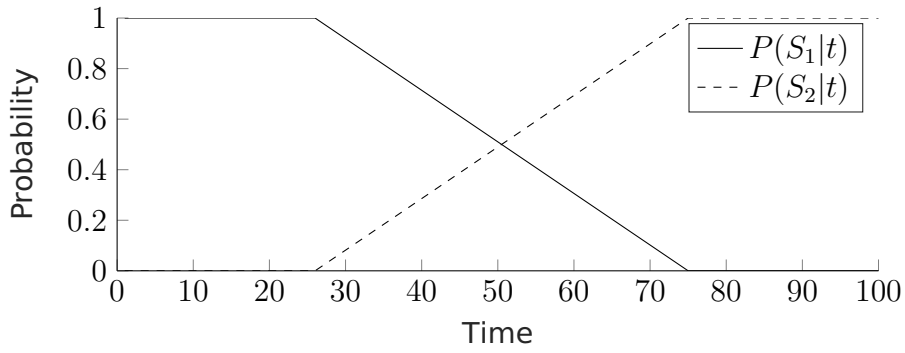



**Figure 2.3:** [Probabilities of example sources over time for a gradual change. [127, 134]

For example, a person sees a product they have never bought before on special offer, and buys it. Having developed a preference, they then buy this product with increasing frequency on subsequent shopping trips until it is a staple. This type is what is most commonly referred to [55, 186, 53] as *gradual* change, and is illustrated in Figure 2.2b. This form of gradual change is initially difficult to distinguish from an outlier or noise and as such challenges change detectors to balance conservatism with responsiveness. The second type is defined by a slow progression from $S_1$ to $S_2$ via a number of mixed intermediate concepts, as illustrated in Figure 2.2c. The latter type is also known as *incremental* change. An example could be a slowly degrading sensor. Gradual and incremental changes are closely related, because a gradual change could be interpreted as an incremental change in the source probabilities, as in Figure 2.3. The final type of change in this list is a *reoccurring concept*, where a previously encountered concept reappears periodically but unpredictably, illustrated in Figure 2.2d.

A fifth category, which has been omitted from this list, is outliers. This is because change detection algorithms should ideally be robust to noise and

outliers for the task at hand. This is by no means an exhaustive list - as Žliobaité [186] notes, there are $2^t$ ways two data sources can be combined over a sequence of length $t$.

There have been a number of attempts in the literature to characterise aspects of change in quantifiable ways, or place changes into mutually exclusive categories [87, 108, 127]. This is an attractive idea to ease the process of change detector design, especially when there are strong assumptions about the type of change. However, it is argued by Žliobaité [186] that categories cannot be mutually exclusive.

All these changes over time can be described within a general framework for non-stationary environments [134, 20]. A data source $S_i$, $i = 1,...,K$ is described by a set of class conditional probability density functions $p_i(\vec{x}|y)$ and prior probabilities $p_i(y)$. If at any given time $t$ there are one or more sources providing data, then a mixing function $\upsilon_i(t) \in [0,1]$, where $\sum_{i=1}^{K} \upsilon_i(t) = 1$ for any $t$, quantifies the influence of each source in a mixture distribution. Therefore the distribution for a time $t$ has prior probabilities

$$p(y,t) = \sum_{i=1}^{K} \upsilon_i(t) p_i(y) \tag{2.1}$$

and class conditional probability density functions

$$p(\vec{x}|y,t) = \sum_{i=1}^{K} \upsilon_i(t) p_i(\vec{x}|y) \tag{2.2}$$

This allows changes to be expressed in terms of mixing proportions of $\upsilon_i(t)$.

### 2.2.2 Streaming Data and Class Labels

This thesis focusses on unsupervised change detection, but it is useful to qualify what we mean by supervised, semi-supervised and unsupervised in the context of streaming data. In Equations 2.1 and 2.2 above, we consider changes probabilistically over time. In this model, changes arise from differing class distributions between data sources. Assuming that the data sources are latent and depending on the latency of class labels, we can think about the streaming data in the following ways.

Starting with the supervised scenario, let each observation $\vec{x}_t$ of $\mathbf{X}$ at time $t$ be a member of some class, $y_i$, $i = 1,...,K$. Let $\mathbf{Y}$ denote the sequence of class labels for $\mathbf{X}$. Here, each example in the stream is paired with its true class label, so $\mathbf{X}$ and $\mathbf{Y}$ have a one-to-one correspondence. The class labels may be available after each example, or in batch after some delay. We denote this as being a sequence of pairs of the form $z_t = (\vec{x}, y)$, $\mathbf{Z} = \{z_1, z_2, ...\}$. In the unsupervised scenario, we deal only with a sequence of observations, $\mathbf{X}$.

The availability of labelled data is often scarce [24, 124], so it may transpire that there are available class labels for some observations but not all – i.e. $|\mathbf{X}| > |\mathbf{Y}| > 0$. This is termed a semi-supervised scenario.

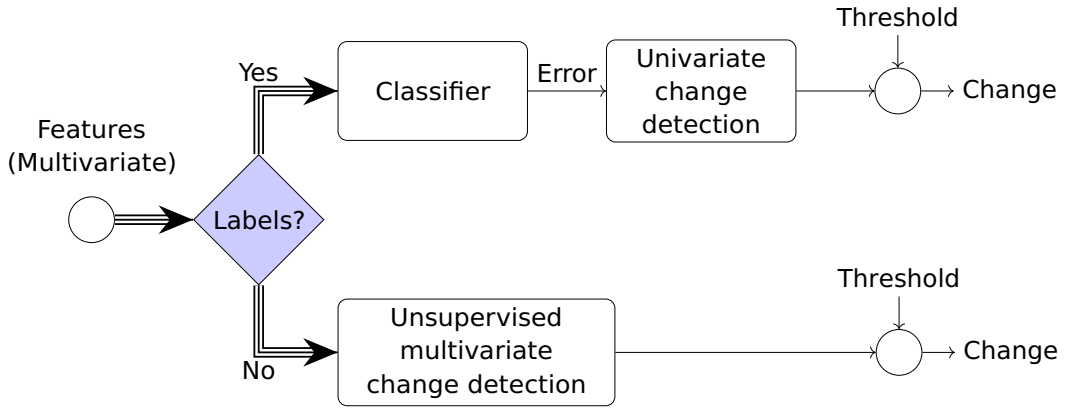



**Figure 2.4:** If labels are immediately or eventually available then we have the option of performing change detection on the univariate error stream of the classifier.

Figure 2.4 illustrates that if class labels are not available, we must inspect the data directly. If they are available then we have the option of performing univariate change detection on the error stream of a classifier, although this will only reveal changes which affect the classification accuracy. The former

scenario is the focus of this thesis, but in the supervised or semi-supervised scenario both options are available to us.

### 2.2.3 Context-Dependent Definition of Change

The definition of change varies depending on the context. We will first discuss the supervised adaptive learning problem as an example context.

Consider the problem of streaming classification in the presence of concept drift – adaptive learning. A classifier assigns labels to incoming examples, and the ground truth is received at some later point. If the optimum classification boundary changes over time, the classifier will perform poorly. The problem context is to minimise the error rate of the classifier. But not all changes in the distribution of incoming data will result in an increased classifier error rate [104]. Equally, a change in the error rate might not be accompanied by a change in the distribution of the data. It is for this reason that the vast majority of change detection for adaptive learning monitors the classifier error rate directly. The supervised learning scenario is a good example of the contextual nature of change, because it can be expressed formally.

Any classification problem can be described by Bayesian Decision Theory [134, 55, 39]. For a sequence of examples of the form $z_t = (\vec{x}, y)$, let $p(y)$ be the prior probabilities of the classes and $p(\vec{x}|y)$ be the class conditional probability density functions. Then using the Bayes rule (Equation 2.3), the classification decision is made on the posterior probabilities of the classes, $p(y|\vec{x})$.

$$p(y|\vec{x}) = \frac{p(y)p(\vec{x}|y)}{p(\vec{x})} \tag{2.3}$$

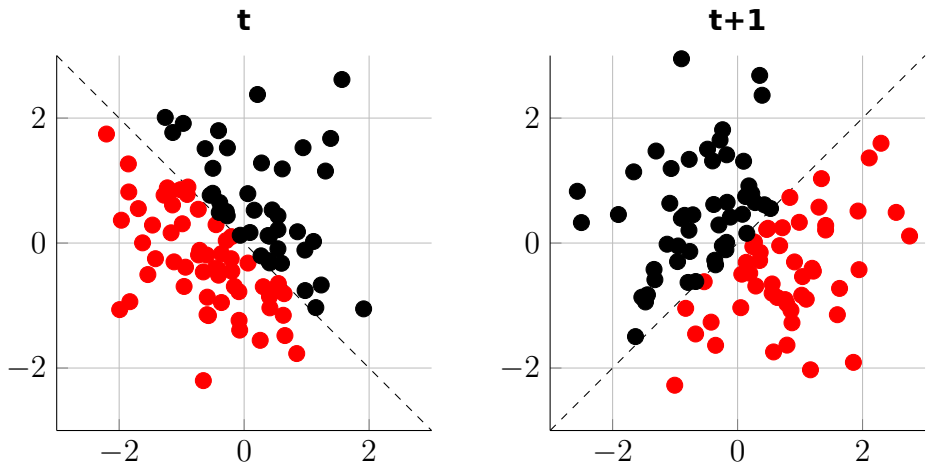



**Figure 2.5:** A change in the class conditional probabilities $p(\vec{x}|y)$, which will affect the classifier error rate, but exhibit no change in the distribution of the raw data, $p(\vec{x})$.

$$p(\vec{x}) = \sum_{y=1}^{K} p(y)p(\vec{x}|y) \tag{2.4}$$

**Table 2.1:** Implied meaning of changes in the component probabilities.

| Change in | Implies |
|---|---|
| $p(\vec{x})$ | The distribution of the incoming data has changed. |
| $p(\vec{x}\|y)$ | True class boundaries have changed. |
| $p(y\|\vec{x})$ | The optimal classification decision has changed. |

Table 2.1 lists the implications of change in these component probabilities [82, 186]. Changes in the prior and class conditional probabilities *might* adversely affect our classifier performance [56], only if the posterior probabilities change as a result. Figures 2.5 and 2.6 illustrate the effect of changes in $p(\vec{x}|y)$ and $p(\vec{x})$ respectively. The change in $p(\vec{x}|y)$, despite being impossible to detect from the unlabelled data, is the contextually important change as it affects the posterior probabilities $p(y|\vec{x})$. However, posterior probabilities may also change without affecting the error rate, as in Figure 2.6. Crucially within the adaptive learning scenario, only a supervised change detector which is monitoring the classifier error rate will be able to detect changes in $p(\vec{x}|y)$ as in Figure 2.5. An unsupervised change detector, despite the advantage of not needing to wait for ground truth, will only be able to detect changes in $p(\vec{x})$.

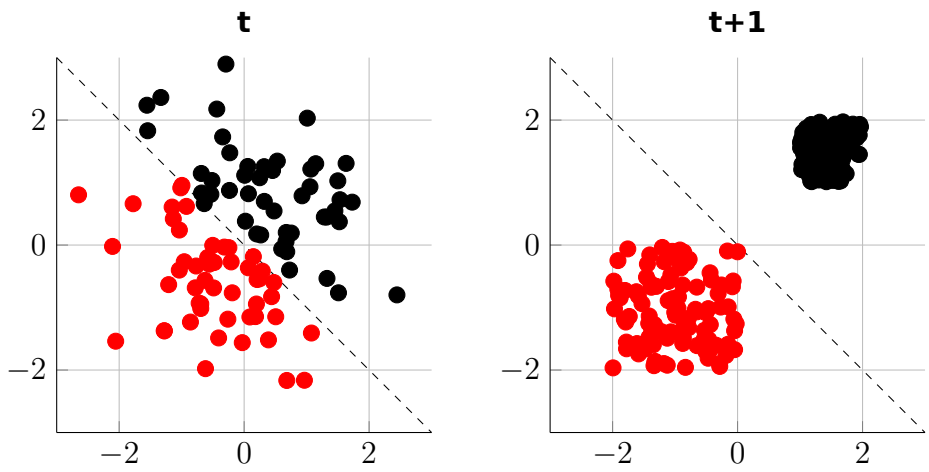



**Figure 2.6:** A change in the distribution of the raw data $p(\vec{x})$ which will not affect the classifier error rate.

Within the adaptive learning literature, these differing types of drifts are distinguished as follows.

**Real Concept Drift** [175, 168, 34, 41, 55] *Concept Shift* [147], *Conditional Change* [56]. Any change in $p(\vec{x})$ or $p(\vec{x}|y)$ leading to change in the posterior probabilities, $p(y|\vec{x})$.

**Virtual Concept Drift** [175, 168, 34, 41, 55] *Sampling Shift* [147], *Feature Change* [56], *Temporary Drift* [108]. The distribution of the data $p(\vec{x})$ changes without affecting the posterior probabilities $p(y|\vec{x})$.

However, as Gama et al. [55] note, the term *Virtual Drift* in particular has had numerous interpretations in the literature, meaning an incomplete data representation in Widmer and Kubat [175], a change in *both* $p(\vec{x})$ and $p(y|\vec{x})$ in Tsymbal [168], and a change in $p(\vec{x})$ but *not* $p(y|\vec{x})$ in Delany et al. [34].

A tangential problem to concept drift is that of *concept evolution* [123, 129, 128], which is the emergence and disappearance of classes in the data stream. Consider a semi-supervised scenario, where there is an abundance of data for modelling $p(\vec{x})$ and a scarcity of data for modelling $p(\vec{x}|y)$. Starting with a minimal assumption of $p(\vec{x}|y)$, we would be very interested in changes in $p(\vec{x})$ such as those in Figure 2.6, to detect evolution of the classes over time.

In summary, if class labels are not available then unsupervised change detection may only detect changes in $p(\vec{x})$. In supervised and semi-supervised scenarios, a hybrid approach may be used to detect changes in both $p(\vec{x})$ and $p(\vec{x}|y)$, but it depends on the context of the problem.

## 2.3 An Overview of Change Detection

### 2.3.1 Applications

In the last half century, there has been a proliferation of interest in applied change detection techniques. With a diverse range of applications, there is significant motivation for improving change detection performance. A selection of these applications are summarised below.

**Quality control** Quality control is the discipline that inspired the development of change detection techniques, with the aim of ensuring consistent quality of manufacturing by identifying defective items or batches [150, 137, 143, 130, 105].

**Classification** Classifiers are often deployed in environments where the nature of the classification problem is subject to change over time. Change detection is used to monitor the classifier error rate or the input space to detect change that would adversely affect the classifier [54, 176, 84, 14, 18]. It is then possible to trigger adaptation mechanisms to resolve the problem. For example, spam filtering systems rely on the classification of emails, the content of which is subject to constant change. [34].

**Data mining** When trying to identify significant events in large volumes of data, change detection or outlier detection is often employed. Some

example areas include fraud detection [66, 44], climate change detection[10, 141] and financial time series analysis [107, 136].

**Monitoring systems** Change detection is used to monitor data from real-time systems to identify problems. For example, network intrusion detection [156, 164, 22], fault detection [132], ECG segmentation [126] and bio-signal monitoring [68, 157].

### 2.3.2 Related Fields

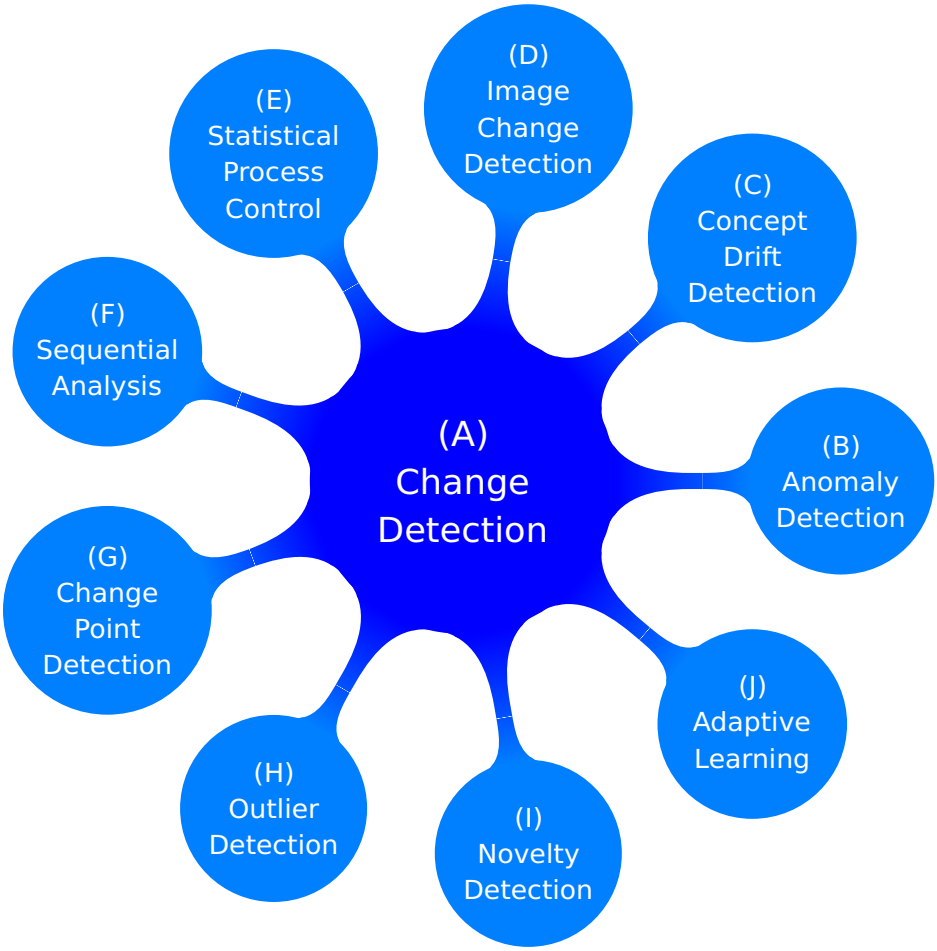



**Figure 2.7:** Closely related fields to change detection.

Figure 2.7 shows the adjacent fields to change detection. Approaches for change detection are often transferable to similar problems, and vice versa. For example, the subsequently referenced publications frequently review the same, or very similar approaches. Basseville and Nikiforov [9] is a substantial

review of abrupt change detection techniques from a data mining perspective. Gama [53] offers an overview also in data stream mining. Ghosh and Sen [58] is a handbook for sequential analysis. Montgomery [130] is a book of techniques for statistical process control. In novelty and anomaly detection, there are method reviews by Markou and Singh [122] and Pimental et al. [139], in outlier detection, by Ben-Gal [12], in climate change detection, by Reeve et al. [141], and in change point detection, by Aminikhanghahi and Cook [6]. Terminological differences can be significant, usually in referring to differing assumptions as well as applications. A brief summary follows of each field from Figure 2.7 and their problem interpretations.

**(A)** Change Detection is attempting to detect when the concept that underlies the data has changed.

**(B)** Anomaly Detection is the detection of patterns in data that do not conform to a well defined notion of normal behaviour [28].

**(C)** Concept Drift Detection refers to change detection usually in the context of an evolving classification problem. 'Drift' is used to refer to gradual change, for example, the target concept changing over time and invalidating a classifier [53].

**(D)** Image Change Detection is detecting changes or motion in images or videos, usually through computer vision techniques or frame-to-frame difference images. It is often referred to simply as change detection.

**(E)** Statistical Process Control refers to a particular family of methods that model a data stream as a stochastic process, and assess whether that process is 'in control' or 'out of control'.

**(F)** Sequential Analysis is a term which implies a statistical approach where the sample size is not fixed. It is typically associated with methods that

use cumulative statistics, such as a ratio of sequential probabilities over a sequence of data [171, 52, 55].

**(G)** Change Point Detection implies that the objective is not only to detect a concept change, but also estimate its time or sequential position.

**(H)** Outlier Detection refers to the detection of individual points that deviate significantly from the population [28, 63].

**(I)** Novelty Detection is the detection of previously unseen, or 'novel' patterns in data [28].

**(J)** Adaptive Learning is the process of adapting a classifier to a changing classification problem, typically relying on change detection in some capacity.

Of the fields just discussed, adaptive learning and typically concept drift refer to the supervised problem. It is clear from the above summaries that methods developed to solve a particular problem are likely to be applicable to others. In many cases, the difference in problem statements is largely semantic. For example, by detecting concept change (Change Detection) we have implicitly arrived at an estimation of a change point (Change Point Detection). Detection of previously unseen patterns in data (Novelty Detection) may strongly imply concept change (Change Detection).

Subjectively speaking, outlier detection, novelty detection and change detection occupy points along a continuum as depicted in Figure 2.8. If outlying data is persistent over time, we might at various points along the continuum call this concept a novelty or anomaly, and eventually a change. Techniques discussed within this thesis are applicable along this continuum assuming that they offer a parameter related to the persistence of change.

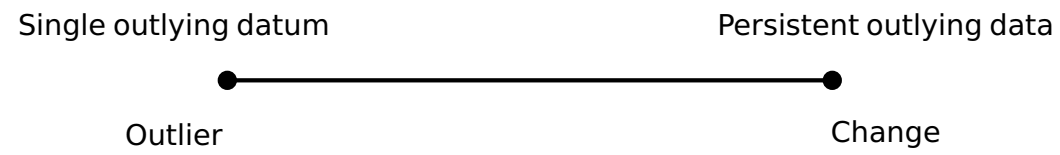



**Figure 2.8:** A selection of the fields lie subjectively along a continuum of outlier persistence.

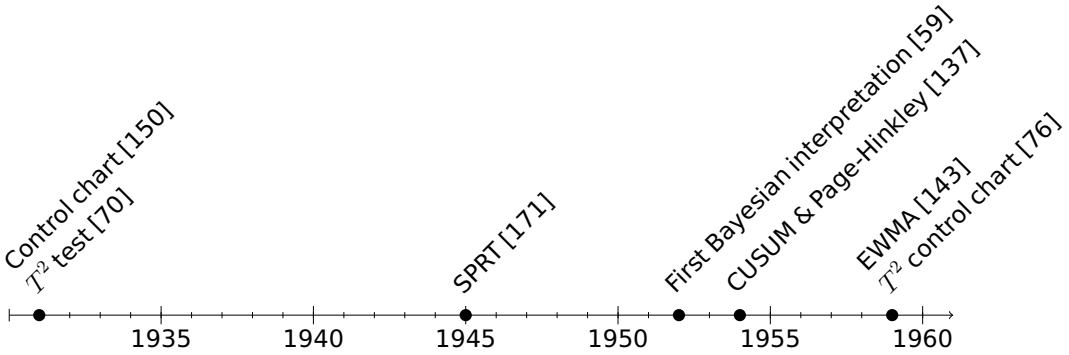



**Figure 2.9:** Early advances in Change Detection 1930–1960.

### 2.3.3 A Chronology of Advances

Applied change detection has been an active area of research since at least the early 20th century, pioneered by Shewhart's 1931 work on the 3-sigma control chart [150] as a method for quality control in manufacturing. The Central Limit Theorem implies that the average of a sequence of independent observations from any distribution will itself be normally distributed. The three-sigma chart deems a process to be 'Out of control' if any observation lies greater than three standard deviations from the sample mean. In 1959, Roberts [143] formulated a control chart based on geometric moving averages[1], to incrementally downweight old data. Also in 1959, a multivariate $T^2$ control chart was formulated by Jackson [76]. Today there are many variations on the control chart, which can describe any change detection scheme that defines limits on a statistic. A number of multivariate control charts became prominent in the 80s and 90s. MacGregor and Kourti [118] offer a review of multivariate control charts. They discuss multivariate $T^2$ and $\chi^2$ Shewhart charts, multivariate cumulative sum (CUSUM) charts, and a multivariate EWMA chart.

[1] Frequently called an exponentially-weighted moving average (EWMA) chart

In 1943, Abraham Wald devised the Sequential Probability Ratio Test (SPRT) – a statistical hypothesis test where the sample size is not fixed in advance. The impact of this technique on quality control for military manufacturing was not lost to the Allies, and it was classified under the Espionage Act [172], eventually published after the war [171]. This test would become the basis for many future approaches, and is the foundation of the field of Sequential Analysis [77]. The SPRT framework underpins a significant proportion of well known methods such as the CUSUM procedure and the Page-Hinkley test, which were first published in 1954 by Page [137]. In 1961, Hotelling's $T^2$ test for multivariate data [70] was adapted into sequential form by Jackson and Bradley [77].

As a statistical problem, change detection has had both Bayesian and Frequentist interpretations [9], with Sequential Analysis being the latter. The first Bayesian interpretation was by Girschick and Rubin [59] in 1952, followed by work on optimality by Shiryaev [151] in the 1960s. In 2007, Adams and MacKay [1] published a complete Bayesian framework for streaming change detection.

From the late 1980s onwards, there was a proliferation of interest in change detection techniques to adapt classifiers learning from noisy or concept-drifting data streams (adaptive learning) and also for making sense of very high volume data streams (data mining). In the early 90s, two large reviews of change detection techniques were published; Basseville and Nikiforov [9] and Ghosh and Sen [58]. Research from these adjacent communities yielded well known adaptive learning systems such as STAGGER by Schlimmer and Granger [149], FLORA by Widmer and Kubat [176] and SEA by Street and Kim [160]. Inspired by efficient stream classification approaches like VFDT [74], data management techniques like ADWIN [17] were developed, which can act as change detectors. Most of these systems monitor performance indicators in the data [53] or the log likelihood ratio of an individual observation in the SPRT-derived systems.

As computing power becomes more readily available, methods compare the distributions of two time windows of the data instead [83, 17, 102].

Most recently, there is increasing interest in the use of ensemble and consensus clustering approaches [89, 60, 119, 177, 38, 64, 166] as well as neural networks [178, 133].

## 2.4 A Review of Taxonomies

There are many avenues from which to approach a change detection problem and this makes a taxonomy very challenging. Taxonomic surveys of change detection approaches often focus on a single application area [55, 186], or sub-domain [139, 63, 181]. Wide ranging surveys are rare due to the non-mutually exclusive nature of many potential categories, and the scale of review required. Here we will review a selection of the most transferable taxonomies in the literature on change detection, anomaly, novelty and outlier detection, adaptive learning and change point detection. Table 2.2 lists the top level categories of the most transferable taxonomies from the subsequently discussed publications.

Gama et al. [55] is a survey of approaches for concept drift adaptation. The survey focuses specifically on adaptive learning rather than change detection, but some of the taxonomies are transferable. They create modular taxonomies which describe a particular part of any given adaptive learning approach. Their taxonomies are; data management, forgetting mechanism, change detection, learning and loss estimation. The first three are transferable to our general case here. The learning and loss estimation taxonomies are less transferable, because they are properties closely tied to supervised adaptive learning. The data management and forgetting mechanism categories refer to how the methods manage updating of the model with new information, and the phasing out

of old information. Change detection methods are placed into four categories based on common algorithmic roots. Sequential Analysis refers to detectors for which decisions are based on a ratio of sequential probabilities or use cumulative statistic, like Wald's Sequential Probability Ratio Test (SPRT) [171]. Control charts refers to detectors which model the input as a stochastic process, and decide whether this process is "In control" or "Out of control" based on a set of rules, as Shewhart's original control chart [150] did. Detectors that monitor the distributions of two time-windows also occupy a category, typically keeping a window of reference data and comparing its distribution to a constantly updated window of new data. The final category is contextual approaches, which partition the input space and attempt to identify localised concepts in the data. In a semi-supervised construction, a classifier is then trained to recognise these localised concepts in subsequent data to monitor the evolution of the stream.

Basseville and Nikiforov [9] is a reference text on the detection of abrupt changes. Whilst not offering an explicit taxonomy, we can take the categories from the sections of Chapter 2, "Change Detection Algorithms", where they present similar methods grouped together. These include control charts, filtered derivative methods, sequential probability ratio tests and bayes-type algorithms.

Pimentel et al. [139] review the related field of novelty detection. They place methods into 5 global categories as follows. Probabilistic methods are those which estimate the probability density function (pdf) of the data and subsequently test the likelihood of future observations against it. Distance based methods include clustering and nearest-neighbour approaches, which rely on distance metrics between data points and cluster centroids. Reconstruction based methods train an estimator on incoming data, and attempt to predict the next example in the stream. The distance between this prediction and the actual observed value is used to calculate a novelty score. Domain-based methods use training data to define a class boundary (domain). This

**Table 2.2:** Transferable taxonomies of similar methods, and their top level categories

| Taxonomy | Top Level Categories |
|---|---|
| Change Detection for Concept Drift, Gama et al. [55] | • Sequential Analysis<br>• Control Charts<br>• Monitoring two distributions<br>• Contextual |
| Abrupt Change Detection, Basseville and Nikiforov [9] | • Sequential Probability Ratio Tests<br>• Control Charts<br>• Filtered Derivatives<br>• Bayesian |
| Novelty Detection, Pimentel et al. [139] | • Probabalistic<br>• Distance-based<br>• Domain-based<br>• Reconstruction-based<br>• Information-theoretic |
| Supervised Change Point Detection, Aminikhanghahi and Cook [6] | • Multi-class classifiers<br>• Binary classifiers<br>• Virtual classifiers |
| Unsupervised Change Point Detection, Aminikhanghahi and Cook [6] | • Likelihood Ratio<br>• Subspace model<br>• Probablistic methods<br>• Kernel based methods<br>• Graph based methods<br>• Clustering |
| Anomaly Detection, Chandola et al. [28] | • Classification Based<br>• Clustering Based<br>• Nearest Neighbour Based<br>• Statistical<br>• Information Theoretic<br>• Spectral |

then decides if an observation will be considered a novelty, irrespective of the observed class density. Information theoretic approaches base decisions upon the information content of a dataset by analysing measures such as the entropy, although this is implied to be a typically offline approach, utilising the entire dataset. Pimental et al. further divide probabilistic methods into parametric and non-parametric categories depending on whether they estimate parameters for a distribution (e.g. Gaussian Mixture Model) or build a model from the density of the input space (e.g. Kernel Density Estimation).

Aminikhanghahi and Cook [6] survey and categorise methods for change point detection in time series. They discern firstly between *Supervised* and *Unsupervised* methods, producing separate taxonomies for each. By Multi class classifiers, they refer to a supervised learning scenario where the classes are known *a priori*, a classifier is trained from a training set containing all such classes and tested on a sliding window over the data. Binary or one-class classification comprises a supervised learning scenario where the concept transitions in the data stream represent one class, with the other class comprising all other data.

Chandola et al. [28] is a wide-ranging survey of anomaly detection deliberately attempting to bridge several research areas and application domains. Their top level categories are *Classification Based*, *Clustering Based*, *Nearest Neighbour Based* and *Statistical*. However, from their meta analysis of reviews they identify two other common categories, *Information Theoretic* and *Spectral*.

Žliobaité [186] presents an overview of learning under concept drift, with a taxonomy of concept drift learners. The taxonomy is a wider focus on adaptive learning techniques, abstracted too far from change detection techniques to be directly transferable here. Gupta et al. [63] is a survey of outlier detection in temporal data, which takes a data-first perspective. Whilst under the "Data Streams" category the taxonomy contains common subcategories

**Table 2.3:** Comparison matrix showing categories for change detection techniques appearing in at least two of the reviewed surveys. The field each survey is drawn from is denoted by the following acronyms. Concept Drift: **CDr**, Novelty Detection: **ND**, Change Point Detection: **CPD**, Anomaly Detection: **AD**, Change Detection: **CD**.

| **Category** | Basseville & Nikiforov [9] | Gama et al. [55] | Aminikhanghahi & Cook [6] | Pimentel et al. [139] | Chandola et al. [28] |
|---|---|---|---|---|---|
| | **CD** | **CDr** | **CPD** | **ND** | **AD** |
| Supervised | | | ○ | ○ | ○ |
| Unsupervised | | | ○ | | ○ |
| Likelihood Ratio | ○ | ○ | ○ | | |
| Control Chart | ○ | ○ | | | |
| Sequential Analysis | ○ | ○ | | | |
| Probabalistic | | ○ | ○ | | |
| Subspace | | ○ | ○ | | |
| Clustering | | | ○ | | ○ |
| Information Theoretic | | | | ○ | ○ |
| Distance-Based | | | | ○ | ○ |

with Pimental et al. [139] (Evolving Prediction Models $\simeq$ Reconstruction-based, Distance Based outliers $\simeq$ Distance-based), the data-first approach is too restrictive when categorising change detection techniques for the purposes of this thesis.

The comparison matrix in Table 2.3 provides some interesting insights into the proximity of fields. We see considerable proximity between change detection and concept drift detection, and between anomaly and novelty detection. The concept drift survey naturally omits the "supervised" category because adaptive learning is a supervised problem. The novelty and anomaly detection surveys were more likely to categorise supervision and discuss empirical spatial differences in the data rather than probabilistic modelling of a process.

## 2.5 Modular Taxonomies for Change Detectors

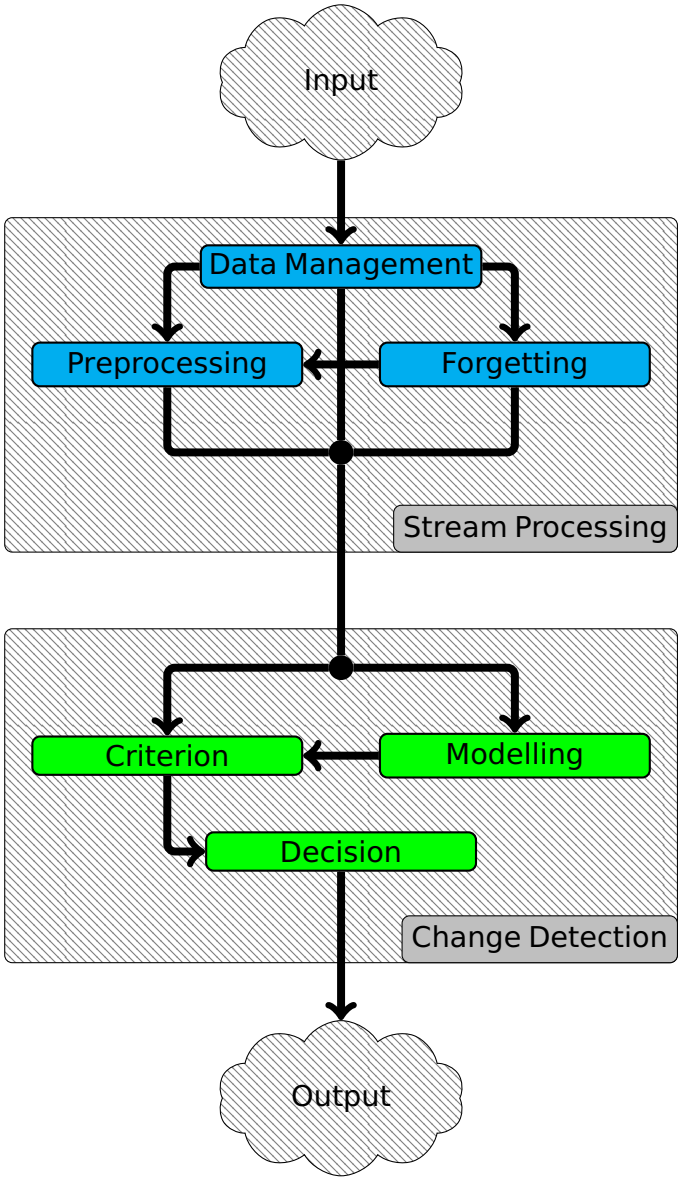



**Figure 2.10:** Flowchart demonstrating the pipeline relationship between modules in a change detector.

The categories discussed in Section 2.4 are not mutually exclusive. Observing that the categories generally refer to a particular characteristic of a method, Figure 2.10 proposes a general pipeline of modules for building change detectors, split into two primary groups – Stream Processing and Change Detection. The former group refers to the memory management of streaming data (e.g. windowing) and the extraction of features ready for change detection. The latter refers to the actual change detection process using those features. The modules are defined as follows.

**Data Management** takes examples from the data stream and retains an appropriate amount in memory for the task at hand.

**Forgetting** removes examples from the memory when they are deemed no longer relevant.

**Preprocessing** extracts features of interest for the task at hand.

**Modelling** builds an estimation of the data source or the properties of the data source to reason about.

**Criterion** reduces the problem to state about which we can make a decision.

**Decision** makes a binary decision about the criterion it is provided: change or no change.

Note that the construction of the flowchart implies that a number of steps are optional. For example, a simple three-sigma control chart might take one example at a time (data management), calculate the new rolling mean and variance (criterion), and check whether the mean is within three standard deviations (decision). This particular detector takes a path which omits preprocessing, forgetting and modelling.

Figure 2.11 lists taxonomies for the modules which were just defined. The Forgetting Mechanism and Data Management taxonomies from Gama et al. [55], are already modular, because they describe particular facets of stream processing and are therefore applicable beyond adaptive learning. The Criterion categories are informed from the categories reviewed in Tables 2.2 and 2.3. A taxonomy for the Decision module was omitted because it is typically a threshold tied to the criterion assumptions.

In addition to the modules, Figure 2.12 lists two global taxonomies that any change detection approach can be placed within. Supervision depends on

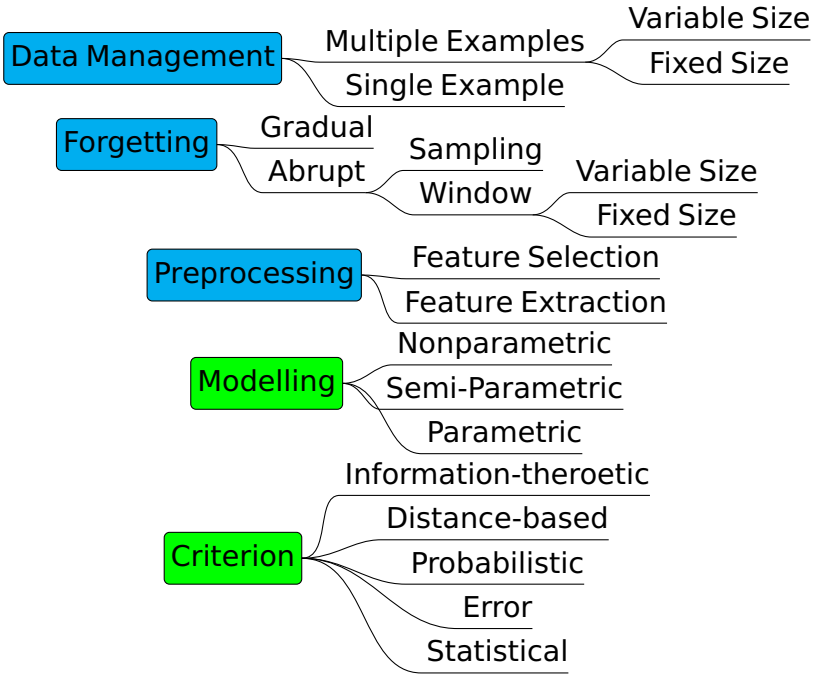



**Figure 2.11:** Modular taxonomies for change detection. Blue relates to Stream Processing and green to Change Detection.

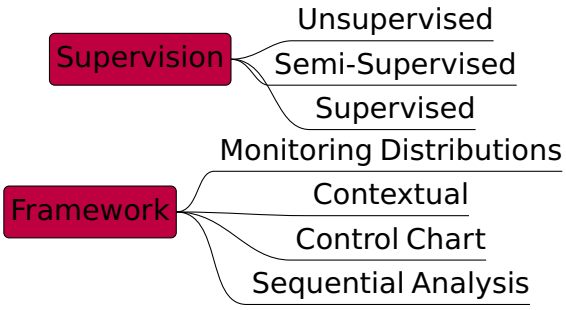



**Figure 2.12:** Global taxonomies for change detection.

whether the method incorporates labelled training data in its workflow. "Framework" is the taxonomy by Gama et al. [55] which discriminates by common algorithmic roots.

## 2.6 Methods for Change Detection

In the previous sections, we have seen how change detection methods are categorised in the literature. This section will provide detail on specific building blocks and change detection methods that are used throughout the experimental portions of this thesis.

### 2.6.1 Building Blocks

#### Distances and Divergences

We commonly wish to quantify the difference between distributions, or express the likelihood that a particular example belongs to a distribution. The following measures can fulfil this purpose.

The Mahalanobis distance is a multivariate distance between a $p$ dimensional point $\vec{x}$ and a distribution $P$ where $\vec{\mu}$ and $\mathbf{\Sigma}$ respectively are the mean and covariance of $P$.

$$D_M(\vec{x},P) = \sqrt{(\vec{x}-\vec{\mu})^T \mathbf{\Sigma}^{-1} (\vec{x}-\vec{\mu})} \tag{2.5}$$

Intuitively, this is related to the Euclidean distance, except taking into account the covariance of the distribution. The distance reduces to the Euclidean distance if $\mathbf{\Sigma}$ is the identity matrix. For normal distributions, the squared Mahalanobis distance is chi-squared distributed, with $p$ degrees of freedom.

The Bhattacharyya distance approximates the overlap between two distributions. Let $P(x)$ and $Q(x)$ be probability distributions of the random variable $x$. Assuming, without loss of generality, that $P$ and $Q$ are continuous, the Bhattacharyya distance between the two distributions is

$$D_B(P,Q) = -\ln \int \sqrt{P(x)Q(x)} dx. \tag{2.6}$$

In this thesis we use the formulation of the distance between two multivariate Gaussians

$$
\begin{gathered}
D_B(P,Q) = \frac{1}{8}(\vec{\mu}_P - \vec{\mu}_Q)^T \mathbf{\Sigma}^{-1}(\vec{\mu}_P - \vec{\mu}_Q) + \frac{1}{2}\ln\left(\frac{\det\mathbf{\Sigma}}{\sqrt{\det\mathbf{\Sigma}_P \det\mathbf{\Sigma}_Q}}\right), \\
P \sim \mathcal{N}(\vec{\mu}_P, \mathbf{\Sigma}_P), \quad Q \sim \mathcal{N}(\vec{\mu}_Q, \mathbf{\Sigma}_Q), \\
\mathbf{\Sigma} = \frac{\mathbf{\Sigma}_P + \mathbf{\Sigma}_Q}{2}
\end{gathered}
\tag{2.7}
$$

where $\vec{\mu}_p$ and $\mathbf{\Sigma}_p$ is the mean and covariance of distribution $P$, and $\mathbf{\Sigma}$ is the pooled covariance matrix of $P$ and $Q$.

The Kullback-Leibler divergence [95], also called *relative entropy* is a measure of the difference between two probability distributions. For two discrete distributions $P$ and $Q$ with $K$ bins, the statistic is defined as:

$$
D_{KL}(P||Q) = \sum_{i=1}^{K} P(i) \log \frac{Q(i)}{P(i)} \tag{2.8}
$$

If the two distributions are identical, then the value of $D_{KL}(P||Q)$ is zero. A larger value indicates a higher likelihood that $P$ and $Q$ are different.

### Tests of Equality

Consider the basic assumption in change detection that we expect to see samples drawn from different data sources. Taking two independent, time-adjacent samples from a data stream, we wish to test the null hypothesis that both samples came from the same data source. Here we discuss several such tests.

The Mann-Whitney U test [121], also known as the Wilcoxon Rank Sum test is nonparametric test for median equivalence of two continuous, univariate

samples. Rank based statistics are difficult to generalise to multiple dimensions because of the difficulty of ordering the data [33]. Despite this, Kifer et al. [83] showed that it was possible in principle and Lung-Yut-Fong et al. [117] proposed a generalisation called *MultiRank*.

Hotelling [70] proposes a statistical test for equivalence of the means of two multivariate samples, $W_1$ and $W_2$. The null hypothesis is that $W_1$ and $W_2$ are drawn independently from two multivariate normal distributions with the same mean and covariance matrices. Denote the sample means by $\hat{\vec{\mu}}_1$ and $\hat{\vec{\mu}}_2$, the pooled sample covariance matrix by $\hat{\mathbf{\Sigma}}$, the cardinalities of the two windows by $M_1 = |W_1|$ and $M_2 = |W_2|$ and the data dimensionality by $p$. The $T^2$ statistic is calculated as

$$T^2 = \frac{M_1 M_2 (M_1 + M_2 - p - 1)}{p(M_1 + M_2 - 2)(M_1 + M_2)} \times (\hat{\vec{\mu}}_1 - \hat{\vec{\mu}}_2)^T \hat{\mathbf{\Sigma}}^{-1} (\hat{\vec{\mu}}_1 - \hat{\vec{\mu}}_2) \tag{2.9}$$

Under the null hypothesis, $T^2$ has $F$ distribution with degrees of freedom $p$ and $M_1 + M_2 - p + 1$. The $T^2$ statistic is the Mahalanobis distance between the two sample means multiplied by a constant. The $p$-value of the statistical test is instantly available and the desired significance level will determine the change threshold.

The obvious problem with the Hotelling test is that it is only meant to detect changes in the position of the means. Thus it will not be able to indicate change of variance or a linear transformation of the data that does not affect the mean.

### Maintaining Statistics

Change detection algorithms depend on the maintenance of basic statistics over streaming data. To classically calculate the mean and variance at time $t$ would require $O(t)$ space complexity. Therefore simple cumulative estimations

are very useful [53]. To arrive at cumulative estimations of the mean, variance and standard deviation we need to store three cumulative statistics.

- The number of examples, $t$.
- The sum of the data points from $0...t$, $\sum x_i$.
- The sum of the squares of the data points from $0...t$, $\sum x_i^2$.

Then the estimated mean, variance and standard deviation at time $t$ is given by

$$\mu_t = \frac{\sum x_i}{t} \tag{2.10}$$

$$\sigma_t^2 = \frac{\sum x_i^2}{t} - \mu_t^2 \tag{2.11}$$

$$\sigma_t = \sqrt{\sigma_t^2} \tag{2.12}$$

These measures easily extend into the general multivariate case – to estimate these statistics for a data stream with $p$ features will only require $O(3p)$ space.

### Sliding Windows

Consider that we are monitoring a stream that we *expect* to change. If the stream has changed at time $t$, then all $x_i$ where $i < t$ are now likely to provide a poor representation of the new concept in the stream. This is why commonly, algorithms are concerned with looking at the recent past, rather than the whole past [53]. This is achieved through maintaining windows of recent examples, in effect sliding them over the data, hence the term *sliding window*. A typical implementation might involve any *first in first out* (FIFO) data structure, such as a queue.

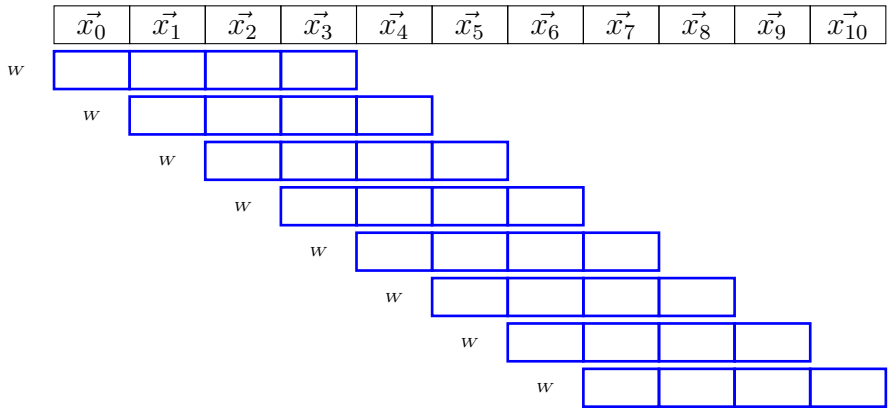



**Figure 2.13:** Fixed size single sliding window.

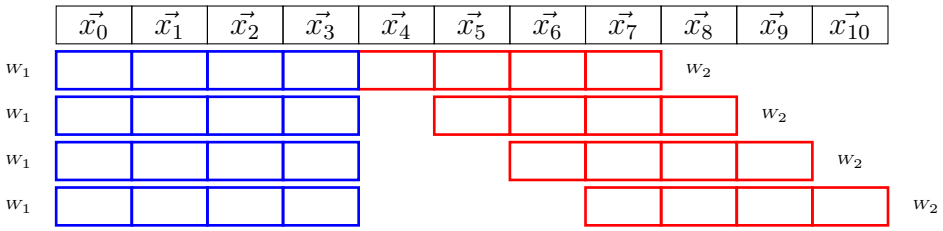



**Figure 2.14:** Pair of fixed size sliding windows with fixed reference window.

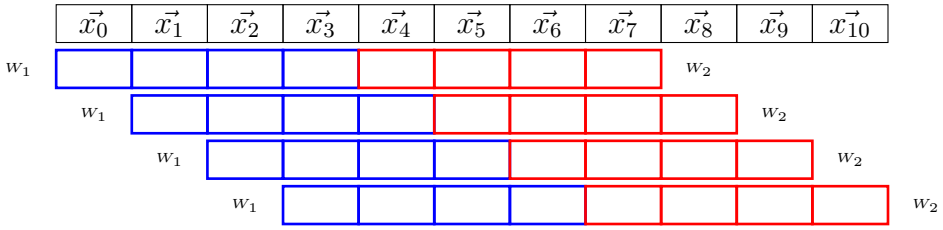



**Figure 2.15:** Pair of fixed size adjacent sliding windows.

In the simplest case, we choose a fixed window size as a parameter of our change detection algorithm. As we encounter each new example, we append it to the window $W \cup \vec{x}_t$. Simultaneously, we drop the oldest element from the window $W \setminus \vec{x}_{t-|W|}$. This process is illustrated in Figure 2.13.

A single window is useful for methods which compute statistics on recent data. An alternative construction is where the data in two windows is compared to establish whether they were generated by the same process. The two common dual-window schemes are illustrated in Figures 2.14 and 2.15. In the first, we fix the reference window to be the first examples we encounter, and slide a second window over the data. Typically if change were detected, the windows would be cleared and a new reference window taken from that

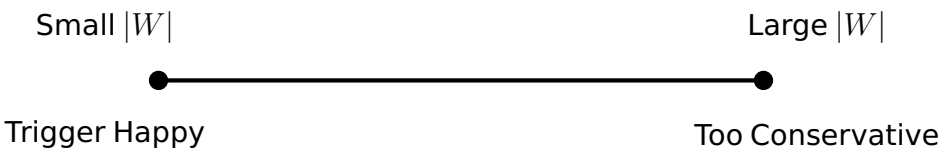



**Figure 2.16:** The continuum of window size choices.

point in the stream. In the second, we effectively take a single sliding window of $2|W|$ elements, and divide it into head and tail windows for comparison.

There is a continuum that must be straddled to achieve an ideal window size, depicted in Figure 2.16. A window that is too small will not include enough examples to be a stable description of the concept [176] – noise or natural variation in the stream may trigger a false alarm. On the other hand, a window that is too large may be very slow to detect concept drift due to the weight of previous examples. Fixing a window size introduces a parameter into change detection algorithms that may require optimization, tuning or insight to achieve an optimal outcome. Whilst this may be an acceptable cost, there has been considerable research into dynamic sliding windows which adjust to supposedly optimal lengths from the observed data.

The adaptive windowing scheme for FLORA by Widmer and Kubat [174, 176] was one of the first such algorithms. FLORA is a rule-based stream classifier that builds a concept description online. Incoming examples are placed into three sets of descriptors – accepted, potential and negative, depending on their compliance with the current hypothesised concept. The window size is adapted using a heuristic based on the current accuracy and the proportion of accepted descriptors in the current window. Klinkenberg and Renz [85] adapted window size based on the accuracy, precision and recall at the current point in the stream. Klinkenberg and Joachims [84] adjusted the window size to minimise the estimation of the leave-one-out error. Gama et al. [54] adjust to minimise the generalisation error. Bifet and Gavaldà [17] split a window using a threshold of subwindow differences based on the Hoeffding bound. Koychev and Lothian [88] split a window using a procedure based on the golden ratio.

### 2.6.2 Univariate Methods

#### Sequential Analysis and Control Charts

Recall the categories from the framework taxonomy of Gama et al. [55] in Figure 2.12. In the literature, the terms "Sequential Analysis" and "Control Charts" are not used mutually exclusively [55, 145]. The canonical Sequential Analysis method is the Sequential Probability Ratio Test (SPRT) [172]. Consider a sequence of examples $X = [x_1,...,x_N]$. The null hypothesis $H_0$ is that $X$ is generated from a given distribution $P(x)$, and the alternative hypothesis $H_1$ is that $X$ is generated from another (known) distribution $Q(x)$. A cumulative statistic $\Lambda_N$ is calculated as the logarithm of the likelihood ratio for the two distributions:

$$\Lambda_N = \sum_{i=1}^{N} \log \frac{P(x_i)}{Q(x_i)} \tag{2.13}$$

Two thresholds, $\alpha$ and $\beta$ are defined depending on the target error rates. If $\Lambda_N < \alpha$, $H_0$ is accepted, else if $\Lambda_N > \beta$, $H_1$ is accepted. In the case where $\alpha \leq \Lambda_N \leq \beta$, the decision is postponed, the next example in the stream, $x_{N+1}$, is added to the set, and $\Lambda_{N+1}$ is calculated and compared with the thresholds.

Control charts are a category of methods that are based upon Statistical Process Control (SPC), originating from the work of Shewhart [150]. In SPC, the modus operandi is to consider the problem as a known statistical process, and monitor its evolution. Figure 2.17 shows the canonical XBar control chart from the MATLAB statistics toolbox [125]. Observations of manufactured products $X = [x_1,...,x_N]$, (e.g. load bearing capacity of a joint) are taken over time and batched into $K$ fixed size subgroups $S = [S_1,...,S_K]$. A chart is then plotted. The centre line $\bar{x}$ of the chart is calculated as the mean of the subgroup averages. A thresholding scheme defines upper and lower limits on an in-control process,

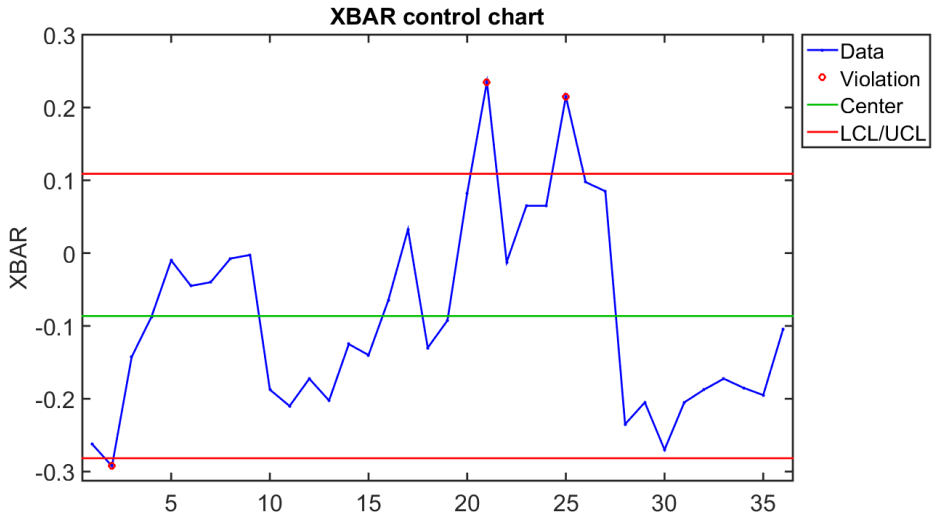



**Figure 2.17:** An example XBar chart created by the MATLAB statistics toolbox. [125]

calculated as $\bar{x} \pm 3\bar{\sigma}$. There are numerous types of control chart including X-Bar, Moving Range, and Individuals. A control chart simply comprises a plotted statistic and control rules or limits to decide whether the process under observation is in control or out of control. Under this definition, cumulative statistics from Sequential Analysis such as SPRT and CUSUM which define thresholds can be and often are described as control charts.

Cumulative sum (CUSUM) [137] is a sequential analysis technique based on the principle of accumulating how much a statistic varies from a desired value. The test is widely used for detecting significant change in the mean of input data. Starting with an upper cumulative sum statistic $g_0^{\Delta} = 0$, CUSUM updates $g^{\Delta}$ for each subsequent example as

$$g_t^{\Delta} = \max(0, g_{t-1}^{\Delta} + (x_t - \delta)) \tag{2.14}$$

where $\delta$ is the magnitude of acceptable change. Change is signalled when $g_t^{\Delta} > \lambda$, where $\lambda$ is a fixed threshold. If we wish to detect both positive and negative shifts in the mean, we can also compute and threshold the lower sum as

$$g_t^{\nabla} = \min(0, g_{t-1}^{\nabla} - (x_t - \delta)) \quad . \tag{2.15}$$

The Page-Hinkley test [137] is derived from CUSUM, and adapted to detect an abrupt change in the average of a Gaussian process [55, 53, 132]. First, the cumulative difference between the observed values and their mean at the current point in time is calculated.

$$m_t = \sum_{i=1}^{t} (x_i - \mu_t - \delta) \tag{2.16}$$

where the $\delta$ parameter again represents the magnitude of acceptable change. The minimum observed value of this statistic is retained as $M_t = min\{m_1,...,m_t\}$. Taking the statistic as

$$PH_t = m_t - M_t \tag{2.17}$$

change is signalled if $PH_t > \lambda$, where $\lambda$ is a chosen threshold.

There are a number of control chart based approaches employed for concept drift detection in the literature, typically assuming the supervised setting. As the canonical chart relies on direct observations of a fixed number of examples, stream classification problems typically use a construction as follows. Assume that we monitor classification error. This error can be interpreted as a Bernoulli random variable with probability of "success" (where error occurs) $p$. The probability is unknown at the start of the monitoring, and is re-estimated with every new example as the proportion of errors encountered thus far. At example $i$, we have a binomial random variable with estimated probability $p_i$ and standard deviation $\sigma_i = \sqrt{p_i(1-p_i)/i}$. One way to use this estimate is described below [54, 55]:

1. Denote the (binary) streaming examples as $x_1, x_2, ....$ To keep a running score of the minimum $p$, start with estimate $p_{\min} = 1$, and $\sigma_{\min} = 0$. Initialise the stream counter $t \leftarrow 1$.

2. Observe $x_i$. Calculate $p_i$ and $\sigma_i$. For an error and a standard deviation ($p_i$, $\sigma_i$) at example $x_i$, the method follows a set of rules to place itself into one

of three possible states: in control, warning, and out of control. Under the commonly used confidence levels of 95% and 99%, the rules are:

- If $p_t+\sigma_t<p_{min}+2\sigma_{min}$, then the process is deemed to be in control.
- If $p_t+\sigma_t \geq p_{min}+3\sigma_{min}$, then the process is deemed to be out of control.
- If $p_{min}+2\sigma_{min} \leq p_t+\sigma_t < p_{min}+3\sigma_{min}$, then this is considered to be the warning state.

3. If $p_t+\sigma_t < p_{\min}+\sigma_{\min}$, re-assign the minimum values: $p_{\min} \leftarrow p_t$ and $\sigma_{\min} \leftarrow \sigma_t$.

4. $t \leftarrow t+1$. Continue from 2.

Drift Detection Method (DDM) [54] is designed to monitor classification error using the above control chart construction for streaming classification. It assumes that the error rate will decrease while the underlying distribution is stationary. Similarly, the Early Drift Detection Method (EDDM) [8] is an extension of DDM which takes into account the time distance between errors as opposed to considering only the magnitude of the difference, which is aimed at improving the detector's performance on gradual change. HDDM$_A$ and HDDM$_W$ are extensions which remove assumptions relating the to probability density functions of the error of the learner. Instead, they assume that the input is an independent and bounded random variable, and use the confidence interval from the Hoeffding Bound [48]. The Hoeffding bound implies that the true mean of a real-valued random variable $r$ with range $R$ for which we have $n$ observations is at least $\bar{r}-\epsilon$ with probability $1-\delta$ [36, 67] where

$$\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}} \tag{2.18}$$

It is one of several useful concentration inequalities on random variables along with the Chebyshev, Chernoff and Bernstein bounds [53] which find common use in data stream mining.

The geometric moving average chart (GEOMMA), introduced by Roberts [143], assigns weights to each observation such that the weight of older observations decreases in geometric progression. This biases the method towards newer observations, improving the adaptability. This concept is applied in the EWMA charts used by Ross et al. [144], which are also designed to monitor classification error. They maintain a weighted estimate of the error rate, and signal change based on the expected mean and standard deviation of this estimate.

### Monitoring Two Distributions

Another common pattern found in change detectors is monitoring the distributions of two windows of data. This framework is rigourously defined by Kifer et al. [83]. The basic construction involves a reference window composed of old data, and a detection window composed of new data. This can be achieved with a static reference window and a sliding detection window, or a sliding pair of windows over consecutive observations. The old and new windows can be compared with statistical tests, with the null hypothesis being that both windows are drawn from the same distribution.

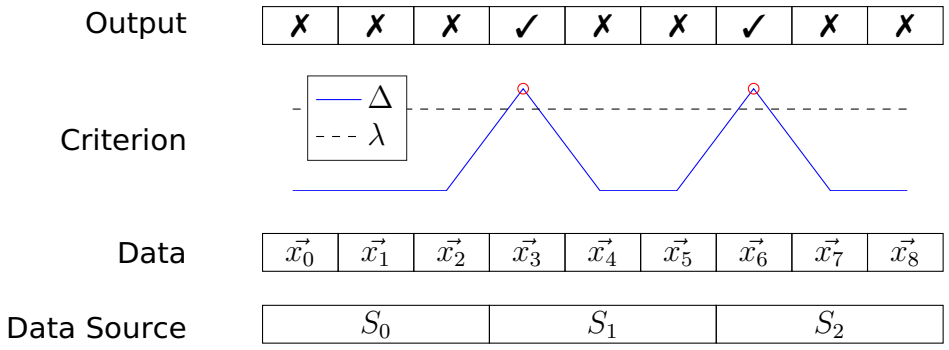



**Figure 2.18:** Distance between window distributions is expected to maximise around the sequence boundaries.

Such a detector would work by computing a distance[2] between distributions of two consecutive windows of observations, $\Delta = D(W_1, W_2)$, and signalling change when this statistic exceeds our threshold $\Delta > \lambda$. If this was the case, we should see a maximisation of $\Delta$ at the points where one distribution changes to another, as illustrated in Figure 2.18. The criterion in this example assumes two windows of one element each. Therefore it maximises when there is one example from each data source.

Detectors of this construction are dependent on a good choice of window size. For fixed-sized windows, their sizes need to be decided *a priori*, which poses a problem. A choice must be made along the continuum in Figure 2.16. A method may be intended for growing and shrinking sliding windows on the fly [17, 84, 176], but its choice of when to grow or shrink a window has a convenient use as a change point estimation.

A widely-used approach of this type is Adaptive Windowing (ADWIN) by Bifet and Gavaldà [17]. It keeps a variable-length window of recently seen examples, and attempts to find a "cut point" to split it into two sufficiently large and distinct subwindows. In its formulation as a change detector, change is signalled when the difference of the averages of two subwindows exceeds a computed threshold, $\lambda$. When this threshold is reached, the older subwindow is dropped and the remaining window is then regrown from subsequent observations. The meaning of sufficiently large and distinct is defined within a rigorous statistical test. To partition a window $W$ into two subwindows, $W_1$ and $W_2$, a threshold for the distinctness of the means is provided based on the Hoeffding bound. However in practice, the authors suggest a threshold based on the observation that the difference of the window averages will tend towards a normal distribution for large windows.

[2] Whilst we discuss this framework here using the notion of "distance", in practice this may also be a "divergence", i.e. a measure which may not be symmetric nor need to satisfy the triangle inequality.

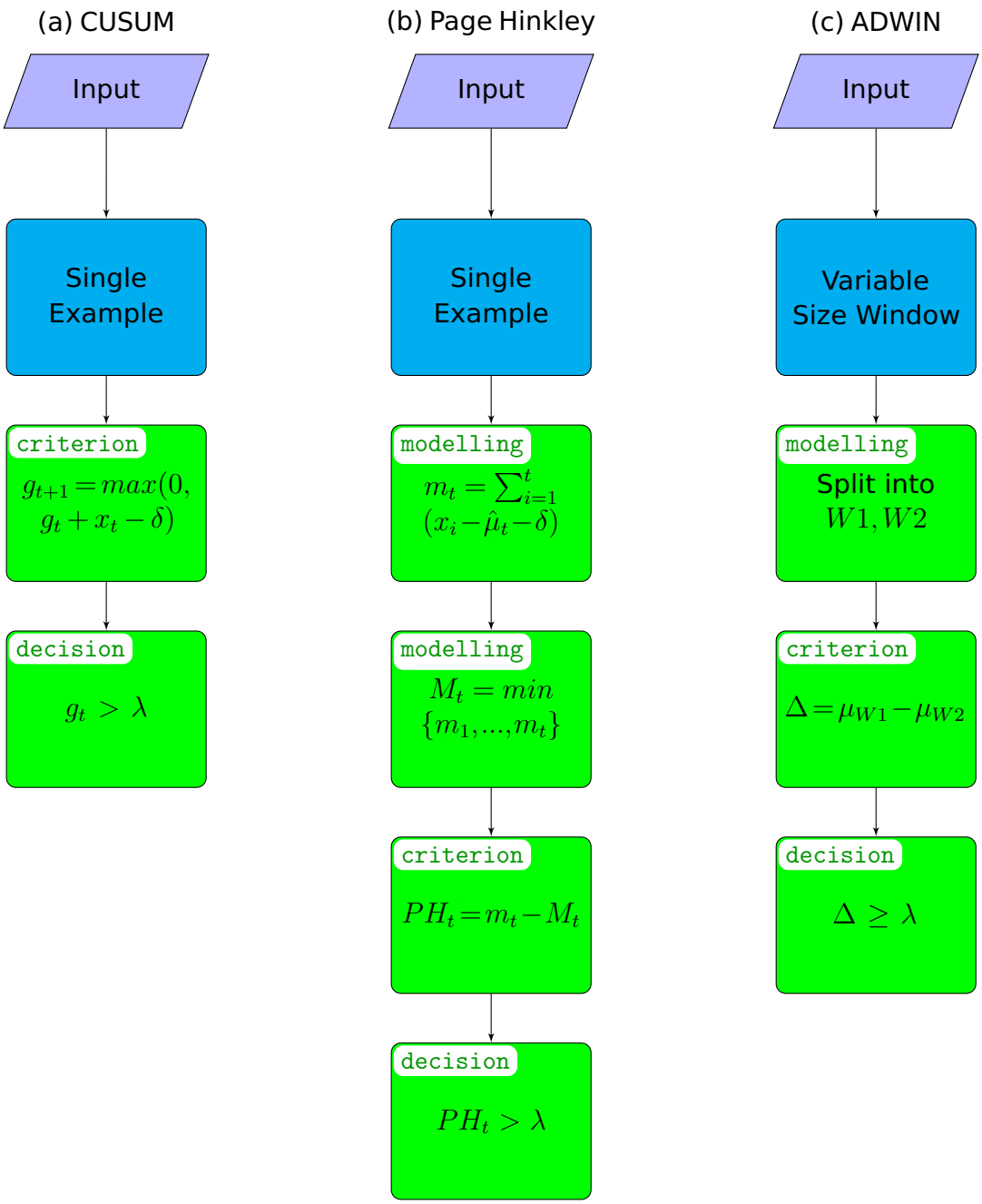



**Figure 2.19:** Flowcharts for CUSUM, PH and ADWIN detectors. Steps are coloured according to the same scheme as in Figure 2.10.

The SEQ1 algorithm [146] is an evolution of the ADWIN approach with a lower computational complexity. Cut-points are computed differently – where ADWIN makes multiple passes through the window to compute candidate cut-points, SEQ1 only examines the boundary between the latest and previous batch of elements. Secondly, the means of data segments are estimated through random sampling instead of exponential histograms. Finally, the authors employ a threshold based on the Bernstein bound instead of the Hoeffding bound to

establish whether two sub-windows are drawn from the same population. The Hoeffding bound was deemed to be overly conservative. In the SEED algorithm by Huang et al. [72], the data comes in blocks of a fixed size, so the candidate change points are the block's starting and ending points. Adjacent blocks are examined and grouped together if they are deemed sufficiently similar. This operation, termed 'block compression', removes candidate change points which have a lower probability of being true change points. Pooling blocks together amounts to obtaining larger windows, which in turn, ensures more stable estimates of the probabilities of interest compared to estimates from the original blocks. Change detection is subsequently carried out by analysing possible splits between the newly-formed blocks.

Figure 2.19 illustrates the pipelines of CUSUM, Page Hinkley and ADWIN detectors, using the colouring from the flowchart in Figure 2.10.

### 2.6.3 Multivariate Approaches

Modern methods for multivariate change detection usually require two components: a means to estimate the distribution of the incoming data, and a test to evaluate whether new data points fit that model. Estimation of the streaming data distribution is commonly done by either clustering, or multivariate distribution modelling. Gaussian Mixture Models (GMM) are a popular parametric means to model a multivariate process for novelty detection, as in Zorriassatine et al. [187]. Tarassenko et al. [162] and Song et al. [155] use nonparametric Parzen windows (kernel density estimation) to approximate a model against which new data is compared. Dasu et al. [33] construct *kdq* trees to a similar effect. Krempl et al [90] track the trajectories of online clustering, while Gaber and Yu [50] use the deviation in the clustering results to identify evolution of the data stream. Kuncheva [102] applies $k$ means clustering to the input data and uses the cluster populations to approximate the distribution of the data.

Multivariate statistical tests for comparing distributions such, as Hotelling's $T^2$ test [70] need to be adapted into the sequential form over time windows of the data [102]. Bespoke statistics continue to be developed for this purpose [2, 135]. Kuncheva [102] introduces two multivariate detectors based on the likelihood that data from a pair of windows was drawn from the same distribution. The detectors are compared with Hotelling's $T^2$ test [70], where the two test samples are given by adjacent windows maintained on the data stream. Together these three methods take a parametric, semiparametric and nonparametric approach respectively. These three methods are used extensively in the remaining chapters as a baseline for multivariate change detection performance.

Consider a random vector $\vec{x}$. We assume that $\vec{x}$ are drawn from a probability distribution $P(\vec{x})$ up to a certain point $c$ in the stream, and from a different distribution thereafter. The objective is to find the change point $c$. We can estimate $P$ from the incoming examples and compute the likelihood $\mathcal{L}(\vec{x}|P)$ for subsequent examples. A successful detection algorithm will be able to identify $c$ by a decrease of the likelihood of the examples arriving after $c$. To estimate and compare the likelihoods before and after a candidate point, the data is partitioned into a pair of adjacent sliding time-windows of examples, $W_1$ and $W_2$.

The semi parametric detector – called the semi-parametric log-likelihood criterion (SPLL) comes as a special case of the log-likelihood framework, and is modified to ensure computational simplicity. Suppose that the data before the change comes from a Gaussian mixture $P(\vec{x})$ with $c$ components each with the same covariance matrix. The parameters of the mixture are estimated from the first window of data $W_1$. The change detection criterion is derived using an upper bound of the log-likelihood of the data in the second window, $W_2$. The criterion is calculated as

$$SPLL = \max\{SPLL(W_1,W_2), SPLL(W_2,W_1)\}. \tag{2.19}$$

where

$$SPLL(W_1,W_2) = \frac{1}{M_2} \sum_{\vec{x} \in W_2} (\vec{x} - \vec{\mu}_{i*})^T \Sigma^{-1} (\vec{x} - \vec{\mu}_{i*}). \tag{2.20}$$

where $M_2$ is the number of objects in $W_2$, $\mathbf{\Sigma}$ is the common covariance matrix and

$$i* = \operatorname{argmin}_{i=1}^{c} \left\{ (\vec{x} - \vec{\mu}_i)^T \Sigma^{-1} (\vec{x} - \vec{\mu}_i) \right\} \tag{2.21}$$

is the index of the component with the smallest squared Mahalanobis distance between $\vec{x}$ and its centre.

If the assumptions for $P$ are met, and if $W_2$ comes from $P$, the squared Mahalanobis distances have a chi-square distribution with $p$ degrees of freedom. The expected value is $p$ and the standard deviation is $\sqrt{2p}$. If $W_2$ does not come from the same distribution, then the mean of the distances will deviate from $p$. Subsequently, we swap the two windows and calculate the criterion again, this time $SPLL(W_2,W_1)$. By taking the maximum of the two, SPLL becomes a monotonic statistic.

Small values will indicate identical distributions in $W_1$ and $W_2$, while large values will indicate potential change. While SPLL has been found to produce a robust statistic [102] which can sense changes in the variance of the data, its assumptions are rarely met in real-life data streams. This makes it difficult to set up a theoretical threshold or determine a confidence interval. This difficulty is not uncommon for change detection criteria, especially the semi-parametric and the non-parametric ones.

It should be noted that the SPLL and Hotelling detectors are closely related to $\chi^2$ and $T^2$ multivariate control charts respectively. From MacGregor and Kourti [118], the statistic for the $\chi^2$ chart is calculated as

$$\chi^2 = (\vec{x} - \tau)^T \Sigma^{-1} (\vec{x} - \tau) \tag{2.22}$$

where $\tau$ is the target value of the mean. Revisiting Equation 2.5, this is the squared Mahalanobis distance. They suggest an upper control limit on this statistic given by a Chi-squared distribution with degrees of freedom equal to data dimensionality. SPLL computes this statistic from two independent samples (and from the clustering of $W_1$), whereas the chart computes it point-to-point.

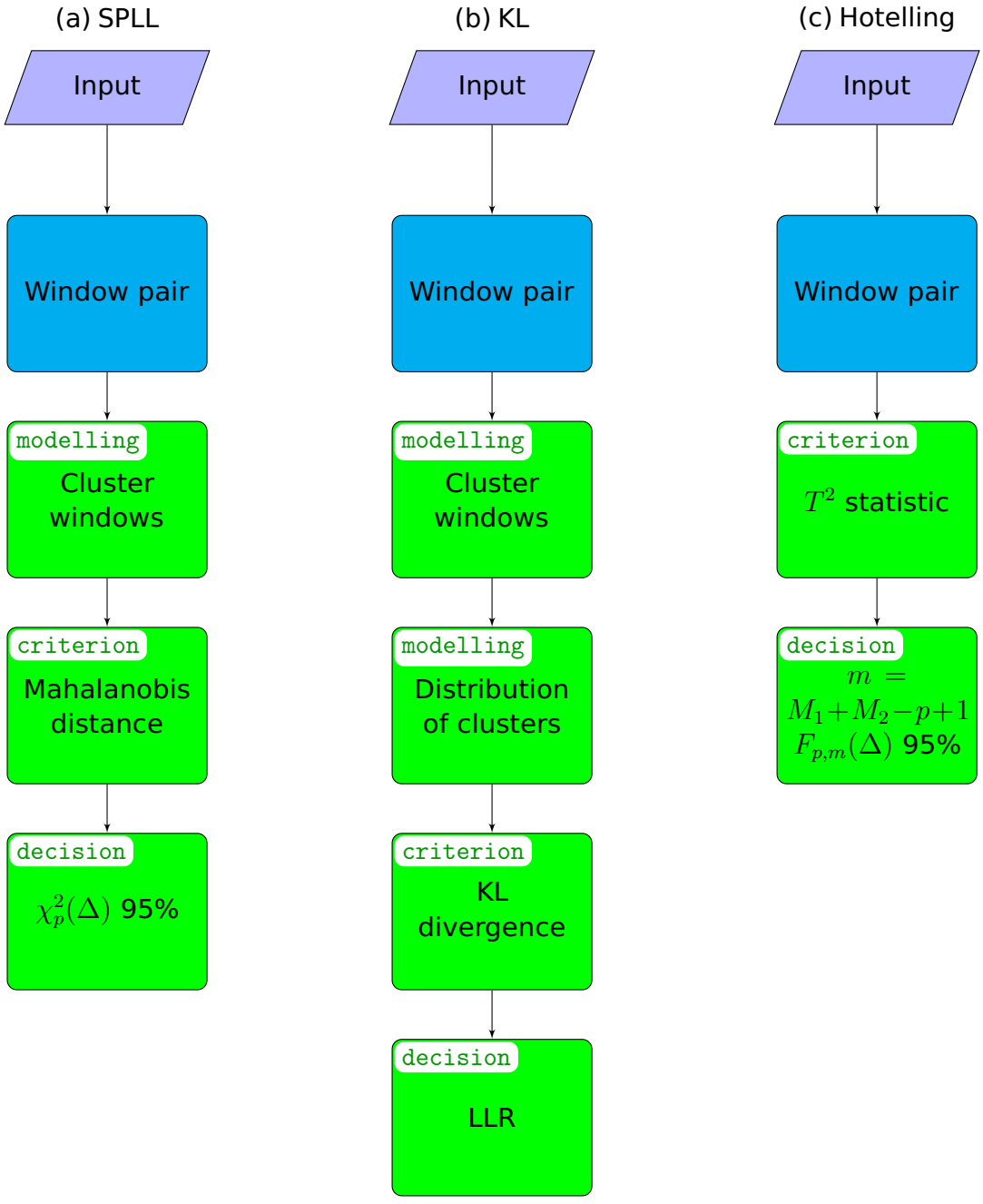



**Figure 2.20:** Flowcharts for each of the three multivariate detectors. Steps are coloured according to the same scheme as in Figure 2.10.

In the nonparametric approach, the KL divergence between discrete empirical distributions is used to compute a likelihood. The data distribution in

window $W_1$ is represented as a collection of $K$ bins (regions in $\Re^p$), with a probability mass value assigned to each bin. Call this empirical distribution $\hat{P}$. The data in $W_2$ is distributed in the bins according to the points' locations, giving empirical distribution $\hat{Q}$. Like SPLL, this is achieved by k-means clustering $W_1$ and then observing the nearest neighbour cluster membership of the points in $W_2$, although any approach that estimates a discrete probability mass function could be substituted. The criterion function is the Kullback-Leibler divergence [95] as in Equation 2.8, of the empirical distributions.

Note that we have only approximations of $P$ and $Q$. The usefulness of the $D_{KL}$ criterion depends on the quality of the approximations and on finding a threshold $\lambda$ such that change is declared if $D_{KL} > \lambda$.

This construction is very similar to the change detector of Dasu et al [33]. They expand $W_1$ until change is detected, giving a good basis for approximating $P$. On the other hand, $Q$ has to be estimated from a short recent window, hence the estimate may be noisy. Dasu et al. approximate the $P$ probability mass function by building $kdq$ trees which can be updated with the streaming data.

Figure 2.20 shows the pipelines of the three multivariate detectors coloured in the same manner as the flowchart in Figure 2.10.

## 2.7 Evaluating Change Detection Methods

Change detection methods are often evaluated indirectly in a domain-specific manner because of the diversity of related fields and applications. For example, when evaluating classifiers, error rate is one of the most important performance metrics [53]. When evaluating change detection methods for adaptive learning, we can ultimately judge the effectiveness of our approach by its performance in reducing the error rate of the classifier – its effectiveness

in achieving a domain-specific goal. Many adaptive learning papers evaluate their methods in this manner [176, 84, 54, 41]. For a reader wishing to evaluate and compare change detectors themselves, such metrics provide little insight about the characteristics of the change detector. The aim of this section is to establish how change detection methods can be evaluated, what measures should be recorded and what interpretations can be taken.

### 2.7.1 Metrics of Change Detector Performance

Not all change detection approaches offer magnitude estimation. Also, accuracy of the change time is directly related to the delay to detection. This leaves us with four quantities to measure. These quantities and their ideal values are summarized in Table 2.4.

An ideal change detector should detect *all* changes immediately and signal no false alarms [101]. Change is often only detectable after consuming more observations and the incurred delay is the Time To Detection (TTD), a measure of responsiveness.

**Table 2.4:** Metrics for evaluating change detectors and their ideal values

| Metric | Ideal Value | Semantic Meaning |
|---|---|---|
| Time To Detection | TTD $\to 0$ | How many observations passed on average between a change and the detector signalling. |
| False Alarm Rate | FAR $\to 0$ | The proportion of false alarms that were signalled. |
| Missed Detection Rate | MDR $\to 0$ | The proportion of changes that were not detected. |
| Average Run Length | ARL $\to ARL_I$ | How long on average the detector ran without signalling. |

The ideal value for the ARL depends on the data. Consider that we wish to compare the ARL of detectors. A detector which never signals will have the highest possible ARL, even though it was not desirable. In this situation we may

calculate an ideal ARL by observing the ARL of a perfect 'cheating' detector which uses the ground truth to signal immediately for every change ($TTD = 0$). Then detectors can be ranked by the closeness of their ARL to that figure. The rates of false alarm and missed detection as a proportion of total observations can be monitored as the False Alarm Rate (FAR) and Missed Detection Rate (MDR) respectively [52].

The evaluation of change detection techniques varies greatly across the literature, in part due to the variation of the application subfields. Basseville and Nikiforov [9] list five intuitive performance indexes for change detection algorithms.

- Mean time between false alarms (related to ARL).
- Probability of false detection (FAR).
- Mean delay to detection (TTD).
- Probability of non detection (MDR).
- Accuracy of the change time and magnitude estimates (only applicable if we measure these).

However, Table 2.5 summarises the evaluation techniques of a small cross-disciplinary sample of change detection literature. Adaptive learning literature usually focuses on the implicit measure; the error of the attached learner. The table serves to demonstrate the divergence of explicit metrics for change detector performance – which makes it difficult to perform a direct comparison of methods from the literature alone.

### 2.7.2 Datasets

There are a multitude of benchmark datasets available for the purposes of static classification, with both real and artificial data, for example from the UCI machine learning repository [112]. Datasets specifically for the problem of

**Table 2.5:** How change detectors are evaluated across a sample of the literature.

| Evaluation | Literature |
|---|---|
| Classification Error (Attached Learner) | [176, 84, 54, 34, 97, 41, 8, 144] |
| Classification Error (Change Detector) | [97] |
| TTD and FAR | [96] |
| FAR only | [94, 155] |
| ROC Curve | [102, 163] |
| Detect single event | [164] |
| Precision-Recall curve | [37] |
| FAR-Accuracy curve | [80] |
| ARL only | [183] |
| Detected/Late/False Alarm/Missed | [33] |

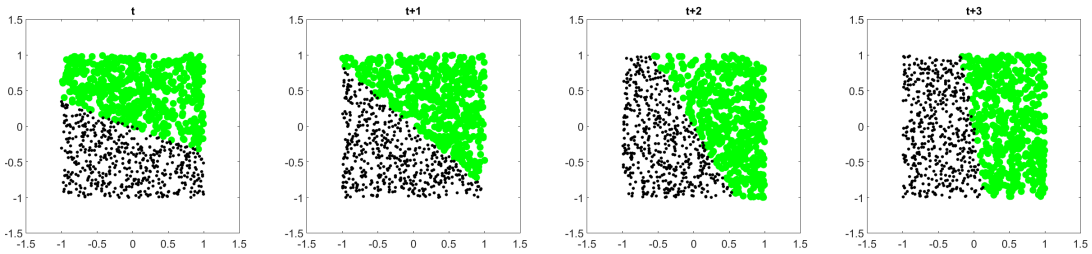



**Figure 2.21:** A rotating hyperplane problem changes the optimal classification boundary over successive time increments.

unsupervised change detection are far less common, especially those collected from genuine applications rather than toy data.

Whether a concept drift dataset has any value in our unsupervised situation depends on whether the dataset exhibits real or virtual concept drift. Recall Section 2.2.3. Unsupervised methods only have the opportunity to detect changes in $p(\vec{x})$. If the changes only affect $p(\vec{x}|y)$, then the data is not applicable to our problem. An example of this is the well known rotating hyperplane dataset [43]. A hyperplane representing the optimal classification boundary is rotated over time so that the nature of the classification problem is constantly evolving. The problem is illustrated in two dimensions in Figure 2.21, where we can see that the gradual change only affects $p(\vec{x}|y)$.

STAGGER by Schlimmer and Granger [148] is a well known adaptive learning system, and the accompanying concept generator has been used by many authors as a benchmark for evaluating such systems [34, 99, 176, 86, 20]. It

is used to generate very simple three-dimensional categorical data, with three distinct concepts.

Another popular dataset in the adaptive learning community is the SEA (Streaming Ensemble Algorithm) concepts [160]. SEA is an adaptive classifier ensemble which the authors evaluate in the presence of artificial concept drift. Like STAGGER, this dataset has been used as a benchmark in many subsequent works [86, 87, 127, 18, 41] and is commonly cited as an example of a concept drift dataset [20].

The KDD Cup 1999 dataset [81] is often used in anomaly detection literature, as it is one of the only fully labeled network intrusion datasets. It has $4{,}900{,}000$ examples and $42$ features extracted from seven weeks of network capture on a U.S. Air Force LAN [165]. Over this time period, the network was periodically subjected to 24 distinct categories of network attack, which produce anomalous patterns in the packet capture data.

### 2.7.3 Simulating Non-stationary Environments

Real world, labelled change detection and concept drift datasets are relatively sparse [20]. The limitation of simulated change datasets is that they do not reflect the complex causal relationships or statistical imperfection of changes that may occur in practice. It is therefore very difficult to simulate a wide enough variety of change to be representative of what a detector may encounter once deployed.

In the interest of being able to produce quantitative studies there has been research into the simulation of concept drift [134, 20], both from scratch and through interpretation of existing classification datasets. One way to simluate change is to take a labelled dataset, sample from one class before the change point and a different class after the change point. Being able to take advantage

of the separability challenges in the huge numbers of such publicly available datasets could result in much more robust evaluation of change detectors.

Narasimhamurthy and Kuncheva [134] suggest a framework for generating data to simulate changing environments, which was discussed in Section 2.2 in the context of describing change. This framework is extended by Bifet et al. [20] and presented as a means of generating concept drift. Given a labeled dataset with $k$ classes $y_1,...,y_k$, we can sample the data points from each class as a data source within the framework. Let $v_i(t)$ be a mixing function for data source $i$ at time $t$. Then the distribution $D(t)$ at time $t$ is described by:

$$D(t) = \{v_1(t), v_2(t), ..., v_k(t)\} \tag{2.23}$$

$$\sum_i v_i(t) = 1 \tag{2.24}$$

Assume that we have two data sources, $S_1$ and $S_2$, where their data points are sampled from $y_1$ and $y_2$ respectively. if we set the mixing functions at time $t$ to be $[v_1(t), v_2(t)] = [0,1]$ and $[v_1(t), v_2(t)] = [1,0]$ at $t+1$, this is an abrupt change from $y_1$ to $y_2$.

Under the same assumptions, we can generate gradual change with considerable flexibility through careful choice of the mixing functions. For example, change could be a simple linear function between sources or a sigmoid function as Bifet et al. [20] suggest.[3]

[3] Change generation libraries using this framework are available in both MATLAB and Java

- `https://github.com/LucyKuncheva/SDCD-Simulated-Data-for-Concept-Drift`
- `https://github.com/wfaithfull/meander`

## 2.8 Summary

This chapter has presented an overview of streaming change detection and its related fields, with discussion of the literature, common configurations, applications and terminology. It has been demonstrated that what constitutes change varies depending on the context. There has been a discussion of how change detectors can be broken down into modules, and how these modules can be taxonomised using existing categories from the literature. The change detection methods and building blocks used in subsequent chapters have been detailed in Section 2.6. An overview of how to evaluate change detection problems has been presented, along with a discussion of datasets and a framework for generating artificial change.

# Chapter 3

# PCA Feature Extraction for Multivariate Change Detection[1]

Much like classification, the detection of change relies upon there being a separable representation of each data source within the feature space. In a multivariate feature space, features which are stable in response to a change in the underlying data source are irrelevant in the context of change detection. If those features can be identified and discarded, we are left with a better representation of the change we are looking for. Principal Component Analysis (PCA) is a widely used statistical procedure for dimensionality reduction, feature extraction and feature selection across many disciplines. This chapter investigates the use of a PCA step for feature extraction and selection in the change detection pipeline. In the context of a detector pipeline from Section 7, this would constitute a preprocessing step as depicted in Figure 3.1.

## 3.1 Introduction

There are at least three caveats in choosing or designing a criterion for change detection from multidimensional unlabelled data. First, change detec-

[1] Most of this chapter was published as Kuncheva, L.I. and Faithfull, W.J., 2014. PCA feature extraction for change detection in multidimensional unlabeled data. IEEE transactions on neural networks and learning systems, 25(1), pp.69-80. It is an extension of work originally published as Kuncheva, L.I. and Faithfull, W.J., 2012, November. Pca feature extraction for change detection in multidimensional unlabelled streaming data. In Pattern Recognition (ICPR), 2012 21st International Conference on (pp. 1140-1143). IEEE.

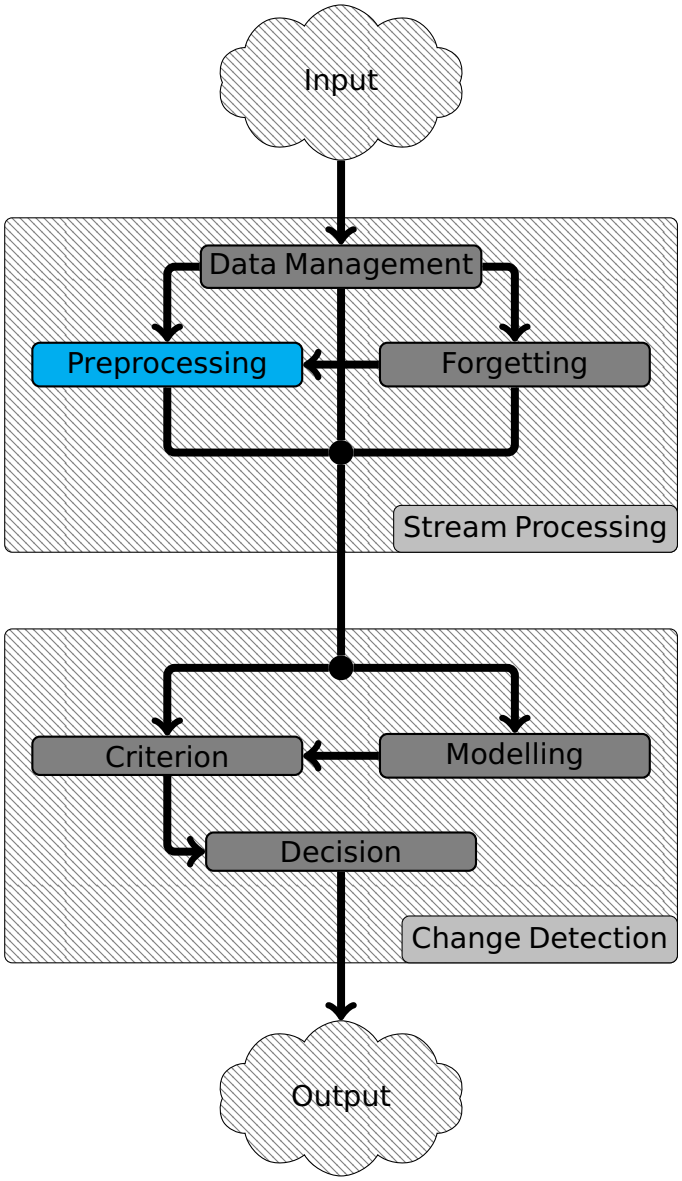
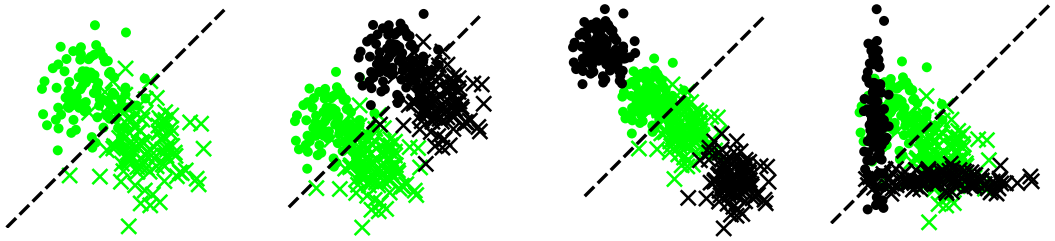



**Figure 3.1:** The contribution of this chapter can be used to map examples into a lower-dimensional representation as a useful preprocessing step for multivariate change detection.

tion is an ill-posed problem, especially in high-dimensional spaces. The concept of change is highly context-dependent. How much of a difference and in what feature space constitutes a change? For example, in comparing X-ray images, a hair-line discrepancy in a relevant segment of the image may be a sign of an important change. At the same time, if colour distribution is monitored, such a change will be left unregistered. The second caveat is that in the context of adaptive learning not all substantial changes of the distribution of the unlabelled data will manifest themselves as an increase of the error rate of the classifier. In some cases the same classifier may still be optimal for the new distributions.

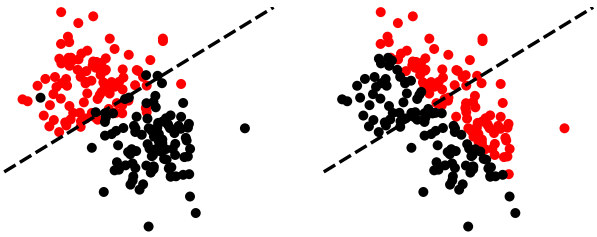
Figure 3.2 shows three examples of substantial distribution changes which do not affect the error rate of the classifier built on the original data. Conversely, classification error may decline with an adverse change in the class labels, without any manifestation of this change in the distribution of the unlabelled data. An example scenario is change of user interest preferences on a volume of articles. Figure 3.3 illustrates a label change which will corrupt the classifier but will not be picked up by a detector operating on the unlabelled data.

Finally, change detection depends on the window size. Small windows would be more sensitive to change compared to large windows.



(a) Original (b) Change 1 (c) Change 2 (d) Change 3

**Figure 3.2:** Example of 3 changes (plotted with black) which lead to the same optimal classification boundary as the original data (dashed line).



(a) Before change (b) After change

**Figure 3.3:** Example of a change in classification accuracy with no change in the unlabelled pdf.

To account for the uncertainties and lack of a clear-cut definition, we make the following starting assumptions: (1) changes that are likely to affect adversely the performance of the classifier are detectable from the unlabelled data, (2) changes of the distribution of the unlabelled data are reasonably correlated with the classification error, and (3) the window sizes for the old and the new distributions are specified.

Given the context-dependent nature of concept change, feature extraction can be beneficial for detecting changes. For example, extracting edge information from frames in a video stream can improve the detection of scene change [109]. A more general approach to change detection in multivariate time series is identifying and removing stationary subspaces [23].

In the absence of a bespoke heuristic, here it is proposed that principal component analysis (PCA) can be used as a general preprocessing step for feature extraction to improve change detection from multidimensional unlabelled incoming data. The theoretical grounds of the approach are detailed in Section 3.1.2. Section 3.2 describes the criterion for the change detection. Section 3.3 contains the experiment with 35 data sets, and Section 3.4 gives an illustration of change detection with feature extraction for a simple video segmentation task.

### 3.1.1 Rationale

Distribution modelling of multidimensional raw data is often difficult. Intuitively, extracting features which are meant to capture and represent the distribution in a lower dimensional space may simplify this task.

PCA is routinely used for preprocessing of multi-spectral remote sensing images for the purposes of change detection [153]. The concept of change, however, is different from the interpretation we use here. In remote sensing, 'change' is understood as the process of identifying differences in the state of an object in space by observing it at different times, for example a vegetable canopy. It is also well known in the context of anomaly detection in network traffic [26, 142, 73, 179], fault detection [173, 57, 61] and event detection [140, 62]. Recalling Figure 2.8, these can be related to our change detection problem here.

If there is no knowledge of what the change may be, it is not clear whether the representation in a lower-dimensional space will help. Our hypothesis is that, if the change is "blind" to the data distribution and class labels, the principal components with a smaller variance will be more indicative compared to the components with larger variance. This means that, contrary to standard practice [140, 62, 26, 142, 73, 57, 61], the components which should be retained and used for change detection are not the *most* important ones but the *least* important ones. Such blind change could be, for example, equipment failure, where the signal is replaced by random noise or signals bleeding into one another.

By leaving the most important principal components aside, we are not necessarily neglecting important classification information. PCA does not take into account class labels, therefore less relevant components may still have high discriminatory value.

Therefore we propose to use the components of lowest variance for detecting a change between data windows $W_1$ and $W_2$.

### 3.1.2 An Empirical Example

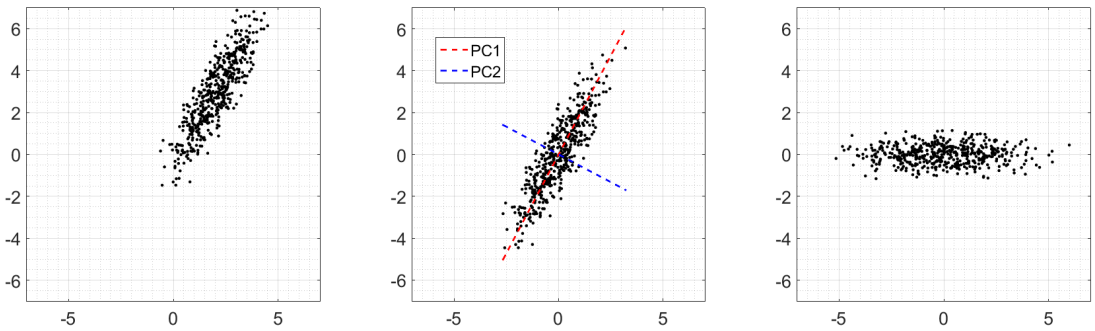



**Figure 3.4:** An illustration of the PCA process. Left: original Gaussian with $\mu = \left[\begin{smallmatrix}2\\3\end{smallmatrix}\right]$ $\Sigma = \left[\begin{smallmatrix}1 & 1.5\\1.5 & 3\end{smallmatrix}\right]$. Centre: data translated to the origin and principal component axes superimposed. Right: transformation of the original data into the principal component space.

The objective of PCA is to isolate the important characteristics of a set of data, transforming the variables into linearly uncorrelated *principal components* which best explain the variance in the data. The process is illustrated in Figure 3.4. An orthogonal linear transformation is applied to the data such that the first principal component lies along the axis of greatest variance. For an $m$ examples by $n$ features matrix $\mathbf{X}$, the principal component axes are the eigenvectors of the covariance matrix $\Sigma = cov(\mathbf{X})$. The relative magnitudes of the corresponding eigenvalues are proportional to the amount of variance explained by that principal component. The $n$ eigenvectors of $\Sigma$ are placed in descending eigenvalue order in the columns of a matrix, $\mathbf{W}$. Then $\mathbf{T} = \mathbf{XW}$ will transform the original data in $\Re^n$. Inspecting the cumulative sum of the eigenvalues we find the proportion of the total variance explained by each principal component. $\mathbf{W}$ can be used to transform new data into the same principal component space defined by the decomposition of $\mathbf{X}$. We use this process to inspect the change in newer data.

We will demonstrate empirically how the second component is likely to be more sensitive to changes than the first[2]. Suppose that we have two adjacent time windows of data, $W_1$ and $W_2$ of size $s$ where

$$W_1 = [\vec{x}_{t-s},...,\vec{x}_{t-1}]^T$$

$$W_2 = [\vec{x}_{t},...,\vec{x}_{t+s-1}]^T$$

$$|W_1| = |W_2| = s$$

The examples are drawn from a bivariate Gaussian. A PCA transformation of $W_1$ is computed, and used to transform $W_1$ and $W_2$ into the same principal component space.

We induce artificial change at $t+1$. Firstly, $W_2$ contains the data in $W_1$ translated to another point. The detectability of this change can be estimated

[2] Please refer to Kuncheva and Faithfull [104] for a proof.

from the Bhattacharyya distance between the distributions in $W_1$ and $W_2$. A larger Bhattacharyya distance implies that the change will be easier to detect because it is more distinct from the original distribution.

**Algorithm 1:** Process to generate Figure 3.6

$\mu = \begin{bmatrix}2\\3\end{bmatrix}$ $\Sigma = \begin{bmatrix}2 & 1{,}5\\1.5 & 3\end{bmatrix}$
- Sample $W_1 \sim \mathcal{N}(\mu, \Sigma)$.
- Perform PCA on $W_1$.
- Store transformed data as $W_{1PCA}$ and transformation coefficients as $\mathbf{W}$.

**for** $(x, y) \in \mathbb{R}^2$ **do**
- Let $W_2$ be $W_1$ translated to $(x, y)$
- Let $W_{2PCA} = W_2 \cdot \mathbf{W}$

**for** *each principal component* $i$ **do**
$\mathbf{I}_{x,y,i} = D_B(W_{1PCA*,i}, W_{2PCA*,i})$

**end**

**end**

Figure 3.5 illustrates the process we will use to assess the component sensitivity, which is described in Algorithm 1.

Using this process we arrive at Figure 3.6 which contains the images for the first and second principal components $I_{x,y,1}$ and $I_{x,y,2}$. This is the component sensitivity for an initial Gaussian with $\mu = \begin{bmatrix}2\\3\end{bmatrix}$ $\Sigma = \begin{bmatrix}2 & 1{,}5\\1.5 & 3\end{bmatrix}$ transformed into the principal component space. It is clear that if the data in $W_2$ were translated randomly, it is more likely to fall in an area of the space in which component 2 is more sensitive. Figure 3.7 shows in blue the regions where component 1 is more sensitive, and in yellow where component 2 is more sensitive. Intuitively, this effect is relative to the variance in each principal component. A component which exhibits less variance would require less movement in it's axis to produce a higher Bhattacharyya distance.

## 3.2 Choosing the change detection criterion

We use the semi-parametric log likelihood (SPLL) criterion detailed in Section 2.6. We argue our choice of SPLL by comparing it with three criteria used

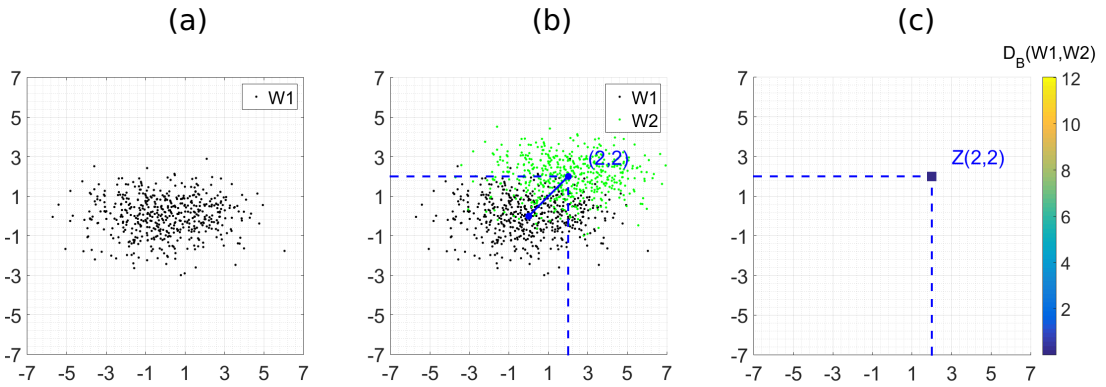



**Figure 3.5:** This shows how one pixel (2,2) in the sensitivity image is generated. (a) Data transformed into the principal component space. (b) $W_2$ (green) is translated to (2,2). (c) $Z(2,2)$ is the Bhattacharyya distance along each component $D_B(W_{1PCA*,i},W_{2PCA*,i})$, which sets the pixel intensity.

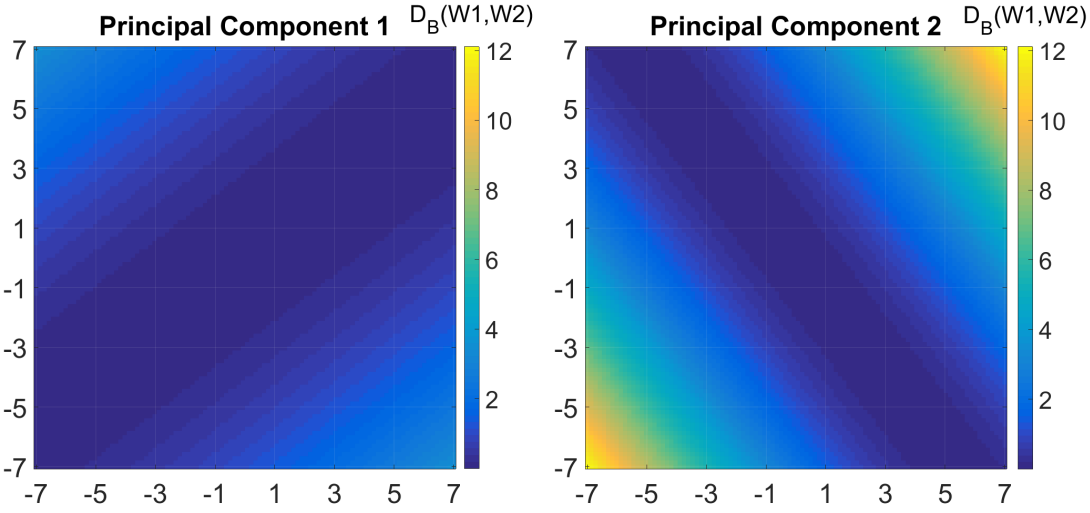



**Figure 3.6:** Left: Component 1 translation sensitivity in the PCA space. Right: Component 2 translation sensitivity in the PCA space.

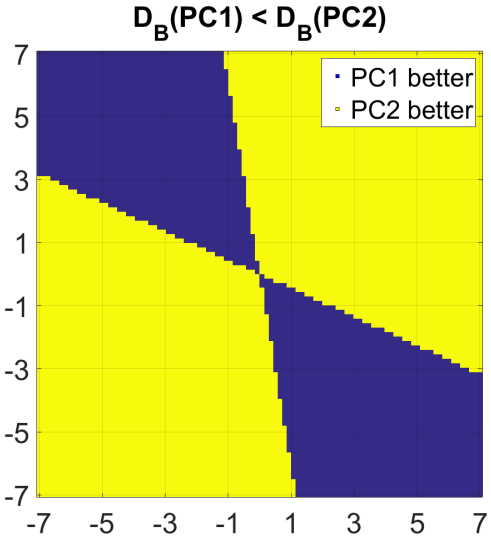



**Figure 3.7:** Regions where component 1 and component 2 are respectively more sensitive to translation change.

in multidimensional change detection: Hotelling's $T^2$, Multirank [117] and Kulback-Leibler (KL) distance [33].

### 3.2.1 Comparison with Hotelling, Multirank and KL

We have found that SPLL statistic compares favourably for detecting changes to its main competitor, the Hotelling $T^2$ test [102]. The reason behind this finding is that a Gaussian mixture is usually a more reasonable model than the single Gaussian assumed for the Hotelling test. The Hotelling criterion will not be able to detect change in the variance of the data, while the SPLL criterion is equipped to do so. The same holds for the nonparametric version of this test based on multi-dimensional ranking. The Multirank test [117] compares the medians of the distributions in the two windows but again leaves aside changes in the variance.

To support our criterion choice, we include here a simulation example. The experiment is detailed in Algorithm 2. We used three types of artificial change defined as follows.

**Translation** A new mean was sampled from $2z$, where $z \sim \mathcal{N}(0,1)$. $W_2$ was sampled anew from $P$ and the new mean was added (Figure 3.8 (b)).

**Random linear transformation** A random matrix $R$ of size $5 \times 5$ was generated, where each element was sampled from $\mathcal{N}(0,1)$. Window $W_2$ was sampled from $P$ and all objects were multiplied by $R$ (Figure 3.8 (c)).

**Change of variance** $W_2$ was sampled from a normal distribution with mean **0** and covariance matrix $\mathbf{\Sigma} \times D$, where $D$ is a diagonal matrix with diagonal elements sampled from $3|z|$, where $z \sim \mathcal{N}(0,1)$ (Figure 3.8 (d)).

Figure 3.8 shows scatterplots of the two windows in the space of the first two features.

**Algorithm 2:** Comparison experiment for Hotelling, Multirank, KL and SPLL.

**for** *1 to 100* **do**

- Sample 100 points as window $W_1$ from a 5-dimensional normal distribution with mean **0** and a diagonal covariance matrix $\Sigma$. The variances of the features are sampled from the positive half of the standard normal distribution.
- Denote this distribution by $P$. Sample Window $W_2$ from $P$ (with the same covariance matrix).
- Apply translation change to $W_2$ as $W_{2,T}$.
- Apply random linear translation change to $W_2$ as $W_{2,RLT}$.
- Apply change of variance to $W_2$ as $W_{2,CV}$.

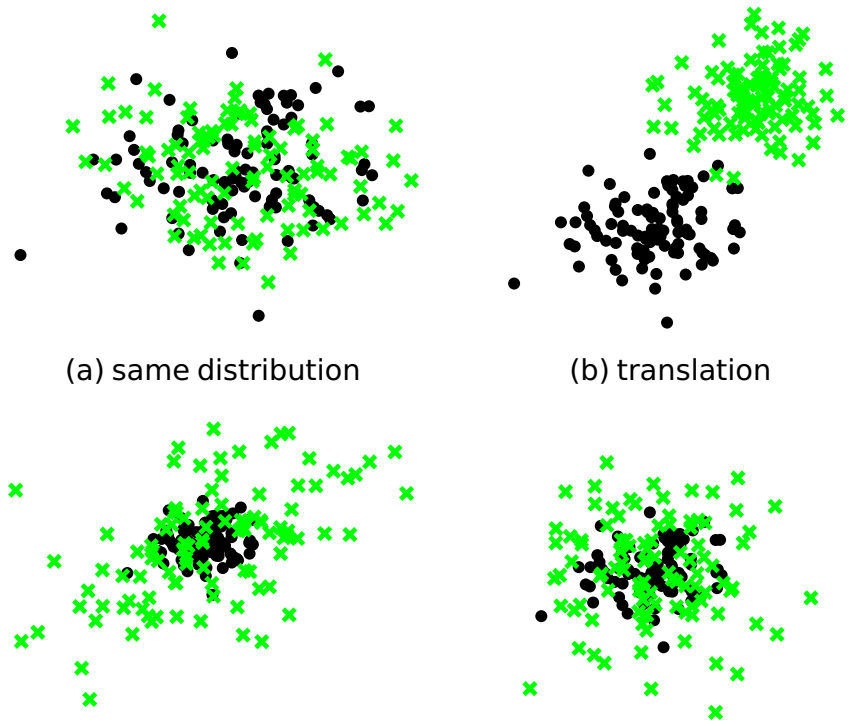
**for** *each* $\texttt{criterion} \in \{$ *SPLL, Hotelling, KL, Multirank* $\}$ **do**

- Produce ROC curve for `criterion`($W_1$, $W_{2,T}$)
- Produce ROC curve for `criterion`($W_1$, $W_{2,RLT}$)
- Produce ROC curve for `criterion`($W_1$, $W_{2,CV}$)

**end**

**end**

The procedure of generating $W_1$ and 4 versions of $W_2$ was repeated 100 times. Four change detection criteria were calculated: Kullback-Leibler (KL) divergence, the Hotelling's $T^2$ , Multirank [117] and SPLL. Both KL and SPLL were used with 3 clusters. Note that no thresholds were applied as we are evaluating the raw criteria values. The Receiver Operating Characteristic (ROC) curves were constructed for each criterion and each change type. Figure 3.9 shows the curves for the three changes. The graphs illustrate the behaviour of the four criteria. While for the mean change the two bespoke criteria (Hotelling and Multirank) are superior to KL and SPLL, the two latter changes favour SPLL. This is why we take SPLL for the experiment reported in the next section. We note that the choice of the criterion is not crucial for supporting our hypothesis that change detection will be aided by preserving the low-variance principal components.



(a) same distribution (b) translation

(c) random linear transformation (d) change of variance

**Figure 3.8:** Example of windows $W_1$ (black) and $W_2$ (green) for comparing the change detection criteria.

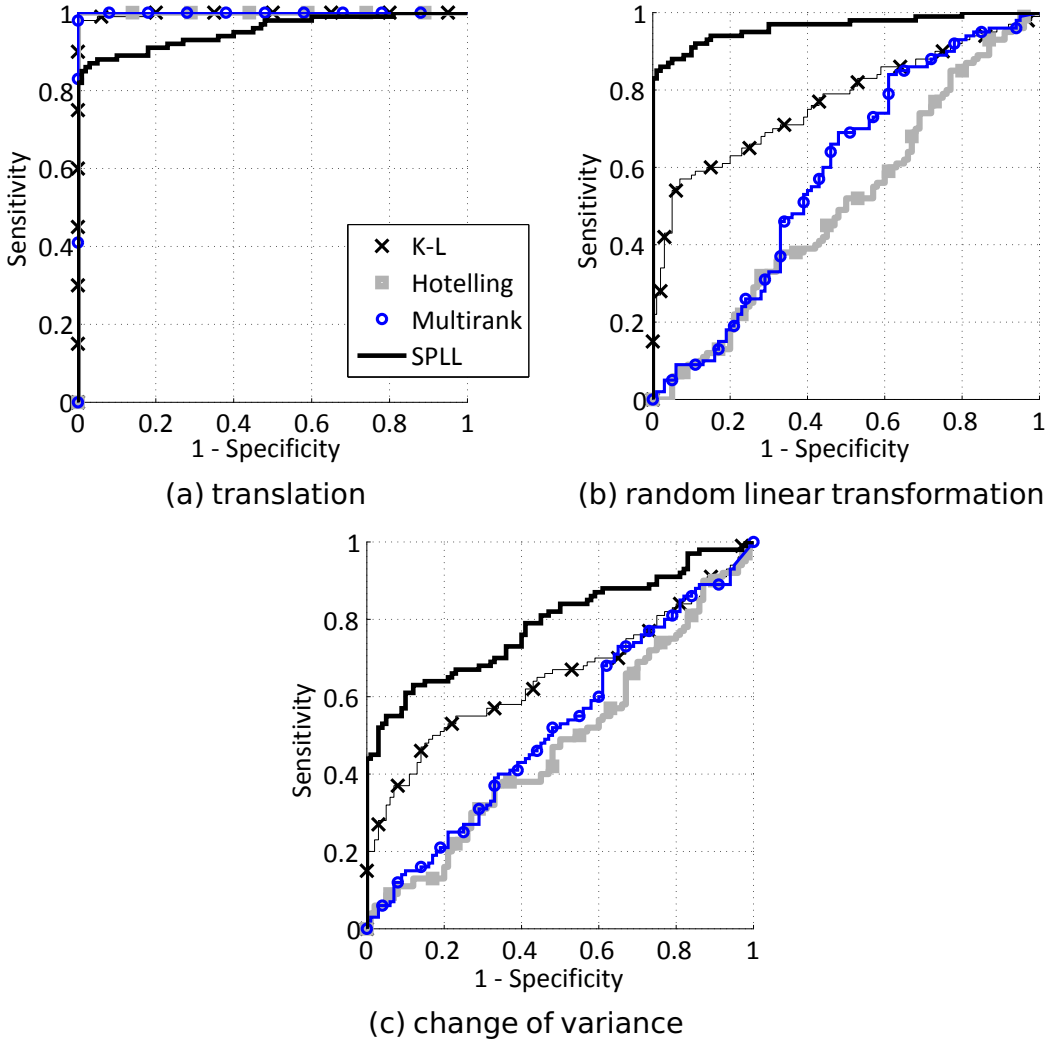
## 3.3 Experiment

### 3.3.1 Preliminaries

Our aim is to compare SPLL with and without PCA in order to demonstrate the benefit from the feature extraction.

**Acid test** It is difficult to find an acid test for change detection in unlabelled multidimensional data. Here we chose two change heuristics which could be regarded as instances of equipment failure.

*Shuffle Values.* A random integer $k$, $1 \leq k \leq n$, was generated to determine how many features out of $n$ will be affected. $k$ random features were chosen, and the values of each feature were randomly permuted within window $W_2$.




(a) translation

(b) random linear transformation

(c) change of variance

**Figure 3.9:** ROC curves for the 4 criteria and the three types of change.

*Shuffle Features.* Again, a random integer $k$, $1 \leq k \leq n$, was generated to determine how many features will be affected. $k$ random features were chosen, and *their columns* were randomly permuted within window $W_2$.

The Shuffle Values change resembles a case where a group of sensors stop working due to a technical fault and produce random readings within the sensor ranges. The Shuffle Features change can be likened to "bleeding" of signals into one another. We previously experimented with setting a number of features to zero or infinity but that seemed to be too easy a change to detect.

**Change detection is context-specific** We should also bear in mind that identifying changes is the first step in a process. The concept of "change"

depends on what we will be using the result for. There could be, for example, a scenario where a change in the mean of the distribution is irrelevant, and only a change in the variance should be flagged. The magnitude of change is also context dependent. How big a change should be accepted as worthy of triggering an alarm?

Therefore, here we do not offer a change detector as such. We investigate the ability of a *criterion* (SPLL and PCA+SPLL) to respond to changes. Setting up a threshold for this criterion is a separate problem. Such a threshold may be data-specific, and can be tuned to the desired level of false positives versus true positives.

**Indirect detection for classification** In the context of classification, there may be a problem-specific threshold on the classification error that should not be exceeded. Any changes of the distributions of the classes that do not lead to increased error can be perceived as insignificant.

As we argued in the Introduction, not all changes in the unconditional pdf will lead to change in the classification error. Thus a genuine change detected through the criterion may fail to correlate with the classification error. On the other hand, classification error may suffer with no change in the distribution of the unlabelled data. Even though such a correlation is an *indirect* quality measure, we include it here because of the importance of classification performance measure.

### 3.3.2 Experimental protocol

The experiment was run on 35 data sets listed alphabetically in Table 3.1, with differing numbers of instances, features and classes. The sets were

sourced from UCI [112] and a private collection. All data sets were standardised prior to the experiments.

**Algorithm 3:** Method for Experiment 1.

$K = \{$ 0%, 50%, 80%, 85%, 90%, 95% $\}$
$M = 50$
**for** *1 to 50* **do**
- Take a stratified random sample of size $M$ as window $W_1$.
- Run PCA on $W_1$ and keep the components beyond the $K$% of dismissed variance. Denote the PCA-transformed and clipped data set as $W_{1,PCA}$.

**for** $i = 1$ *to* $100$ **do**
- Take a random sample of $M$ instances from the remaining data as the i.i.d. window $W_2$. Calculate SPLL for windows $W_1$ and $W_2$ as in (2.19) and store the criterion value in $b(i)$.
- Transform $W_2$ in the PC space using the eigenvectors of the retained components. Call this set $W_{2,PCA}$. Calculate SPLL for windows $W_{1,PCA}$ and $W_{2,PCA}$ as in (2.19) and store the result in $c(i)$.
- Apply a change (value shuffle or feature shuffle) to $W_2$ to obtain a new set called $W_2'$. Calculate SPLL for windows $W_1$ and $W_2'$ as in (2.19) and store the result in $b'(i)$.
- Transform $W_2'$ in the PC space using the eigenvectors of the retained components. Call this set $W_{2,PCA}'$. Calculate SPLL for windows $W_{1,PCA}$ and $W_{2,PCA}'$ as in (2.19) and store the result in $c'(i)$.
- Concatenate the values SPLL for the cases with and without a change, to obtain $B = [b,b']$ and $C = [c,c']$. Calculate the ROC curves from $B$ and $C$ and the areas under the curves (AUC). If our hypothesis is correct, the AUC for $B$ will be smaller than the AUC for $C$.

**end**
**end**

In the first experiment we examined the difference between change detection on raw data and PCA data. For the PCA feature extraction, we varied the proportion of dismissed variance as: $K = \{$ 0%(keep all components), 50%, 80%, 85%, 90% and 95% $\}$. For example, consider $K = 90\%$ and a 4-dimensional data set, whose eigenvalues are $\{12,8,5,2\}$. Taking the cumulative sum and dividing by the sum of the eigenvalues, the cumulative explained variance (in %) is $\{44,74,93,100\}$. The first three components explain 93% of the variance in the data. We dismiss these components and keep only the last component

which explains the remaining 7% of the variability of the data. The process for the experiment is described in Algorithm 3.

The purpose of the second experiment was to find out how the SPLL change statistic correlates with the classification accuracy with and without PCA.[3] Larger values of SPLL signify a change in the distribution, which is likely to result in lower classification accuracy. Therefore we hypothesise that SPLL in the selected PCA space results in a stronger negative correlation compared to SPLL calculated from the raw data. By carrying out 50 runs of this procedure for each data set, 50 correlation coefficients are obtained. The process for the experiment is described in Algorithm 4.

### 3.3.3 Results

**Experiment 1.** Figure 3.10 shows the mean difference AUC(PCA)$-$AUC(raw) across the 35 data sets as a function of the percentage of dismissed variance $K$. The differences are positive if the low-variance components are retained. Using the 35 data sets, we carried out a paired two-tailed t-test between AUC(raw) and AUC(PCA,$K$), for the 6 values of $K$. The test was applied only for values of $K$ for which the Jarque-Bera hypothesis test indicated normality of the pairwise differences of the AUC. For the remaining values of $K$ we used the Wilcoxon signed rank test for zero median of the differences. The circled points correspond to statistically significant differences. Thresholds $K = 90\%$ and $K = 95\%$ lead to significantly better change detection than raw data. Interestingly, using all principal components ($K = 0\%$) leads to significantly worse AUC compared to detection from raw data. One possible explanation for this finding is that PCA "fools" the clustering algorithm so that the (rough) approximation of the pdf as a mixture of Gaussians becomes inadequate.

[3] We used the SVM classifier from the MATLAB bioinformatics toolbox.

[4] For multiple classes, we applied SVM to all pairs of classes and labelled the data point to the class with the most votes.

**Algorithm 4:** Method for Experiment 2.

$K = \{$ 0%, 50%, 80%, 85%, 90%, 95% $\}$
$M = 50$
**for** *1 to 50* **do**

- Take a stratified random sample of size $M$ as the window with the training data, $W_1$, and train an SVM classifier on it.[4]

- Run PCA on $W_1$ and keep the components beyond the $K = 95\%$ of explained variance. Denote the PCA-transformed and clipped data set as $W_{1,PCA}$.

**for** $i = 1$ *to* $100$ **do**

- Take a random sample of $M$ instances from the remaining data as the i.i.d. window $W_2$. Calculate the classification accuracy of the SVM trained on $W_1$, say $a(i)$. Calculate SPLL for windows $W_1$ and $W_2$ as in (2.19) and store the result in $b(i)$.

- Transform $W_2$ in the PC space using the eigenvectors of the retained components. Call this set $W_{2,PCA}$. Calculate SPLL for windows $W_{1,PCA}$ and $W_{2,PCA}$ as in (2.19) and store the result in $c(i)$.

- Apply a change (described above) to $W_2$ to obtain a new set called $W_2'$. Calculate the classification accuracy of the SVM trained on $W_1$ and store in $a'(i)$. Calculate SPLL for windows $W_1$ and $W_2'$ as in (2.19) and store the result in $b'(i)$.

- Transform $W_2'$ in the PC space using the eigenvectors of the retained components. Call this set $W_{2,PCA}'$. Calculate SPLL for windows $W_{1,PCA}$ and $W_{2,PCA}'$ as in (2.19) and store the result in $c'(i)$.

- Concatenate the accuracies and the SPLL for the cases with and without a change, to obtain $A = [a,a']$, $B = [b,b']$ and $C = [c,c']$. Calculate and store the correlation between $A$ and $B$, and $A$ and $C$. If our hypothesis is correct, $A$ (accuracy) and $C$ (SPLL from PCA-transformed data) will have a stronger negative correlation than $A$ and $B$ (SPLL from raw data).

**end**
**end**

The points where the AUC for the raw data is significantly better than the one with PCA are enclosed in grey squares.

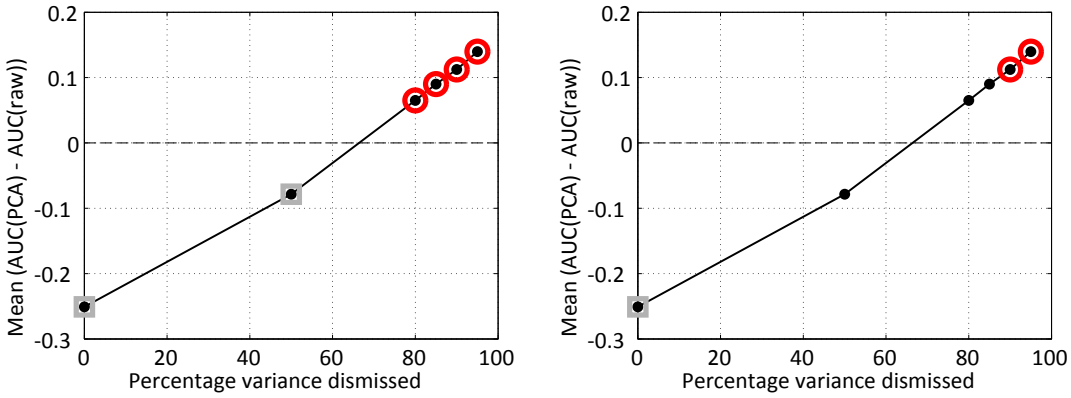
Figure 3.11 shows a scatterplot of the 35 data sets in the space of AUC(raw) and AUC(PCA,$K = 95\%$) for the two types of changes. The reference diagonal for which the PCA extraction does not make any difference is also plotted. It can be seen that most points are above the diagonal, demonstrating the improved change detection capability of the PCA features.

**Table 3.1:** Results from the experiments with two types of change.

| | | | | | | | Shuffle Values | | Shuffle Features | |
|---|---|---|---|---|---|---|---|---|---|---|
| Name | $N$ | $n$ | $c$ | $P_{\max}$ | $P_{\min}$ | #PCA | $\rho_{\text{raw}}$ | $\rho_{\text{PCA}}$ | $\rho_{\text{raw}}$ | $\rho_{\text{PCA}}$ |
| breast | 277 | 9 | 2 | 0.708 | 0.292 | 2.28 | -0.2983 | -0.3451• | -0.1696 | -0.2841• |
| contrac | 1473 | 9 | 3 | 0.427 | 0.226 | 2.18 | -0.2544 | -0.3320• | -0.1844 | -0.2983• |
| contractions | 98 | 27 | 2 | 0.500 | 0.500 | 16.38 | -0.8262 | -0.8169○ | -0.6719 | -0.6811• |
| ecoli | 336 | 7 | 8 | 0.426 | 0.006 | 2.94 | -0.5667 | -0.7546• | -0.6066 | -0.6161– |
| german | 1000 | 24 | 2 | 0.700 | 0.300 | 8.20 | -0.1395 | -0.3500• | -0.0918 | -0.3330• |
| glass | 214 | 9 | 6 | 0.355 | 0.042 | 4.32 | -0.4585 | -0.6713• | -0.3134 | -0.5876• |
| image | 2310 | 19 | 7 | 0.143 | 0.143 | 12.58 | -0.6516 | -0.8294• | -0.3206 | -0.6878• |
| intubation | 302 | 17 | 2 | 0.500 | 0.500 | 6.00 | -0.5045 | -0.6702• | -0.3571 | -0.6016• |
| ionosphere | 351 | 34 | 2 | 0.641 | 0.359 | 21.64 | -0.6755 | -0.7811• | -0.3253 | -0.5368• |
| laryngeal1 | 213 | 16 | 2 | 0.620 | 0.380 | 9.02 | -0.6387 | -0.6791• | -0.4225 | -0.5262• |
| laryngeal2 | 692 | 16 | 2 | 0.923 | 0.077 | 9.02 | -0.4272 | -0.5304• | -0.2845 | -0.4525• |
| laryngeal3 | 353 | 16 | 3 | 0.618 | 0.150 | 9.38 | -0.5976 | -0.6728• | -0.3683 | -0.5140• |
| lenses | 24 | 4 | 3 | 0.625 | 0.167 | 1.00 | 0.2319 | 0.2586– | 0.2524 | 0.1843• |
| letters | 20000 | 16 | 26 | 0.041 | 0.037 | 6.22 | -0.7074 | -0.8155• | -0.5456 | -0.7715• |
| liver | 345 | 6 | 2 | 0.580 | 0.420 | 1.98 | -0.3360 | -0.3856• | -0.1154 | -0.2779• |
| lymph | 148 | 18 | 4 | 0.453 | 0.014 | 5.48 | -0.2127 | -0.2466• | -0.0597 | -0.2015• |
| pendigits | 10992 | 16 | 10 | 0.104 | 0.096 | 8.12 | -0.9156 | -0.9436• | -0.8133 | -0.8996• |
| phoneme | 5404 | 5 | 2 | 0.707 | 0.293 | 1.02 | -0.3219 | -0.3285– | -0.1969 | -0.1443○ |
| pima | 768 | 8 | 2 | 0.651 | 0.349 | 2.02 | -0.3230 | -0.4637• | -0.0855 | -0.2192• |
| rds | 85 | 17 | 2 | 0.529 | 0.471 | 6.06 | -0.8013 | -0.8302• | -0.6035 | -0.6954• |
| satimage | 6435 | 36 | 6 | 0.238 | 0.097 | 31.98 | -0.9285 | -0.9012○ | -0.5080 | -0.6296• |
| scrapie | 3113 | 14 | 2 | 0.829 | 0.171 | 4.10 | -0.0832 | -0.0999– | -0.0438 | -0.3151• |
| shuttle | 58000 | 9 | 7 | 0.786 | 0.000 | 6.94 | 0.0709 | -0.4929• | 0.2515 | -0.4491• |
| sonar | 208 | 60 | 2 | 0.534 | 0.466 | 40.42 | -0.6630 | -0.7119• | -0.4413 | -0.5570• |
| soybean_large | 266 | 35 | 15 | 0.150 | 0.038 | 17.64 | -0.7492 | -0.9187• | -0.5760 | -0.8726• |
| spam | 4601 | 57 | 2 | 0.606 | 0.394 | 37.34 | -0.0492 | -0.1566• | -0.0074 | -0.1130• |
| spect_continuous | 349 | 44 | 2 | 0.728 | 0.272 | 28.14 | -0.3655 | -0.4721• | 0.0682 | -0.2115• |
| thyroid | 215 | 5 | 3 | 0.698 | 0.140 | 1.98 | -0.6682 | -0.6517– | -0.4921 | -0.6281• |
| vehicle | 846 | 18 | 4 | 0.258 | 0.235 | 12.94 | -0.7721 | -0.8396• | -0.4387 | -0.7444• |
| voice_3 | 238 | 10 | 3 | 0.706 | 0.076 | 4.20 | -0.6433 | -0.6895• | -0.4300 | -0.5481• |
| voice_9 | 428 | 10 | 9 | 0.269 | 0.016 | 4.00 | -0.5985 | -0.6552• | -0.4132 | -0.5356• |
| votes | 232 | 16 | 2 | 0.534 | 0.466 | 6.06 | -0.8193 | -0.7874○ | -0.6825 | -0.6254○ |
| vowel | 990 | 11 | 10 | 0.091 | 0.091 | 3.54 | -0.7907 | -0.8654• | -0.6813 | -0.7560• |
| wbc | 569 | 30 | 2 | 0.627 | 0.373 | 22.84 | -0.7728 | -0.7707– | -0.1849 | -0.4653• |
| wine | 178 | 13 | 3 | 0.399 | 0.270 | 4.98 | -0.8970 | -0.8933– | -0.7403 | -0.8029• |

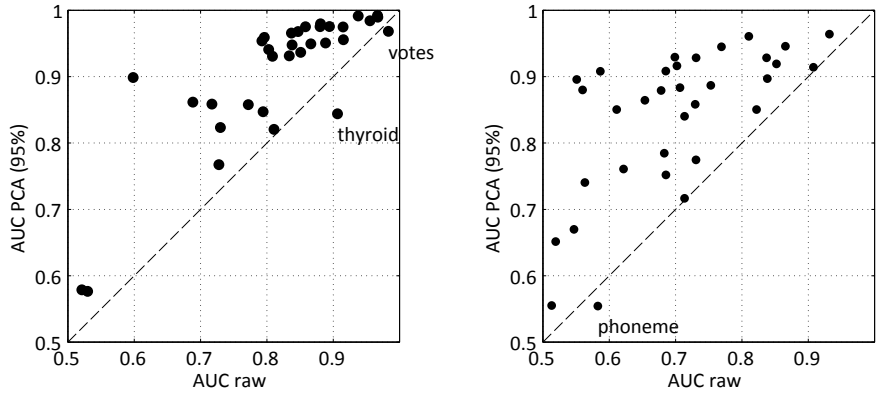
These comparisons of AUC are presented with the caveat that there is discussion over the coherency of AUC when used as an aggregated classification performance metric. Hand [65] states that using AUC is equivalent to using different metrics to evaluate different classification rules – in other words, the value of AUC depends to some extent on the model being evaluated. Ferri et al. [46] counter-argue that this model-dependent interpretation depends on the assumption that thresholds are chosen optimally.

Whilst we use AUC as a measure of performance in Experiment 1, Experiment 2 uses the correlation with the classification accuracy. The conclusions from Experiment 1 should be considered in the scope of the weaknesses of AUC as discussed above.




(a) Change: Value Shuffle (b) Change: Feature Shuffle

**Figure 3.10:** Average difference AUC(PCA)−AUC(raw).




(a) Change: Value Shuffle (b) Change: Feature Shuffle

**Figure 3.11:** Scatterplot of the 35 data sets in the space of AUC(raw) and AUC(PCA,$K = 95\%$).

**Experiment 2.** Table 3.1 shows the correlation coefficients averaged across 50 runs for each data set. The correlation coefficient between the classification accuracy and SPLL calculated from the raw data is denoted by $\rho_{\text{raw}}$, and the one for the features extracted through PCA, by $\rho_{\text{PCA}}$. Using the 50 replicas of the experiment, we carried out a paired two-tailed t-test for the data sets for which the Jarque-Bera hypothesis test indicated normality of the pairwise differences of the correlation coefficients. For the remaining data sets we used the Wilcoxon signed rank test for zero median of the differences. Statistically significant differences ($\alpha = 0.05$) are marked in the table with •, if PCA was better, and with ◦ if the raw data detection was better. Shown in the table are also the prevalences of the largest and the smallest classes in the data

($P_{\max}$ and $P_{\min}$) estimated from the whole data set. The column labelled '# PCA' contains the percentage of retained principal components.

Figures 3.12 and 3.13 show scatterplots of the 35 data sets in the space ($\rho_{\text{raw}}$,$\rho_{\text{PCA}}$) for the 6 values of $K$. The differences that were found to be statistically significant are marked with circles if favourable to PCA and with grey squares if favourable to the raw data.

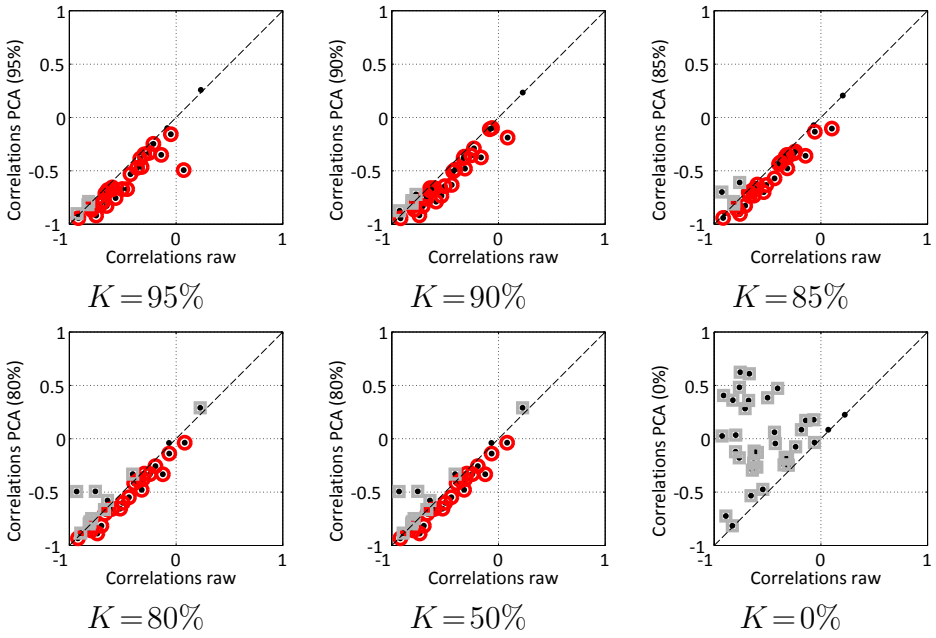



**Figure 3.12:** Shuffle Values: Scatterplot of the 35 data sets in the space ($\rho_{\text{raw}}$,$\rho_{\text{PCA}}$).

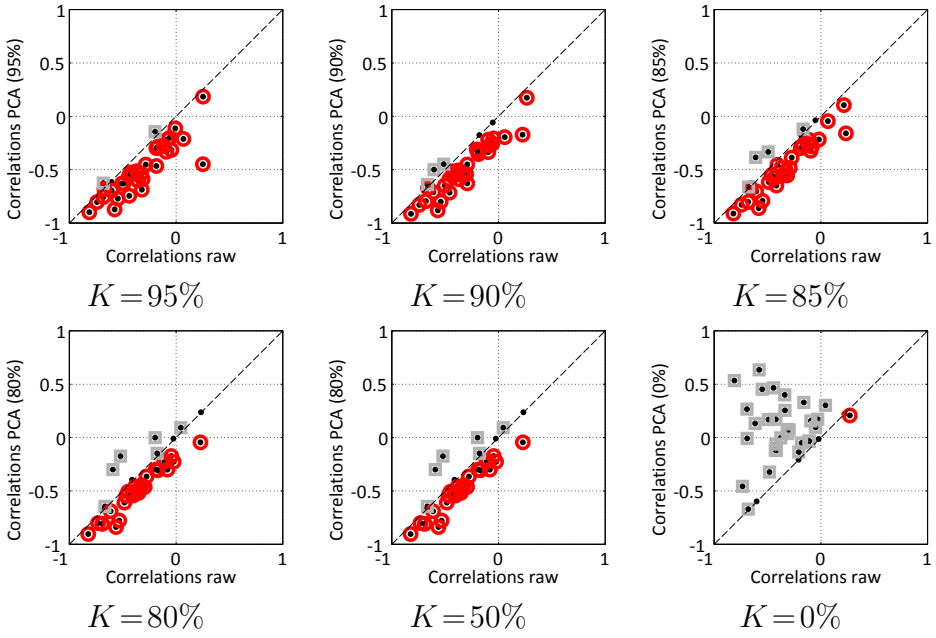


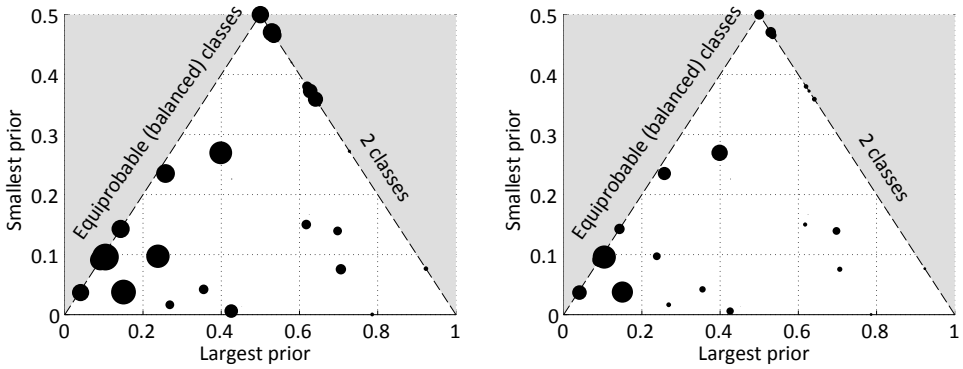

**Figure 3.13:** Shuffle Features: Scatterplot of the 35 data sets in the space ($\rho_{\text{raw}}$,$\rho_{\text{PCA}}$).

The results demonstrate that feature extraction through PCA leads to markedly better change detection and therefore stronger correlation with the classification accuracy than using the raw unlabelled data. As discussed in Section 2.2.3, these results should be interpreted within the scope of changes that affect the classification accuracy.

### 3.3.4 Further analyses

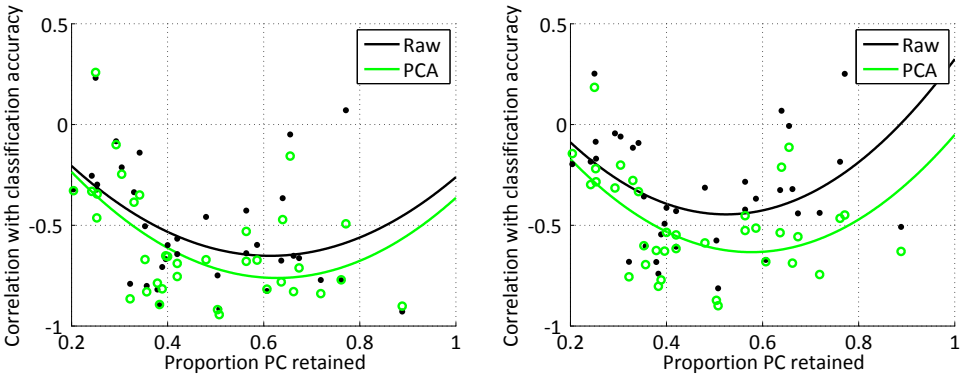
We carried out further analyses to establish which characteristics of the data sets may be related to the feature extraction success. Figure 3.14 shows a scatter plot where each point corresponds to a data set. The x-axis is the prior probability of the largest class and the y-axis is the prior probability of the smallest class. The feasible space is within a triangle, as shown in the figure. The right edge corresponds to 2-class problems, because the smallest and the largest priors sum up to 1. The number of classes increases from this edge towards the origin (0,0). The left edge of the triangle corresponds to equiprobable classes. The largest prior on this edge is equal to the smallest prior, which means that all classes have the same prior probabilities. This edge can be thought of as the edge of balanced problems. The balance disappears towards the bottom right corner. The pinnacle of the triangle corresponds to two equiprobable classes. The size of the marker signifies the strength of the correlation between SPLL with PCA and the classification accuracy.

The figure suggests that the PCA has a stable and consistent behaviour for multi-class, fairly balanced data sets (bottom left of the scatterplot). For smaller number of imbalanced classes (bottom right), the correlation $\rho_{\text{PCA}}$ is not very strong. Our further analyses did not find interesting relationship patterns between the data characteristics and the correlations, except for the pronounced dip for both correlations $\rho_{\text{PCA}}$ and $\rho_{\text{Raw}}$ with respect to the number of retained principal components.




(a) Change: Value Shuffle (b) Change: Feature Shuffle

**Figure 3.14:** Scatterplot of the 35 data sets in the space of the largest and smallest prior probabilities. The size of the marker signifies the strength of the correlation between SPLL with PCA and the classification accuracy.




(a) Change: Value Shuffle (b) Change: Feature Shuffle

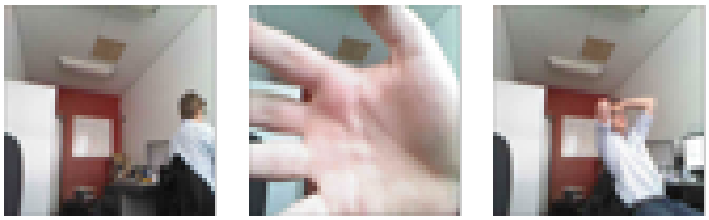
**Figure 3.15:** Correlation with classification accuracy as a function of the proportion of principal components retained.

Figure 3.15 shows the two correlations as functions of the proportion of retained principal components. The fit with the parabolas is not particularly tight but shows a tendency. For both heuristics, change detection is most related to the classification accuracy if about half of the principal components explain 95% of the variance, hence we retain the remaining half. The tendency on the left suggests that change detection is least related to classification accuracy where approximately two thirds or more of the principal components explain 95% of the variance, or as the diagonal covariance matrix approaches equal proportions. As was discussed in Section 3.1.2, the relative sensitivity of each component is proportional to the variance explained by each component. If the principal components were all of equal variance then they are all of equal sensitivity under our assumptions, eroding the benefit of feature extraction. An

intuition for the weak upward tendency on the right may be derived from the "peak effect" [79, 100] from feature selection, where a subset of features works better than the entire set. As can be expected, the PCA curve lies beneath the curve for the raw data, demonstrating the advantage of feature extraction for change detection. The pattern, however is similar for both correlation coefficients. It may be related to the type of changes and the way we induced them but may also benefit from a data-related interpretation. Since we are interested in comparing feature extraction to raw data change detection, we relegate the further analysis of this pattern to future studies.

## 3.4 A simple video segmentation

We applied the change detection with and without PCA to a simple video segmentation problem. A short video clip of an office environment was produced, with small movements of the chairs and the posture of one of the assistants in the office. The change was introduced in the middle part of the video by blocking the camera with the palm of a hand. The hand was made into a fist and opened again before removing it from view. Sample frames from the beginning, middle and end part of the video are shown in Figure 3.16.



(a) Beginning (b) Middle (c) End

**Figure 3.16:** Frames from the three parts of the video being segmented.

For the purposes of showcasing the feature extraction, we were only interested in the admittedly easy detection of the change in the middle. The features which formed the on-line multi-dimensional stream were the read, green and blue averages of each frame. We set $W_1$ to be the sequence of

the first 50 frames, and took a sliding window of 25 frames as $W_2$. The PCA was applied to $W_1$ only. Figure 3.17 plots the SPLL value with and without PCA across the frame sequence. Both criteria identify correctly the middle part with the change, but the values obtained through PCA are much larger. Figure 3.18 depicts the difference between SPLL with PCA and without PCA. Again, the results favour the feature extraction approach to change detection.

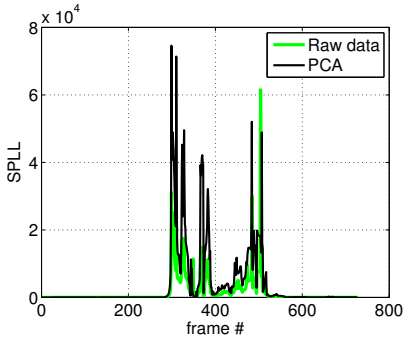



**Figure 3.17:** SPLL criteria values for the video frames

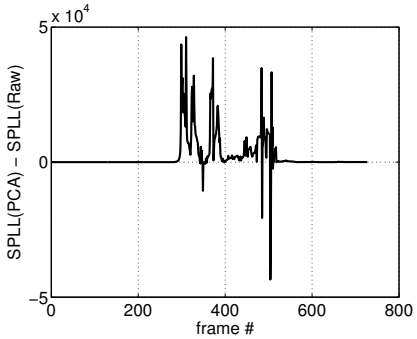



**Figure 3.18:** Difference between the two SPLL criteria

## 3.5 Conclusions

The lack of a rigorous methodology for feature extraction for the purposes of change detection in multidimensional unlabelled data has been noted in the literature. This chapter offers a step in this direction. Assuming change that is random with respect to the observed covariance, we argue that after applying PCA, the components with the smaller variance should be kept because they are likely to be more sensitive.

With regard to hypothesis (3) and scoped by the assumptions of hypothesis (1), the results certainly show a situation in which PCA is *beneficial* to change detection. In terms of PCA being *context-free*, the main two experiments within this chapter measure AUC and correlation with classification accuracy, respectively. These metrics belong within the same problem context, so further analysis is needed under a different context.

# Chapter 4

# Chaining Detectors[1]

Most change detection algorithms for multi-dimensional data reduce the input space to a single statistic and compare it with a threshold to signal change. Arrival at this threshold is typically tightly integrated with the change detection approach such as a confidence bound on the expected distribution of a computed statistic, or similar. Here it is proposed to 'chain' a multivariate and a univariate detector together, such that the univariate detector acts as a threshold. Figure 4.1 illustrates that the concept amounts to replacing the decision pipeline step for a compatible detector. This chapter investigates the performance of two generic methods for thresholding: bootstrapping and control charts. The methods are tested on a challenging dataset of emotional facial expressions, recorded in real-time using Kinect for Windows. Our results favoured the control chart threshold and suggested a possible benefit from using multiple detectors.

## 4.1 Introduction

We have seen that there are a variety of multivariate change detectors [115, 33, 102, 96], each with considerably different approaches. Many of these reduce the multidimensional data to a single statistic which should ideally

[1] Most of this chapter was published as Faithfull, W.J. and Kuncheva, L.I., 2014, August. On Optimum Thresholding of Multivariate Change Detectors. In Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR) (pp. 364-373). Springer, Berlin, Heidelberg.

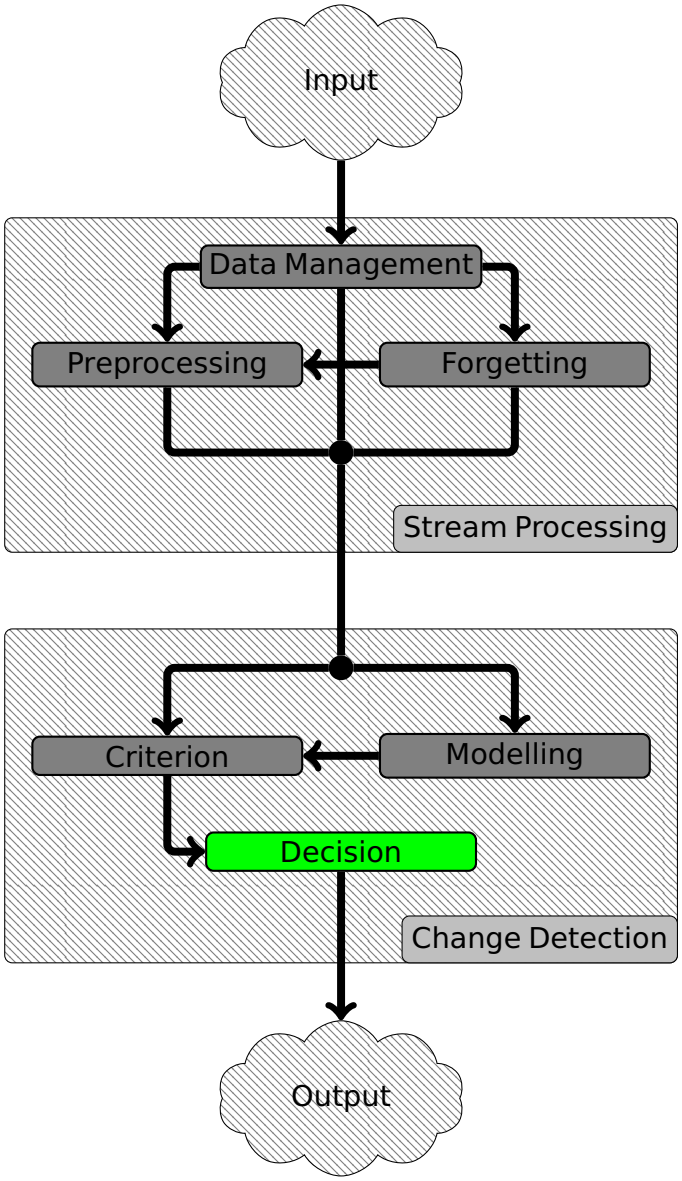



**Figure 4.1:** Where a poorly performing multivariate detector uses a static threshold, it may be improved by employing a univariate detector to monitor its statistic. The contribution of this chapter represents a replacement of the decision module within the pipeline.

correlate with the appearance of change, despite arriving at this statistic in very different ways. One of the main issues with such detectors is identifying a threshold on the statistic for flagging a change. We can take advantage of this homogeneity of approaches to swap out the mechanisms which threshold such statistics in order to signal change. Here we examine the suitability of two general approaches to setting a threshold: bootstrapping and control charts. Figure 4.2 illustrates the multivariate change detection process.

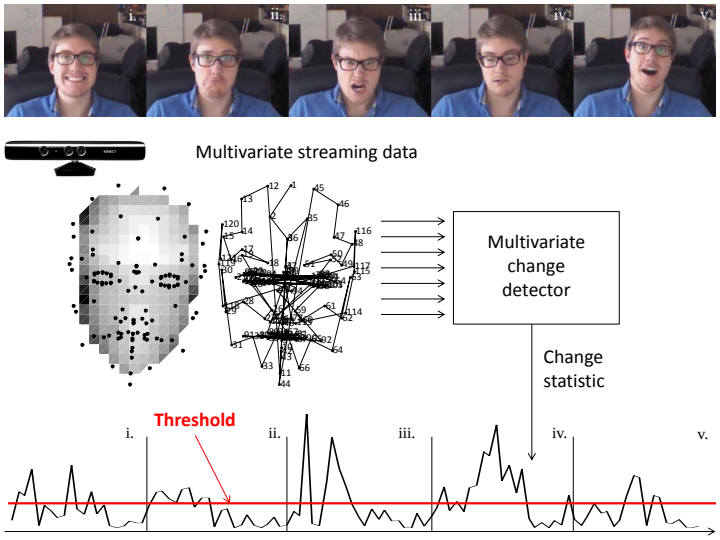



**Figure 4.2:** Illustration of the process of change detection in streaming multidimensional data and the role of the threshold. The data was obtained from Kinect while a participant was acting a sequence of emotional states: $i$. Happiness, $ii$. Sadness, $iii$. Anger, $iv$. Indifference, $v$. Surprise.

The rest of the chapter is organised as follows. Section 4.2 details some related work. Section 4.3 lays out our motivation. Section 4.4 describes the threshold setting approaches and our experimental study. The results are discussed in Section 4.5.

## 4.2 Related Work

There are differing approaches to the problem of detecting change in multivariate data. Lowry and Montgomery [114] reviewed multivariate control charts for quality control. Consider $n$ $p$-dimensional vectors of observations $\vec{x}_1, \vec{x}_2, ..., \vec{x}_n$. It is possible to simply create $p$ individual charts, one for each feature, not reducing the dimensionality of the data. However, this approach does not account for correlation between the features. Even truly multivariate control chart approaches such as the $T^2$ chart [71] can be equated to dimensionality reduction and thresholding, as it reduces the $p$ dimensions of the data to a single $T^2$ statistic. The list below demonstrates the inconsistency of approaches to setting such a threshold.

| Work: | Decision method |
|---|---|
| Zamba & Hawkins [180]: | $\lambda$ set according to a desired false alarm rate. |
| Song et al. [155]: | Original statistical test. |
| Dasu et al. [33]: | Monte Carlo Bootstrapping. |
| Kuncheva [104]: | Signficance of log-likelihood ratio. |

The scope of this work is concerned with establishing a method for threshold setting that is applicable to multiple approaches to change detection.

## 4.3 Motivation

In this work, we utilise the three multivariate change detectors due to Kuncheva [102] discussed in Section 2.6 – the parametric Hotelling detector, semi parametric log likelihood (SPLL) detector and the nonparametric Kullback-Leibler (KL) detector.

There are at least two reasons to consider alternative thresholding approaches. Firstly, some criteria such as $KL$ are not related to a straightforward statistical test that will give us a fixed threshold $\lambda$. Secondly, this chained approach may improve upon such statistical tests under certain conditions. To demonstrate, we will investigate the thresholding behaviour of the SPLL detector. The change statistic generated by SPLL is the mean squared Mahalanobis distance of each element of $W_2$ to the distribution approximated from $W_1$. Because this statistic is chi-squared distributed, the threshold to signal change is computed as $\chi^2_p(\Delta) < 0.05$ for the statistic in an $p$-dimensional chi-squared cumulative distribution function, the 95th percentile.

In practice, the SPLL assumptions are rarely met, which makes it difficult to set up a threshold or determine a confidence interval. This difficulty is not uncommon for change detection criteria in general. Bootstrap Monte Carlo

sampling and permutation tests have been suggested for estimating a suitable threshold [33, 83, 155]. We propose here to *chain* SPLL to a univariate change detector, which provides an adaptive threshold on the statistic.

Leaving aside whether the assumptions of SPLL are violated, why else might this be an improvement? Suppose that the statistic generated by SPLL has a very consistent, useful behaviour for detecting change in the context at hand, but it is not well represented in the $p$ values with regard to the 95th percentile.

**Algorithm 5:** Pseudocode for pilot experiment.

1. Take a 2000x10 matrix of gaussian random noise.
2. Multiply rows 501–1000 by a 10x10 matrix of gaussian random noise.
3. Slide a pair of 25-element windows over the rows, $W_1$ and $W_2$.
4. Run $SPLL(W_1,W_2)$ and store the change, p value and statistic.

The psuedocode for a pilot experiment is shown in Algorithm 5. We induce some detectable change in a 2000 element dataset, and iterate over this dataset with two 25-element windows. It is clear when we see the results in Figure 4.3 that the $p < 0.05$ threshold has delivered a very poor performance, with many false positives, despite an excellent representation of the true change in the generated statistic. This demonstrates the inherent danger in choosing a fixed threshold, even if that threshold is a confidence interval of a distribution.

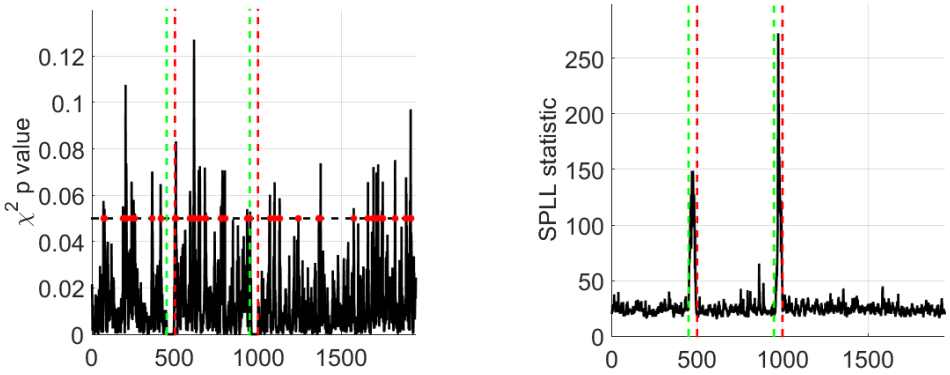



**Figure 4.3:** For $T = 25$; Left: the $p$ values and subsequent activations where $p < 0.05$. Right: the change statistic generated by SPLL. The true index of change is plotted vertically in red. Green indicates where the first observation from the new concept enters the leading window.

The reason for this performance is a poor choice of window size, $T = 25$. The data is noisy enough and the sample small enough that the estimation of the

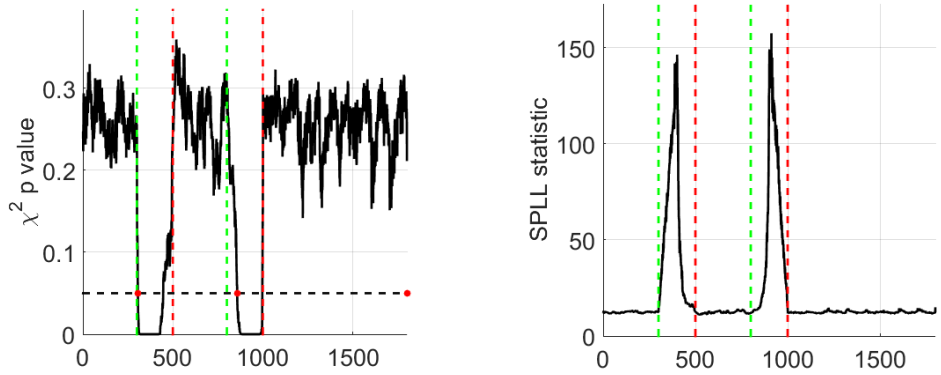



**Figure 4.4:** For $T = 100$; Left: the $p$ values and subsequent activations where $p < 0.05$. Right: the change statistic generated by SPLL. The true index of change is plotted vertically in red. Green indicates where the first observation from the new concept enters the leading window.

distribution from $W_1$ is often sufficiently different from $W_2$ to signal change. If we increase the window size to $T = 100$, we achieve the results in Figure 4.4. However, the pattern in the average Mahalanobis distance of the clusters is remarkably similar as it directly relates to the *scale* of the change in the context of previous observations. This suggests that an adaptive threshold may be able to perform better with a wider range of window sizes in practice, or perform equally well with fewer observations.

Recall SPLL's pipeline from Figure 2.20. Empirical observation suggests that the threshold can be improved. Therefore the proposition we investigate in this work is to replace the $\chi^2$ threshold with a bootstrapping procedure, or a univariate change detector.

## 4.4 Experiment: Bootstrapped Versus Control Chart Threshold

Here we examine two threshold setting approaches for the SPLL and KL statistics. These are compared with the baseline performance of the $T^2$ detector.

### 4.4.1 Bootstrapping

Let $|W_1| = M_1$. To determine a threshold, a bootstrap sample of $M_1$ objects is drawn from $W_1$. A discrete probability distribution $\hat{P}$ is approximated from this sample. Subsequently, another sample of the same size is drawn from $W_1$ and its distribution $\hat{Q}$ is evaluated. For example, if $\hat{P}$ is a set of bins, $\hat{Q}$ is calculated as the proportion of the data from the second bootstrap sample in the respective bins. The match between $\hat{P}$ and $\hat{Q}$ is estimated using, for example, KL distance (2.8), which gives the change statistic. Running a large number of such Monte Carlo simulations, a distribution of the change statistic is estimated, corresponding to the null hypothesis that there is no change (all samples were drawn from the same window, $W_1$). We can take the $K$th percentile of this distribution as the desired threshold. This approach was adopted by Dasu et al. [33] where the probability mass functions were approximated by a novel combination of kd-trees and quad trees, called kdq-trees. One drawback of this approach is the excessive computation load when a new threshold is needed.

### 4.4.2 Control Chart

A less computationally demanding alternative to bootstrapping is a Shewhart individuals control chart to monitor the change statistic. Inspired by this, our hypothesis is that the process underlying an appropriate change statistic will exhibit an out-of-control state when change occurs. Using a window of $T$ observations, we calculate the centre line $\bar{x}$ as the mean of the values of the statistic returned from the change detector, and its standard deviation $\hat{\sigma}$. The upper and lower control limits are calculated as

$$\bar{x} \pm 1.96 \frac{\hat{\sigma}}{\sqrt{T}}. \tag{4.1}$$

If either of the control limits are exceeded, change is signalled. This (rather naive) threshold estimation assumes that the change statistic has normal distribution, and that we have a sufficiently large window so as to get reliable estimates. The above value is for significance level $\alpha = 0.05$. The bootstrap threshold does not rely on any such assumption but is more cumbersome.

### 4.4.3 Experimental investigation

All thresholds considered here, including the threshold of the Hotelling method, are meant to control the type I error ("convict the innocent", or accepting that there is a change when there is none). If we set all these thresholds to 0.05, we should expect to have false positive rate less than that. Nothing is guaranteed about the type II error ("free the guilty", or missing a change when there is one). Thus we are interested to find out how the three chosen change detectors behave for the two type of thresholds, in terms of both error types.

### 4.4.4 Facial Expression Data

We chose a challenging real-life problem to test the change detectors. Sustained facial expressions of five emotions were taken to be the stable states, and the transition from one emotion to another was the change.

While a number of facial expression databases exist, we opted to use the Face Tracking toolkit distributed with the Kinect SDK. The Kinect software performs the necessary computer vision tasks to directly track facial geometry in real time.

This requires only a minimal experimental setup where a participant sits at a computer with a Kinect facing them, capturing real-time data about their posture and facial expression whilst they interact with the computer.

**Table 4.1:** Features extracted by the Kinect software.

| Feature | Description | Used |
|---|---|---|
| Face Points | 123 3D points on the face | |
| Skeleton Points | 10 3D points on the joints of the upper body | |
| Animation Units | 6 Animation Units $[-1,1]$ | ✓ |

**Table 4.2:** The six Kinect animation units and their equivalents in the Candide3 model.

| Animation Unit | Candide3 [3] | Description |
|---|---|---|
| AU0 | AU10 | Upper Lip Raiser |
| AU1 | AU26/27 | Jaw Lowerer |
| AU2 | AU20 | Lip Stretcher |
| AU3 | AU4 | Brow Lowerer |
| AU4 | AU13/15 | Lip Corner Depressor |
| AU5 | AU2 | Outer Brow Raiser |

The Kinect Face Tracking SDK utilises the Active Appearance Model (AAM) [40], taking into account the data from the depth sensor to allow head and face tracking in 3D. The features extracted by the Kinect are listed in Table 4.1. Those used in this experiment are checked. The "Animation Units" (AUs) refer to specific movements on the face defined in the Candide3 model [3]. The mapping between the Kinect AUs and the Candide3 equivalents is in Table 4.2

### 4.4.5 Data Capture

Each participant sat with their eyes trained on a computer screen, with a Kinect observing them. Emotional transitions are triggered by visual instructions. The participants were asked to hold their facial expression until instructed to change it. The duration of a facial expression is 3 seconds. The timestamps of these instructions are logged to provide the true positive values for the experiment. Thus each experimental run produces about $5$ expressions $\times$ 3 seconds $\times$ 30 FPS $=540$ frames. Figure 4.5 shows an example of one of the animation units throughout one run. The periods of sustained facial expressions are labelled. The initial warm-up period, as well as the transition periods of 7 frames are also indicated.

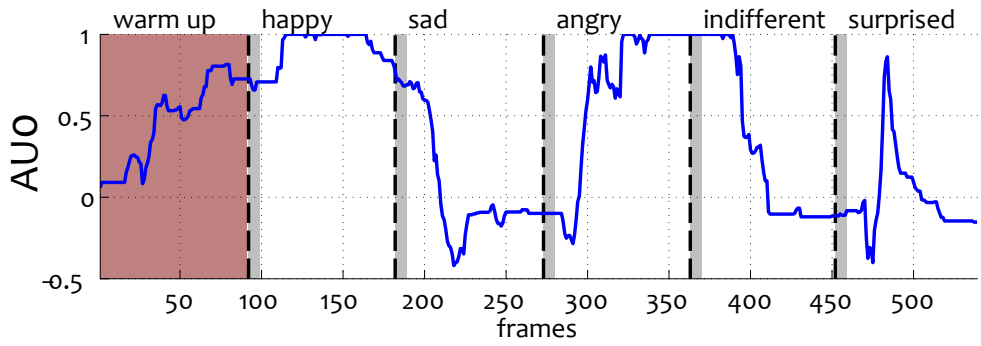



**Figure 4.5:** An example of an animation unit along one experimental run for collecting data. The dashed vertical lines are the time points where the participant is prompted to change their facial expression. The shaded regions are transition stages.

The process is facilitated by a bespoke application[2] written in the C# language, which utilises the Kinect SDK to retrieve frames from the sensor and extract the features. The application acts as a TCP client which connects to a server running in MATLAB, where the extracted features and timestamps are streamed in real-time, ready for analysis.

### 4.4.6 Methodology

The experiment was conducted using the AUs from six participants, each of whom recorded ten runs using the apparatus. Human reaction time to visual stimuli is 180-200 ms. In a recording at approximately 30 frames per second, a true positive detection should appear no earlier than $180/30 = 6$ frames after the labelled change (prompt to change the facial expression). For each run, we test Hotelling, KL Distance with Bootstrapping, KL Distance with Control Charts, SPLL with Bootstrapping and SPLL with Control Charts. The protocol in Algorithm 6 was followed for each run and for each participant.

Five hundred runs were carried out for determining the bootstrapping threshold.

[2] `https://github.com/wfaithfull/KinectConnect`

**Algorithm 6:** Experimental procedure.

1. Split the data into segments by label.
2. Sample a window $W_1$ of $T$ contiguous frames from a random segment $S$, with cardinality $|S| = M$ and random starting frame $F$, $7 \leq F \leq (M - T)$.
3. Sample $W_2$ from a random segment. If drawn from the same label as $W_1$, test for false positives, else test for true positives.
4. Calculate the threshold from $W_1$ using the chosen method.
5. Calculate change statistic from $W_1$ and $W_2$ and compare with the threshold. Store 'change' or 'no change', as well as the time taken to execute the iteration steps.
6. Repeat 1–5 $K$ times sampling $W_1$ and $W_2$ from the same label, $K$ times sampling $W_1$ and $W_2$ from different random labels. Calculate and return the true positive and false positive rates for the chosen detector and threshold.

To simulate a window of running change statistic only from data window $W_1$, we adopted the following procedure. A sliding split point $m$ was generated, which was varied from 3 to $T - 3$. This point was used to create windows $W_1'$, with data from 1 to $m$, and $W_1''$, with data from $m + 1$ to $T$. The statistic of interest was calculated from these sub windows, which were assumed to come from the same distribution.

Recalling Section 4.4.5, the data cycles through a warmup and 5 expressions in approximately 540 frames. Discounting the warmup period, each expression can be expected to last about $490/5 = 98$ frames. $T = 50$ was chosen in order that the window size be above 50% of an expression duration. While there is a great deal of literature on the subject of adaptive windowing [17, 54, 176], this is beyond the scope of this experiment. Such a technique could be used to set $T$. We set $K = 30$. The experiment was performed on a Core i7-3770K 4.6GHz Windows machine with 16GB RAM.

### 4.4.7 Results

We can examine the relative merit of the detectors and thresholds by plotting them on a Receiving Operating Characteristic (ROC) curve. The x-axis is

'1− Specificity' of the test, which is the false positive rate, and the y-axis is the 'Sensitivity' of the test, which is the true positive rate. Each run for each participant can be plotted as a point in this space. An ideal detector will reside in the top left corner (point (0,1)), for which true positive rate is 1 and false positive rate is 0. The closer a point is to this corner, the better the detector is.

Figure 4.6 shows 30 points (6 participants $\times$ 5 detector-threshold combinations).

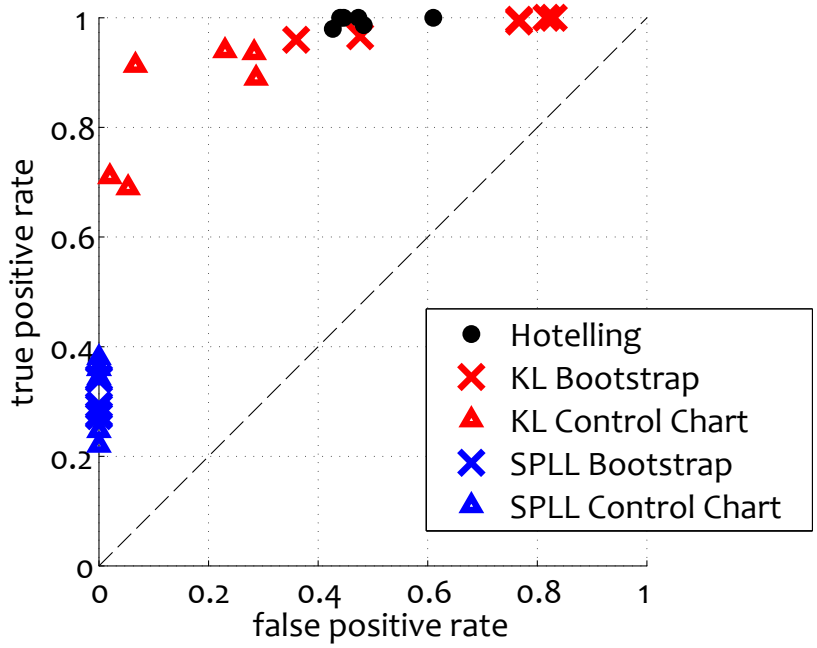



**Figure 4.6:** Results for the 5 detector-threshold combinations. Each point is the average (FP,TP) for one participant, across the $K = 30$ iterations and 10 runs.

Each point corresponds to a participant. The marker and the colour indicate the detector-threshold combination. The figure shows that, although the detectors are not perfect individually, the points collectively form a high-quality ROC curve.

All thresholds were calculated for level of significance 0.05. Applying this threshold is supposed to restrict the false positives to that value. This happened only for the SPLL detector. The price for the zero FP-rate is a low sensitivity, making SPLL the most conservative of three detectors. The Hotelling detector does not live up to the expectation of FP $< 0.05$. It is not guaranteed to have

that FP rate if the assumptions of the test are not met - clearly the situation here. Between this test and KL with bootstrap threshold, Hotelling is both faster and more accurate (lower FP for the same TP). The best combination for our type of data appeared to be the KL detector with the control chart threshold. It exhibits an excellent compromise between FP and TP, and is faster to calculate.

Interestingly, the threshold-setting approach did not affect SPLL but did affect the KL-detector. The control chart approach improved on the original bootstrap approach by reducing dramatically the false positive rate without degrading substantially the true positive rate.

We note that the way we sampled $W_1$ and $W2$ may have induced some optimistic bias because the samples from the same label could be overlapping. This makes it easier for the detectors to achieve low FP rates than it would be in true streaming data. Nevertheless, this set-up did not favour any of the detectors or threshold-calculating methods, so the comparison is fair.

The execution time analyses favoured unequivocally the control-chart approach to finding a threshold. Also SPLL is the slowest of the detectors, followed by KL and Hotelling. Therefore we recommend the KL-detector with a control-chart threshold.

## 4.5 Conclusions

This chapter examined the use of control charts as an alternative to the more traditional bootstrap approach for determining a generalised threshold for change detectors. Our experimental study with a real-life dataset of facial expressions taken in real time favoured the KL-detector with a control chart threshold.

We also observed that the statistical significance of the thresholds (type I error) is not matched in the experiments, except for the SPLL detector. The non-parametric bootstrap approach, was expected to give a more robust threshold, not affected by a false assumption about the distribution of the change statistic. The opposite was observed in our experiments for the KL-detector. The reason for this could be that the window was too small to account for the variability of the data sampled from the same label.

Reflecting on hypothesis (4), the success of the control chart threshold with the KL detector establishes the concept of chained detectors. The experiment demonstrates viability relative to the Hotelling's $T^2$ detector. What is still needed is a before and after comparison of multivariate detectors – first with their suggested thresholds, and then with a range of univariate detectors.

The work here raises an interesting point in relation to hypothesis (1), where it may lead to a weakening of assumptions for certain detectors. For example, the assumptions that the SPLL detector makes are related to the fact that it expects a certain distribution on its statistic – which is its thresholding mechanism. If we replace its threshold with a control chart, then the whole detector is making only the weaker assumption that it expects the average of the squared Mahalanobis distances to increase at the advent of change.

In terms of applicability along the continuum in Figure 2.8, the KL and SPLL detectors tested here take window size as a parameter. The concept of chaining detectors is independent of any temporal perspective on the data, which would be the responsibility of whatever approach is generating the statistic under observation.

# Chapter 5

# Ensemble Combination of Univariate Change Detectors for Multivariate Data[1]

Is it better to consider multivariate data "purely" i.e. as examples of a multivariate process, or should we inspect each feature individually? There is a large body of research on univariate change detection, notably in control charts developed originally for engineering applications. This chapter is an evaluation of 12 univariate detectors built into ensembles where each member observes a feature in the input space. A comparison is presented between the ensemble combinations and three established 'pure' multivariate approaches over 96 data sets, and a case study on the KDD Cup 1999 network intrusion detection dataset. It was found that ensemble combination of univariate methods consistently outperformed multivariate methods on the four experimental metrics. Figure 5.1 illustrates that the work constitutes a new self-contained multivariate change detector.

[1]Most of this chapter was published as Faithfull, W.J., Rodríguez, J.J. and Kuncheva, L.I., 2019. Combining univariate approaches for ensemble change detection in multivariate data. Information Fusion, 45, pp.202-214.

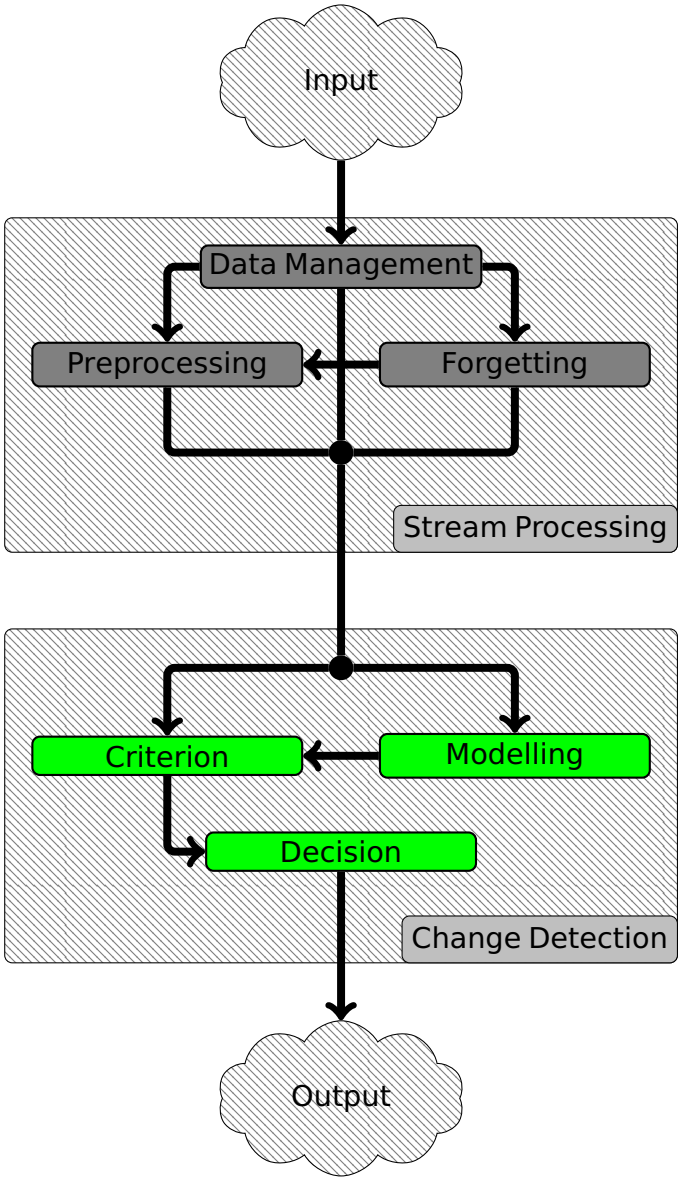



**Figure 5.1:** A multivariate detector can be created as an ensemble of univariate detectors, where each detector monitors a single feature of the input space. The contribution of this chapter is a fully fledged multivariate change detector.

## 5.1 Introduction

There are many approaches from the classification literature intended to monitor the error-rate of the incoming data and adapt a deployed classifier accordingly. The MOA (Massive Online Analysis) framework [19, 21] is a popular open source tool for data stream mining, providing a number of approaches for univariate change detection, all of which we evaluate in this work.

Inspiration is taken from a previous study [96] where Kuncheva uses classifier ensembles to detect concept change in unlabelled multivariate data. It is proposed here to build an ensemble of univariate detectors (which could be called a 'subspace ensemble') as a means of adapting established univariate change detection methods to multivariate problems. Our hypothesis is that such an ensemble should be competitive or better than 'pure' unsupervised multivariate approaches. We contribute the following:

1. An evaluation of which established univariate change detection methods are well suited to subspace ensemble combination over 96 common datasets.

2. Whether subspace ensembles outperform three established multivariate change detection methods, especially in high dimensions.

3. A reproducible reinterpretation of the widely used KDD Cup 1999 [112] network intrusion detection dataset as a change detection problem.

When generalising unsupervised change detection to multiple dimensions, the challenges proliferate – in how many features should we expect to see change before signalling? Can we reasonably assume that all features and examples are independent? Multivariate approaches often assume that each example is drawn from a multivariate process [102, 187, 2, 135]. Thus, we need not assume that the features are independent. Multivariate change detection attempts to model a multivariate process by means of a function to evaluate the fit of new data (an example or a batch) to that model. Some works monitor components independently (Tartatovsky et al. [164] and Evangelista et al. [42]), meaning that the approach is unable to respond to changes in the correlation of the components. Whether or not this is a disadvantage, depends upon the context of the change.

Change may have a different definition for different problems. For example, if we wish to be alerted when the value of a stock is falling, a sudden rise might be irrelevant. If using a control chart with upper and lower limits, only monitoring the lower limit might considerably lower the false alarm rate. If the problem is well known then a heuristic can be applied, but if that is the case, there is most likely training data available for a supervised approach. Unsupervised approaches must be robust in the face of unknown context. The change we wish to detect could be abrupt or gradual. It could be a single change or repeating concepts. When we move into multiple dimensions, there is even more scope for contextual properties to stretch our assumptions. Change could manifest itself in a single feature, all features, or any number of features in-between. From the novelty detection literature, Evangelista et al. [42] conclude that unsupervised learning in subspaces of the data will typically outperform unsupervised learning that considers the data as a whole. In the course of this work, we investigate whether this assertion is reproducible.

The dimensionality of the input data presents a potential challenge. Allipi et. al [4] analyse the effect of an increasing data dimension $d$ on change detectability for log-likelihood based multivariate change detection methods. They demonstrate that in the case of Gaussian random variables, change detectability is upper-bounded by a function that decays as $\frac{1}{d}$. Importantly, the loss in detectability arises from a linear relationship between the variance of the log-likelihood ratio and the data dimension. Evangelista et al. [42] propose that subspace ensembles are also a means to address the curse of dimensionality.

Multivariate detectors treat features as components of an underlying multivariate distribution [102]. We will term such detectors 'pure' multivariate detectors. For pure detectors to work well, the data dimensionality $d$ should not be high, as Allipi et al. argued, and the data coming from the same concept should be available in an *i.i.d* sequence. This is rarely the case in practice.

For example, Tartatovsky et al. [164] observe that the assumption that all examples are *i.i.d* is very restrictive in the domain of network intrusion detection.

The remainder of the chapter is organised as follows. Section 5.2 covers the background and related work for this problem. Section 5.3 details the methods used, explains our combination mechanism, and overviews the experimental protocol. Our results are presented in Section 5.5, and our conclusions follow in Section 5.6.

## 5.2 Related Work

Ensemble methods for monitoring evolving data streams is a growing area of interest within the change detection literature. There are recent surveys on the subject by Krawczyk et al. [89] and Gomes et al. [60]. The former observe that there has been relatively little research on the combination of drift detection methods. The publications that they review in this area [119, 177] deal with the combination of detectors over univariate input data, in contrast to our own formulation. The latter work introduces a taxonomy for data stream ensemble learning methods, and demonstrates the diversity of available methods for ensemble combination. Du et al. [38] utilise an ensemble of change detectors in a supervised approach for a univariate error stream. Alippi et al. [5] introduce hierarchical change detection tests (HCDTs) combining a fast, sequential change detector with a slower, optionally-invoked offline change detector.

In the classification literature, ensemble change detection commonly refers to using these techniques to monitor the accuracy of classifiers in an ensemble, in order to decide when to retrain or replace a classifier [20, 16, 15, 48]. Many of these established univariate methods for change detection are geared towards the supervised scenario which offers a discrete error stream [54, 8]. The Streaming Ensemble Algorithm (SEA) [160] was one of the first of many

**Table 5.1:** Methods for change detection in univariate data

| Method | References | Category |
|---|---|---|
| SEED | [72] | Monitoring Distributions |
| ADWIN | [17, 21] | Monitoring Distributions |
| SEQ1 | [146] | Monitoring Distributions |
| Page-Hinkley | [137, 21] | Sequential Analysis |
| CUSUM1 | [137] | Sequential Analysis |
| CUSUM2 | [21] | Sequential Analysis |
| GEOMA | [144, 143] | Control Chart |
| $\text{HDDM}_A$ | [48] | Control Chart |
| EDDM | [8, 21] | Control Chart |
| DDM | [54, 21] | Control Chart |
| EWMA | [144, 21, 143] | Control Chart |
| $\text{HDDM}_W$ | [48] | Control Chart |

**Table 5.2:** Methods for change detection in multivariate data

| Method | References | Category |
|---|---|---|
| SPLL | [102] | Monitoring Distributions |
| Log-likelihood KL | [102] | Monitoring Distributions |
| Log-likelihood Hotelling | [102] | Monitoring Distributions |

ensemble approaches for streaming supervised learning problems. However, instead of relying on a change detection, SEA creates an adaptive classifier which is robust to concept drift. Evangelista et al. [42] use a subspace ensemble of one-class Support Vector Machine classifiers in the context of novelty detection. The input space is divided into 3 random subspaces, each monitored by a single ensemble member. Kuncheva [96] uses classifier ensembles to directly detect concept change in unlabeled data, sharing the same problem formulation as this work.

## 5.3 Change detection methods

The methods we evaluated are detailed in Tables 5.1 and 5.2. The reader is referred to Section 2.6 for a thorough explanation of each. We chose to evaluate all the univariate detectors offered by MOA [19, 21], an open source project for data stream analysis. Our experiment performs an unsupervised

evaluation of all reference implementations of the `ChangeDetector` interface in the MOA package

`moa.classifiers.core.driftdetection` [2]

The interface contract implies the following basic methods to provide an input and subsequently check if change was detected:

```
public void input(double inputValue);
public boolean getChange();
```

All the univariate detectors are provided by MOA except CUSUM1, which is a CUSUM chart with upper and lower limits which was implemented in Java, and integrated into the experiment to serve as a baseline. We arrive at a final figure of 88 detectors, 3 of which are the multivariate approaches listed in Table 5.2, and the remaining 85 are ensembles of the univariate approaches with varying thresholds. The experimental details will be given in subsection 5.4. A full list of the 96 datasets and their characteristics can be found in Table 5.5. Our metrics for evaluation and our experimental protocol are addressed in subsection 5.4.1. Finally, we discuss the case study in subsection 5.4.2.

## 5.4 Ensemble combination of univariate detectors

In order to evaluate univariate approaches on multivariate data, we adopted an ensemble combination strategy whereby each member monitors a single feature of the input space. This approach is analogous to using a subspace ensemble with a subspace size of 1, with as many subspaces and detectors

[2] https://github.com/Waikato/moa/tree/master/moa/src/main/java/moa/classifiers/core/driftdetection

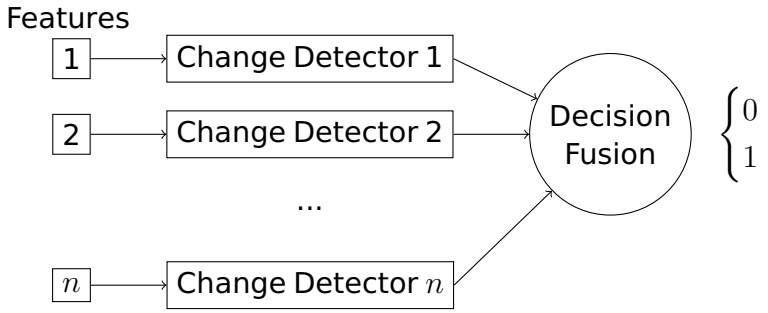



**Figure 5.2:** An illustration of the ensemble combination scheme. All change detectors are of the same type, but each monitors a different feature.

as the dimensionality of the input space. Using subspaces with a size greater than 1, as in Evangelista et al. [42], would require combination of multivariate approaches. Figure 5.2 shows an illustration of the ensemble combination scheme. In this set of experiments, the decisions are combined by a simple voting scheme with a variable threshold. Our naming convention for a single ensemble is as follows:

$$\text{DETECTOR - AGREEMENT THRESHOLD} \tag{5.1}$$

For example, ADWIN-30 refers to an ensemble of univariate ADWIN detectors, which requires 30% agreement at any given point to signal change. The multivariate detectors will simply be referred to as, KL, SPLL and Hotelling, as they are not ensembles.[3]

Diversity is an important consideration when building an ensemble, because it implies that the members will make different mistakes [98, 103] and there have been several analyses of ensemble diversity in evolving data streams [27, 60]. However, unlike in these works, our ensembles consist of identical detectors. Diversity is introduced through the differing input to each detector. On a related note, there will be redundant features in the datasets, which

[3]The ensemble of multivariate detectors is a special case, because, unlike the ensembles of univariate detectors, it consists of only three detectors. In this case, the number of members does not scale with the number of features. As such, there is no benefit in having a scale of agreement thresholds when there are only ever 3 ensemble members. We chose 50% as a simple majority out of 3.

will effect ensemble performance. Ideally this would be addressed through a feature extraction step, but such a measure is both difficult to generalise across datasets and outside the scope of this chapter. As our ensembles are created with identical members, no one type of detector can gain an advantage in the results due to drawing many redundant features by chance.

### 5.4.1 Experimental protocol

The main experiment of this chapter evaluates our multivariate change detection methods across the 96 datasets in Table 5.5. We evaluate the 3 multivariate detectors – SPLL, KL and Hotelling, an ensemble of these multivariate detectors, and 84 feature-wise ensembles of the univariate detectors with varying agreement thresholds, making a total of 88 detectors. A breakdown of the methods is presented in Table 5.3.

We note that when the thresholds in Table 5.3, are utilised on particularly small ensembles, the lower thresholds will become logically equivalent. For example, in ensembles with fewer than 20 members, the 5% and 1% thresholds will make the same decisions ($20 \times 0.5 = 1$). Since 43.33% of the datasets have more than 20 features, the difference in results between these lower thresholds will depend upon the larger datasets.

All the methods were evaluated against three rates of change: Abrupt, Gradual 100 and Gradual 300, for which we recorded separate sets of results. Algorithm 7 is a simplified pseudocode representation of the experiment. For each leg of the experiment, each detector is evaluated 100 times for each dataset. On each of these runs, we choose a random subset of the classes, and take this subset to represent distribution $P$ (before the change). The subset with the remaining classes is taken to represent distribution $Q$ (after the change). Points are then sampled randomly, with replacement, from the

**Table 5.3:** The ensembles and detectors evaluated in the experiment

| Ensemble | Agreement Thresholds | Count |
|---|---|---|
| SEED | 1, 5, 10, 20, 30, 40, 50 | 7 |
| ADWIN | 1, 5, 10, 20, 30, 40, 50 | 7 |
| SEQ1 | 1, 5, 10, 20, 30, 40, 50 | 7 |
| PH | 1, 5, 10, 20, 30, 40, 50 | 7 |
| CUSUM1 | 1, 5, 10, 20, 30, 40, 50 | 7 |
| CUSUM2 | 1, 5, 10, 20, 30, 40, 50 | 7 |
| GEOMMA | 1, 5, 10, 20, 30, 40, 50 | 7 |
| HDDM$_A$ | 1, 5, 10, 20, 30, 40, 50 | 7 |
| EDDM | 1, 5, 10, 20, 30, 40, 50 | 7 |
| DDM | 1, 5, 10, 20, 30, 40, 50 | 7 |
| EWMA | 1, 5, 10, 20, 30, 40, 50 | 7 |
| HDDM$_W$ | 1, 5, 10, 20, 30, 40, 50 | 7 |
| MV | 50 | 1 |
| | | Total 85 |

| Multivariate Detector | Count |
|---|---|
| SPLL | 1 |
| KL | 1 |
| Hotelling | 1 |
| | Total 3 |

$P$ and $Q$ sets – 500 examples in the abrupt case, 600 and 800 respectively in the gradual cases. Denote these samples by $S_1$ and $S_2$, respectively. We add a small random value to each example, scaled by the standard deviation of the data, to avoid examples that are exact replicas. In the abrupt case, $S_1$ and $S_2$ are concatenated to create a 1000-example test sample with i.i.d stream from index 1 to 500, coming from $P$, followed by an abrupt change at index 500 to another i.i.d. stream of examples coming from $Q$. To emulate gradual change over 100 examples, we take $S_1$ and $S_2$ as before, but do not concatenate them. At index 500, we sample with increasing frequency from $S_2$. The chance of an example coming from $S_1$ increases linearly from 1% at index 501 to 100% at index 600. Note that the class subsets for sampling $S_1$ and $S_2$ were chosen randomly for each of the 100 runs of the experiment.

**Table 5.4:** The first 48 datasets used in the main experiment.

$N$ is examples, $n$ is features and $c$ is classes.

| **dataset** | $N$ | $n$ | $c$ |
|---|---|---|---|
| abalone | 4177 | 8 | 3 |
| acute-inflammation | 120 | 6 | 2 |
| acute-nephritis | 120 | 6 | 2 |
| adult | 48842 | 14 | 2 |
| annealing | 850 | 31 | 3 |
| arrhythmia | 295 | 262 | 2 |
| balance-scale | 576 | 4 | 2 |
| bank | 4521 | 16 | 2 |
| blood | 748 | 4 | 2 |
| breast-cancer | 286 | 9 | 2 |
| breast-cancer-wisc | 699 | 9 | 2 |
| breast-cancer-wisc-diag | 569 | 30 | 2 |
| car | 1728 | 6 | 4 |
| cardiotocography-10clases | 2126 | 21 | 10 |
| cardiotocography-3clases | 2126 | 21 | 3 |
| chess-krvk | 28029 | 6 | 17 |
| chess-krvkp | 3196 | 36 | 2 |
| congressional-voting | 435 | 16 | 2 |
| conn-bench-sonar-mines-rocks | 208 | 60 | 2 |
| conn-bench-vowel-deterding | 990 | 11 | 11 |
| connect-4 | 67557 | 42 | 2 |
| contrac | 1473 | 9 | 3 |
| credit-approval | 690 | 15 | 2 |
| cylinder-bands | 512 | 35 | 2 |
| dermatology | 297 | 34 | 4 |
| ecoli | 272 | 7 | 3 |
| energy-y1 | 768 | 8 | 3 |
| energy-y2 | 768 | 8 | 3 |
| glass | 146 | 9 | 2 |
| haberman-survival | 306 | 3 | 2 |
| hayes-roth | 129 | 3 | 2 |
| heart-cleveland | 219 | 13 | 2 |
| heart-hungarian | 294 | 12 | 2 |
| heart-va | 107 | 12 | 2 |
| hill-valley | 1212 | 100 | 2 |
| horse-colic | 368 | 25 | 2 |
| ilpd-indian-liver | 583 | 9 | 2 |
| image-segmentation | 2310 | 18 | 7 |
| ionosphere | 351 | 33 | 2 |
| iris | 150 | 4 | 3 |
| led-display | 1000 | 7 | 10 |
| letter | 20000 | 16 | 26 |
| low-res-spect | 469 | 100 | 3 |
| lymphography | 142 | 18 | 2 |
| magic | 19020 | 10 | 2 |
| mammographic | 961 | 5 | 2 |
| miniboone | 130064 | 50 | 2 |
| molec-biol-promoter | 106 | 57 | 2 |

**Table 5.5:** The second 48 datasets used in the main experiment.

$N$ is examples, $n$ is features and $c$ is classes.

| **dataset** | $N$ | $n$ | $c$ |
|---|---|---|---|
| molec-biol-splice | 3190 | 60 | 3 |
| monks-1 | 556 | 6 | 2 |
| monks-2 | 601 | 6 | 2 |
| monks-3 | 554 | 6 | 2 |
| mushroom | 8124 | 21 | 2 |
| musk-1 | 476 | 166 | 2 |
| musk-2 | 6598 | 166 | 2 |
| nursery | 12958 | 8 | 4 |
| oocytes_merluccius_nucleus_4d | 1022 | 41 | 2 |
| oocytes_merluccius_states_2f | 1022 | 25 | 3 |
| oocytes_trisopterus_nucleus_2f | 912 | 25 | 2 |
| oocytes_trisopterus_states_5b | 898 | 32 | 2 |
| optical | 5620 | 62 | 10 |
| ozone | 2536 | 72 | 2 |
| page-blocks | 5445 | 10 | 4 |
| pendigits | 10992 | 16 | 10 |
| pima | 768 | 8 | 2 |
| planning | 182 | 12 | 2 |
| ringnorm | 7400 | 20 | 2 |
| seeds | 210 | 7 | 3 |
| semeion | 1593 | 256 | 10 |
| soybean | 362 | 35 | 4 |
| spambase | 4601 | 57 | 2 |
| spect | 265 | 22 | 2 |
| spectf | 267 | 44 | 2 |
| statlog-australian-credit | 690 | 14 | 2 |
| statlog-german-credit | 1000 | 24 | 2 |
| statlog-heart | 270 | 13 | 2 |
| statlog-image | 2310 | 18 | 7 |
| statlog-landsat | 6435 | 36 | 6 |
| statlog-shuttle | 57977 | 9 | 5 |
| statlog-vehicle | 846 | 18 | 4 |
| steel-plates | 1941 | 27 | 7 |
| synthetic-control | 600 | 60 | 6 |
| teaching | 102 | 5 | 2 |
| thyroid | 7200 | 21 | 3 |
| tic-tac-toe | 958 | 9 | 2 |
| titanic | 2201 | 3 | 2 |
| twonorm | 7400 | 20 | 2 |
| vertebral-column-2clases | 310 | 6 | 2 |
| vertebral-column-3clases | 310 | 6 | 3 |
| wall-following | 5456 | 24 | 4 |
| waveform | 5000 | 21 | 3 |
| waveform-noise | 5000 | 40 | 3 |
| wine | 130 | 13 | 2 |
| wine-quality-red | 1571 | 11 | 4 |
| wine-quality-white | 4873 | 11 | 5 |
| yeast | 1350 | 8 | 5 |

As the chosen datasets are not originally intended as streaming data, our experiment uses the concept that the separable characteristics of each class are woven throughout the features. Therefore some changes will be easier to detect than others, introducing variety in our test data. Even if the sample size is insufficient to detect changes in a given dataset, this does not compromise experimental integrity because every detector faces the same challenge. A detector which performs well on average has negotiated a diverse range of class separabilities.

Datasets with fewer than 1000 examples will be oversampled in this experiment, but we found no relationship between the oversampling percentage of a dataset and our results. Even if this were to hinder or benefit the task at hand, the challenge is the same for every detector.

**Algorithm 7:** Experimental procedure

```
for dataset in datasets do
    for i = 1,...,100 do
        Choose a random subset of the classes as P;
        if abrupt then
            Sample 500 examples as S_1 from P;
        else if gradual 100 then
            Sample 600 examples as S_1 from P;
        else
            Sample 800 examples as S_1 from P;
        end
        Sample 500 examples as S_2 from the remaining classes;
        Concatenate subsets into 'abrupt' and 'gradual' test data;
        for detector in detectors do
            Evaluate abrupt;
            Evaluate gradual 100;
            Evaluate gradual 300;
        end
    end
    Store average abrupt metrics;
    Store average gradual 100 metrics;
    Store average gradual 300 metrics;
end
```

We measure the following characteristics for each method, averaged over the 100 runs each, for abrupt and gradual change on each dataset:

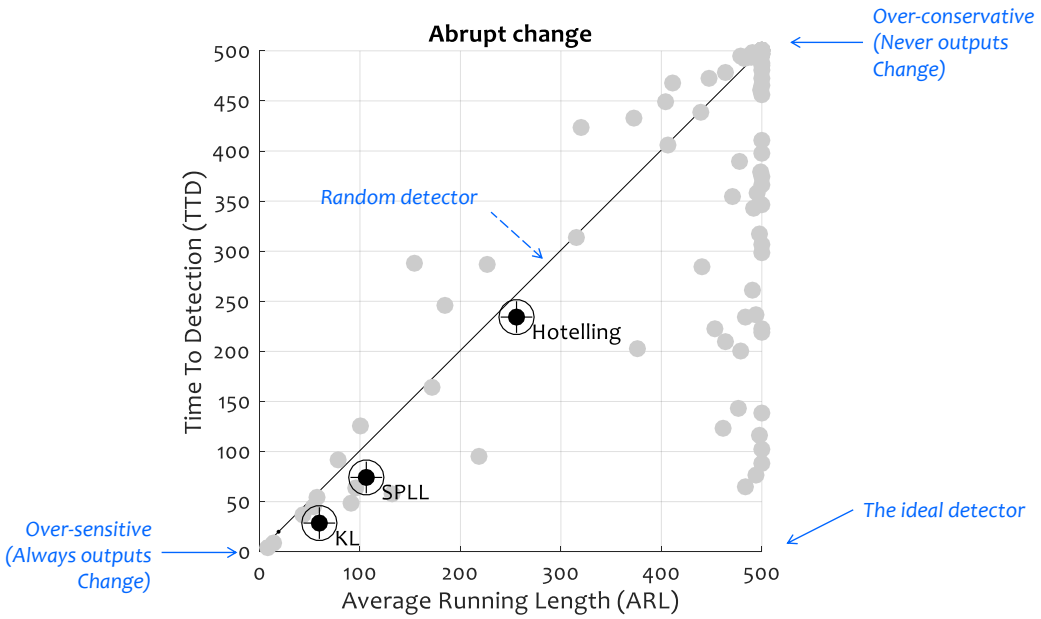



**Figure 5.3:** Scatterplot of the 88 detector methods in the space $(ARL, TTD)$ for the Abrupt-change part of the experiment. The three individual detectors are highlighted.

**ARL** Average Running Length: The average number of contiguous observations for which the detector did not signal change.

**TTD** Time To Detection: The average number of observations between a change occurring and the detector signalling.

**NFA** The percentage of runs for which the detector did not issue a false alarm.

**MDR** The percentage of runs for which the detector did not signal after a true change.

Based on these characteristics, a good method should maximise $ARL$ and $NFA$, and minimise $TTD$ and $MDR$.

Figure 5.3 is the archetype of our result figures. It plots $TTD$ versus $ARL$ for the detection methods. The grey dots correspond to ensemble methods, and the highlighted black dots correspond to the individual detectors (Hotelling, KL, and SPLL). The ideal detector will have $ARL = \infty$ (500 in our experiment, mean-

ing that no false detection has been made before the true change happened), and $TTD = 0$. This detector occupies the bottom right corner of the plot. Dots which are close to this corner are indicative of good detectors.

The two trivial detectors lie at the two ends of the diagonal plotted in the figure. A detector which always signals change has $ARL = 0$ and $TTD = 0$, while detector which never signals change has $ARL = 500$ and $TTD = 500$. A detector which signals change at random will have its corresponding point on the same diagonal. The exact position on the diagonal will depend on the probability of signalling a change (unrelated to actual change). Denote this probability by $p$. Then $ARL$ is the expectation of a random variable $X$ with a geometric distribution ($X$ is the number of Bernoulli trials needed to get one success, with probability of success $p$), that is $ARL = \frac{1-p}{p}$. The time to detection, $TTD$, amounts to the same quantity because it is also the expected number of trials to the first success, with the same probability of success $p$. Thus the diagonal $ARL = TTD$ is a baseline for comparing change detectors. A detector whose point lies above the diagonal is inadequate; it detects change when there is none, and fails to detect an existing change. We follow the same archetype for visualisation of the MDR/NFA space. We plot MDR against 1-NFA for these figures in order to maintain the same visual orientation for performance. Therefore the ideal detector in this space is also at point $(1,0)$, i.e., all changes were detected, and there were no false alarms.

### 5.4.2 A Case Study

In addition to the main experiment, we conducted a practical case study on a network intrusion detection dataset. We chose the popular KDD Cup 1999 intrusion detection dataset, which is available from the UCI Machine Learning Repository [112]. With a network intrusion dataset, the change context is more likely to be longer-lived change from one concept to another, which could be

either abrupt or gradual. The dataset consists of 4,900,000 examples and 42 features extracted from seven-weeks of TCP dump data from network traffic on a U.S. Air Force LAN. During the seven weeks, the network was deliberately peppered with attacks which fall into four main categories.

- Denial of Service (DOS): An attacker overwhelms computing resources in order to deny access to them.
- Remote to Login (R2L): Attempts at unauthorised access from a remote machine, such as guessing passwords.
- Unauthorized to Root (U2R): Unauthorised access to local superuser privileges, through a buffer overflow attack, for example.
- Probing: Surveillance and investigation of weaknesses, such as port scanning.

Of these categories, there are 24 specific attack concepts, or 24 classes. This dataset is most commonly interpreted as a classification task. Viewed as such, it offers some interesting challenges in its deficiencies. For example, there is 75% and 78% redundancy in duplicated records across the training and testing set respectively [165]. This can serve to bias learning algorithms toward frequent records. It also has very imbalanced classes, with the *smurf* and *neptune* DoS attacks constituting 71% of the data points; more than the 'normal' class. We offer an interpretation of this data as a change detection task.

We evaluated the methods on the testing dataset. Since the data is sequential, we pass observations in order, one-by-one to each of the detectors. The objective in our experiment was for the detectors to identify the concept boundaries. When the concept changes from one class to another, we record whether this change point was detected. With this scheme, if we are experiencing a long-lived concept such as a denial of service attack then after a sufficient number of examples of the same concept, we would expect the change detection methods to also detect the changepoint back to the normal class, or to another attack.

One challenge for the change detectors in this interpretation is that some concepts may be very short-lived, that is, the change in the distribution is a 'blip', involving only a few observations, after which the distribution reverts back to the original one. Such blips may be too short to allow for detection by any method which is not looking for isolated outliers.

## 5.5 Results and Discussion

Figure 5.4 visualises the ARL/TTD space for abrupt and gradual change type by the categories in the taxonomy by Gama et al. [55]. Each plot contains all 96 points (one for each data set) of the 88 change detection methods. Empirically, there is a clear and visible distinction between the methods in the Control Chart category, which performed, on average, worse than chance, and those in the other two categories. Table 5.6 confirms that Sequential Analysis and Monitoring distribution methods were much more likely to exhibit a high ARL. Furthermore, distribution monitoring methods exhibited considerably lower TTD whilst being competitive on ARL with Sequential Analysis methods. Observe the two distinct clusters in the ARL/TTD space for this category (the triangle marker), and the relative sparsity in-between. We suspect that this is the effect of gradual change on the TTD statistic. This is visible between the figures, where we observe that, in the gradual change experiment, those methods with a high ARL and low TTD struggle to better a TTD of 50, which is the halfway point of introducing the gradual change. Those methods with an already low ARL do not move significantly in the TTD axis between experiments. We suspect that this is because a low ARL implies an over-eager detector, which in turn increases the probability that a valid detection is due to random chance rather than a response to observation of the data.

The bottom two charts in Figure 5.4 visualise the NFA/MDR space for the aforementioned categories. Interestingly, we see a very similar effect for

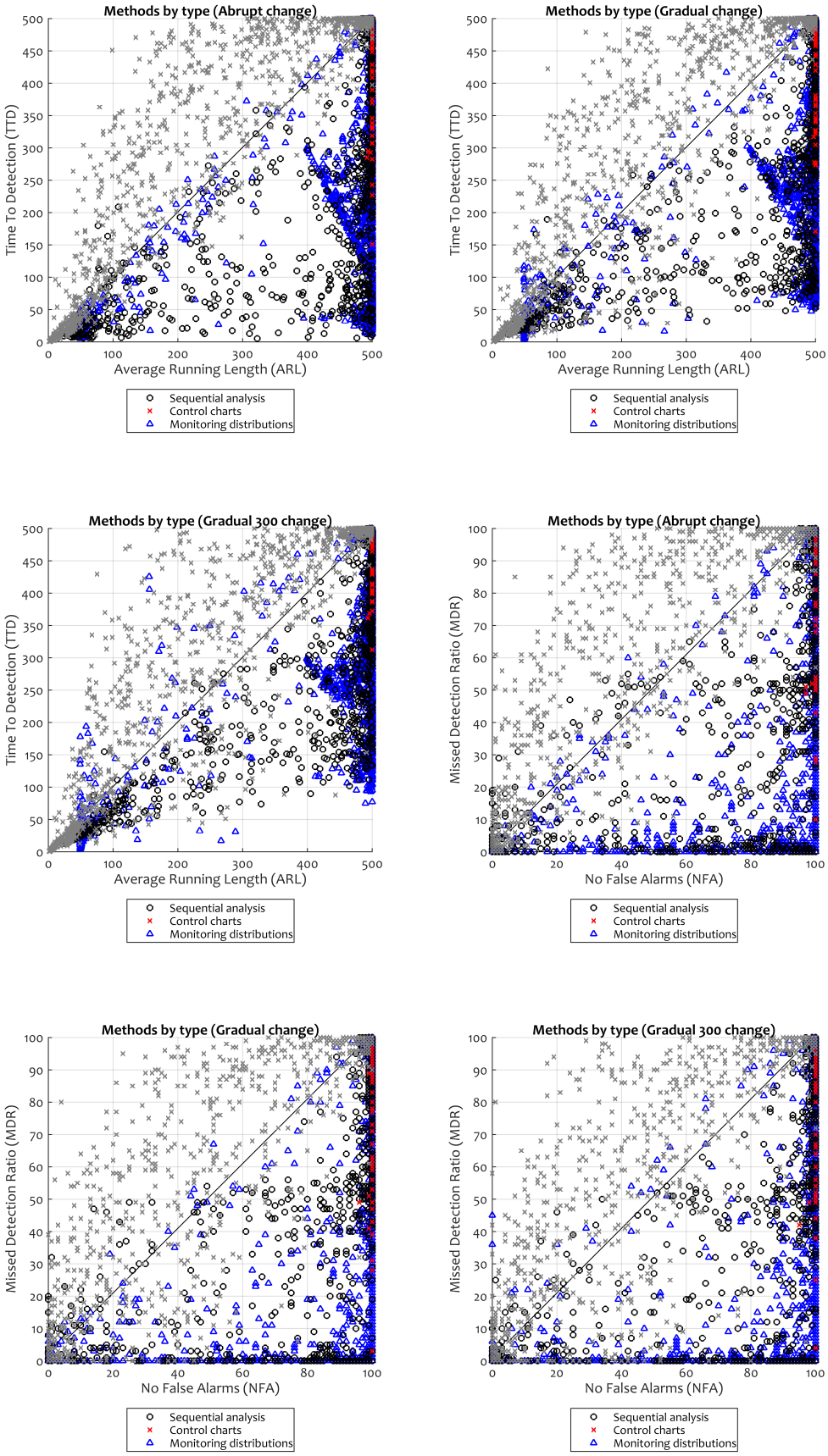



**Figure 5.4:** The three categories of detector, visualised in the ARL/TTD space for the abrupt, gradual 100 and gradual 300 change experiments, respectively. Data points for methods whose assumptions were violated are greyed out, but retain their category marker.

**Table 5.6:** The mean and standard deviation of the metrics for each category.

| Method | ARL | | TTD | | NFA | | MDR | |
|---|---|---|---|---|---|---|---|---|
| | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ |
| Sequential analysis | 433.49 | 134.28 | 323.02 | 187.07 | 80.21 | 34.26 | 59.14 | 42.26 |
| Control charts | 499.93 | 0.68 | 486.67 | 46.14 | 99.97 | 0.28 | 96.46 | 12.02 |
| Monitoring distributions | 435.16 | 145.36 | 219.77 | 176.18 | 81.07 | 34.73 | 29.75 | 38.82 |

**Table 5.7:** The top 20 performers in the main experiment and the case study. The methods are ranked in the listed 2D spaces by minimum Euclidean distance to their respective ideal points, $(500,0)$, $(1,0)$, $(7684.09,0)$ and $(0,0)$. The ranks of the multivariate detectors and multivariate ensemble are also shown if they were not represented in the top 20.

| | Main Experiment Averages | | | | | | Case Study – KDD Cup 1999 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # | Detector | ARL | TTD | Detector | NFA | MDR | Detector | ARL | TTD | Detector | FPR | MDR |
| 1 | SEED-1 | 484.18 | 113.07 | SEED-5 | 0.96 | 0.05 | ADWIN-20 | 10578.05 | 327.71 | HDDMA-1 | 0.14 | 0.07 |
| 2 | SEED-5 | 494.00 | 130.66 | ADWIN-1 | 1.00 | 0.06 | SEED-20 | 10900.19 | 648.86 | CUSUM1-1 | 0.03 | 0.26 |
| 3 | ADWIN-1 | 499.67 | 148.46 | ADWIN-5 | 1.00 | 0.08 | SEQ1-5 | 10930.04 | 578.64 | CUSUM1-5 | 0.01 | 0.31 |
| 4 | CUSUM2-1 | 462.10 | 160.48 | SEED-1 | 0.91 | 0.03 | CUSUM1-30 | 11153.81 | 1179.54 | HDDMA-5 | 0.03 | 0.31 |
| 5 | ADWIN-5 | 499.91 | 165.00 | SEED-10 | 0.98 | 0.14 | SEQ1-1 | 4291.67 | 180.79 | PH-1 | 0.01 | 0.32 |
| 6 | SEED-10 | 497.54 | 172.90 | ADWIN-10 | 1.00 | 0.15 | CUSUM2-5 | 3462.90 | 1281.90 | CUSUM2-1 | 0.01 | 0.32 |
| 7 | PH-1 | 477.96 | 187.96 | SEQ1-20 | 0.94 | 0.18 | ADWIN-10 | 3094.09 | 85.84 | HDDMW-1 | 0.32 | 0.08 |
| 8 | ADWIN-10 | 499.94 | 197.93 | PH-1 | 0.86 | 0.13 | SEED-10 | 2828.08 | 74.83 | GEOMA-1 | 0.01 | 0.33 |
| 9 | SEQ1-5 | 463.90 | 242.38 | SEQ1-10 | 0.79 | 0.06 | DDM-5 | 13724.94 | 1974.68 | MV-50 | 0.02 | 0.36 |
| 10 | SEQ1-10 | 478.91 | 247.84 | CUSUM2-1 | 0.75 | 0.10 | HDDMA-10 | 2605.21 | 3357.99 | Hotelling | 0.02 | 0.36 |
| 11 | SEQ1-1 | 453.59 | 248.14 | ADWIN-20 | 1.00 | 0.33 | CUSUM1-20 | 2646.27 | 3734.60 | EDDM-1 | 0.00 | 0.37 |
| 12 | CUSUM1-20 | 374.97 | 228.50 | SEQ1-5 | 0.64 | 0.03 | ADWIN-5 | 741.15 | 48.96 | CUSUM1-10 | 0.00 | 0.37 |
| 13 | CUSUM2-5 | 484.61 | 264.66 | SEQ1-30 | 0.98 | 0.37 | SEED-5 | 682.78 | 48.51 | KL | 0.01 | 0.37 |
| 14 | ADWIN-20 | 499.99 | 268.90 | CUSUM2-5 | 0.89 | 0.37 | EDDM-1 | 563.22 | 39.67 | SPLL | 0.02 | 0.37 |
| 15 | SEED-20 | 499.52 | 274.41 | SEED-20 | 1.00 | 0.41 | DDM-1 | 441.54 | 1494.63 | EWMA-1 | 0.00 | 0.39 |
| 16 | PH-5 | 491.43 | 293.04 | PH-5 | 0.94 | 0.41 | EWMA-1 | 541.86 | 2015.76 | DDM-1 | 0.00 | 0.39 |
| 17 | SEQ1-20 | 494.19 | 294.23 | SEQ1-1 | 0.54 | 0.03 | SEED-1 | 229.07 | 32.73 | ADWIN-1 | 0.01 | 0.40 |
| 18 | CUSUM1-10 | 219.09 | 114.13 | ADWIN-30 | 1.00 | 0.50 | ADWIN-1 | 187.46 | 24.95 | SEED-1 | 0.00 | 0.41 |
| 19 | CUSUM1-30 | 439.02 | 308.69 | CUSUM1-20 | 0.59 | 0.34 | PH-1 | 113.59 | 15.71 | SEED-5 | 0.00 | 0.43 |
| 20 | ADWIN-30 | 499.99 | 328.74 | CUSUM1-30 | 0.80 | 0.52 | GEOMA-1 | 108.70 | 19.54 | ADWIN-5 | 0.00 | 0.44 |
| | | ⋮ | ⋮ | | ⋮ | ⋮ | | ⋮ | ⋮ | | | |
| # | Detector | ARL | TTD | # | Detector | NFA | MDR | # | Detector | ARL | TTD | |
| 21 | Hotelling | 499.95 | 432.97 | 30 | Hotelling | 0.01 | 0.01 | 23 | KL | $\infty$ | $\infty$ | |
| 34 | SPLL | 484.61 | 264.66 | 47 | SPLL | 0.04 | 0.04 | 25 | SPLL | $\infty$ | $\infty$ | |
| 39 | MV-50 | 499.88 | 497.29 | 54 | MV-50 | 1.00 | 1.00 | 26 | MV-50 | 541.86 | 2015.76 | |
| 47 | KL | 57.02 | 56.73 | 68 | KL | 0.86 | 0.13 | 27 | Hotelling | 9137.79 | 8020.57 | |

control chart methods. To understand why the performance of this category is so poor, we must consider the assumptions of the detectors. This experiment presented the data points directly to the change detection methods in the ensemble. Specifically, this category contains EDDM, HDDM$_A$ and HDDM$_W$, all of which share a common ancestor in DDM. Whilst the MOA interface for change detectors accepts 64 bit floating point numbers, these methods were not intended for continuous-valued data. DDM assumes the Binomial distribution. It also assumes that the monitored value (e.g., error rate of a classifier) will decrease while the underlying distribution is stationary. The derived methods also share this assumption, which is fundamentally violated by the nature of the data presented to them in this experiment.

The top 20 performers averaged over abrupt and gradual change are summarised in the left half of Table 5.7. The performers were ranked by minimum Euclidean distance to the ideal points in the ARL/TTD and NFA/MDR spaces, $(500,0)$ and $(1,0)$.

The results for each individual method are summarised in the ARL/TTD space in Figure 5.5, and in the NFA/MDR space in Figure 5.6. In the ARL/TTD space, the SEED and ADWIN detectors were the best performers, with Page Hinkley, CUSUM2 and SEQ1 showing competitive patterns. The multivariate detectors exhibited a large standard deviation, suggesting that their performance is related to the suitability of the data – an observation which would appear to lend further credence to the conclusions of Allipi et al. [4], as well as our own hypothesis. In the NFA/MDR space, the winners are the low quorum ensembles of the SEED and ADWIN detectors. In fact, all the ensembles outside of the control chart category performed favourably compared to the multivariate detectors. Observing the curves of the SEED, ADWIN, Page Hinkley, CUSUM1, CUSUM2 and SEQ1 detectors across both sets of metrics, we see that the ideal agreement threshold is a case-by-case problem. The ADWIN ensemble improves almost linearly as we reduce the agreement threshold, suggesting that the optimum

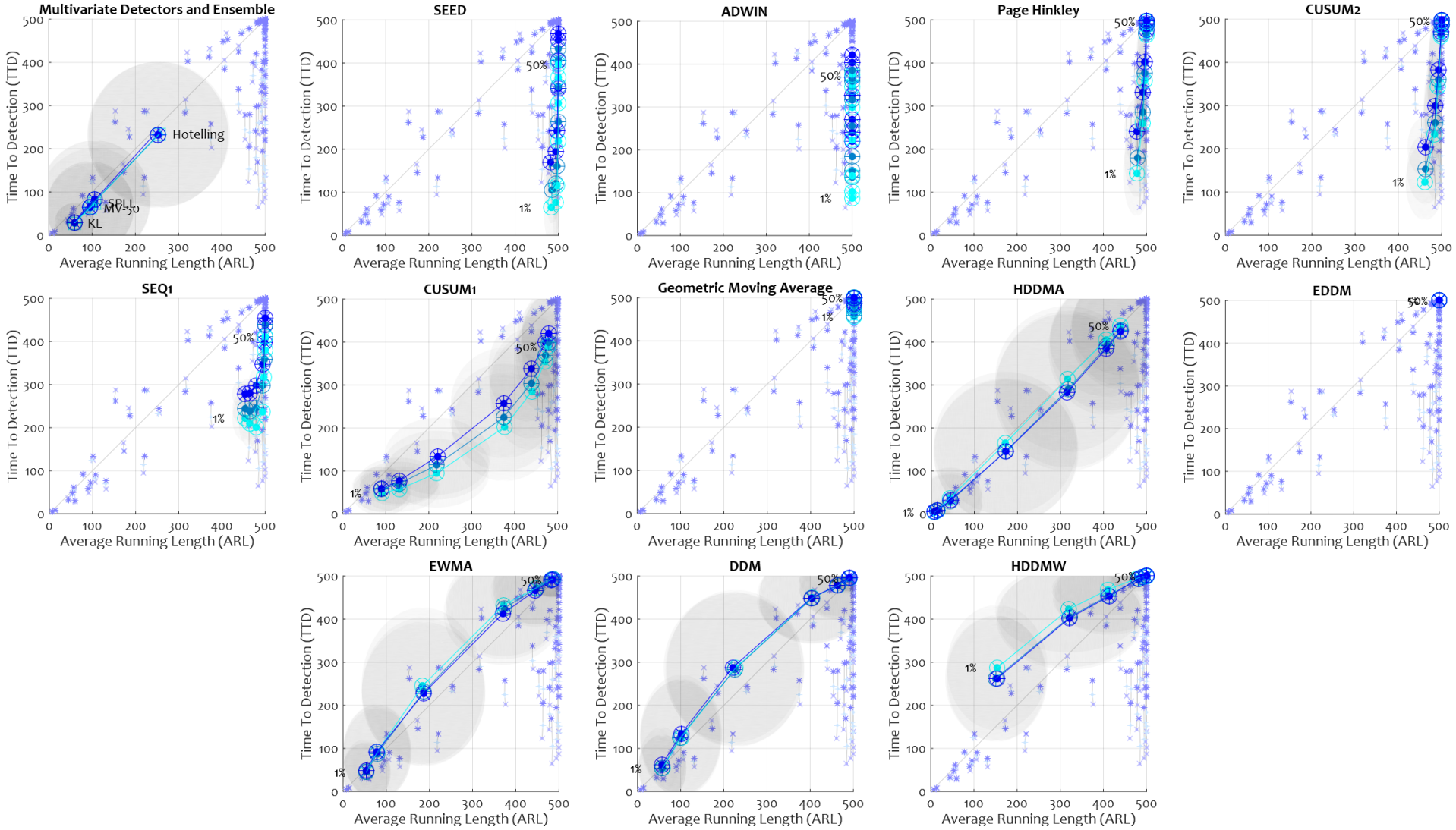



**Figure 5.5:** Change detection methods in the space spanned by ARL and TTD for the main experiment. Each method has been examined with different agreement thresholds. Each plot contains 88 gradual and 88 abrupt detector points, averaged across the 96 data sets – gradual 300 as a blue x (darkest), linked to the paired gradual 100 result as a purple + and the abrupt result as a cyan ∗ (lightest). Each detector's points are highlighted, again in blue, purple and cyan for gradual 300, gradual and abrupt change type, respectively. The shaded ellipses around the mean detector results are the standard deviations across the 96 datasets. The ideal point is $(500,0)$.

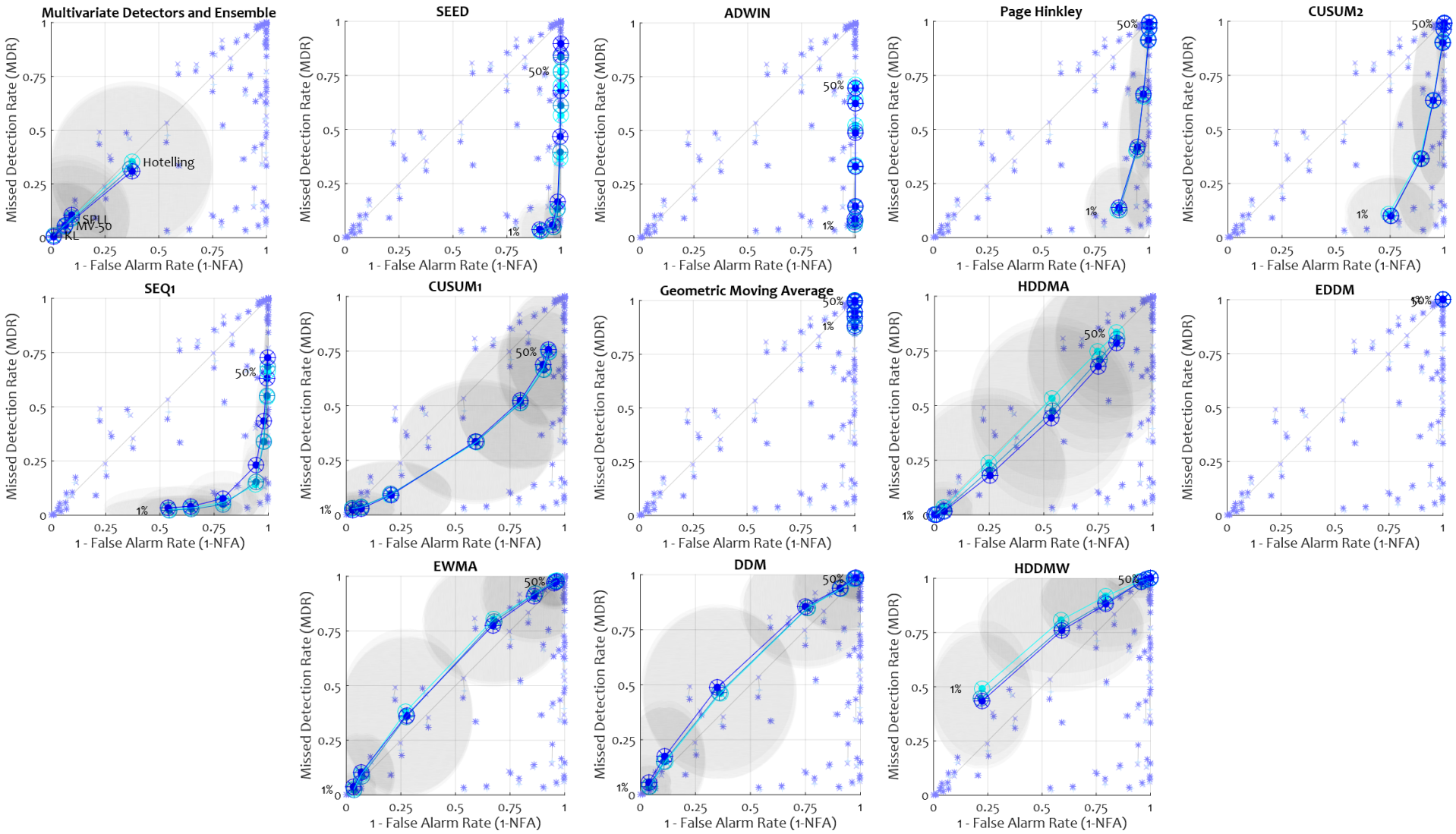



**Figure 5.6:** Change detection methods in the space spanned by NFA and MDR for the main experiment. Each method has been examined with different agreement thresholds. Each plot contains 88 gradual and 88 abrupt detector points, averaged across the 96 data sets – gradual 300 as a blue x (darkest), linked to the paired gradual 100 result as a purple + and the abrupt result as a cyan ∗ (lightest). Each detector's points are highlighted, again in blue, purple and cyan for gradual 300, gradual 100 and abrupt change type, respectively. The shaded ellipses around the mean detector results are the standard deviations across the 96 datasets. The ideal point is $(1,0)$.

scheme is one whereby any member of the ensemble has absolute authority to signal a change. With other ensembles such as SEED and SEQ1, the 1% threshold is beyond the optimal, with the best ensembles having thresholds of 5% and 10%, respectively in the NFA/MDR space. It appears that the optimal choice of threshold differs slightly between the ARL/TTD space and the NFA/MDR space. There is a clear and expected effect between abrupt and gradual change on the ARL/TTD space mostly in the TTD axis, with TTD being marginally lower for abrupt changes in those detectors whose assumptions are not violated.

We note that the ideal agreement threshold varies between detectors. The curves in Figures 5.5 and 5.6 can be used to pick a suitable threshold for each of the successful detectors. Taking ADWIN for example, the lack of movement on the false alarm rate relative to the threshold changes suggests that an ensemble might be close to optimal if any member is given absolute authority for signalling. As a counter example, the SEQ1 ensembles seem to have an optimal agreement threshold of between 10% and 20%.

Bearing in mind the works of Allipi et al. [4] and Evangelista et al. [42], we were interested in observing the effects of data dimensionality on the missed detection rate. Scatterplots of average missed detection rate against dataset dimensionality, for each category of ensemble and for the multivariate detectors, are presented in Figure 5.7. The scatter patterns suggest that changes in higher-dimensional spaces are more likely to be missed.

### 5.5.1 The Case Study

The right half of Table 5.7 summarises the top 20 performers on the case study data. As this experiment was a single run, we present the false positive rate as FPR, instead of the NFA measure. The methods were ranked by the minimum Euclidean distance to the ideal points $(7864.09,0)$ and $(0,0)$ for the ARL/TTD

and FPR/MDR spaces respectively. The ideal ARL of $7864.09$ was calculated by observing the ARL of a perfect, 'cheating' detector, which signalled immediately for all changepoints and recorded no false positives. We see a familiar pattern in the ARL/TTD space, with the SEED, ADWIN and CUSUM-based methods well represented within the top 20. In the FPR/MDR space, the winners are primarily low-threshold ensembles. We note that 8 methods; ADWIN-1, ADWIN-5, SEED-1, SEED-5, EDDM-1, PH-1, GEOMA-1 and EWMA-1 are represented in the top 20 in both spaces. We also observe that the top ranked ensembles across the two spaces here differed modestly from the top performers in the main experiment with the simulated abrupt and gradual changes. The improvement in performance of control chart-based methods may be due to the incidence of a number of contextually important binary features in this dataset. The best performing multivariate detectors were ranked 23rd and 9th in the two spaces respectively. Apart from the high false positive rates of HDDM$_W$–1 and HDDM$_A$–1, the ensembles were competitive or better than the multivariate detectors on TTD and MDR, and generally exhibited less false positives. The dominance of the low-threshold ensembles mirrors their success in the previous experiment, and suggests that between 1% and 5% agreement is a sensible starting point when employing this scheme, across a range of different detectors.

## 5.6 Conclusions

The results of the experiment and the case study demonstrate the viability of ensemble combination of univariate change detectors over multivariate data. Over 96 datasets, ensemble methods frequently outperformed multivariate detectors in all metrics, especially at low agreement thresholds. The multivariate detectors did not even feature in the top 20 overall performers in either space, as seen in Table 5.7. This would appear to tally with the conclusions of Evangelista et al. [42] and lends support to hypothesis (5). The SEED and

ADWIN detectors appear to be the best suited to ensemble combination in this manner. Given that the SEQ1 algorithm is an ADWIN-derivative, we would expect it to exhibit a similar performance. We see that it does exhibit very similar performance to the ADWIN ensembles in terms of missed detections, but it signals far more eagerly for a higher rate of false alarms. This may be a reflection, as noted in Section 2.6.2, of the authors' choice of the Bernstein bound over the more conservative Hoeffding bound to set the threshold.

The main experiment described in Section 5.4.1 is sample based, so the results are consistent with hypothesis (1). However, it is not safe to assume that data within the case study are *i.i.d*, as Tartatovsky et al. make clear [164]. Therefore the case study acts as a foil to the main experiment, and the strong performance of the ensembles in the case study suggests an encouraging versatility.

Those detectors which make strong assumptions on the basis that they are monitoring the error stream of an attached learner were unsurprisingly poor when applied to raw data in this scheme. This accounts for the worse-than-chance performances of the HDDM$_A$, HDDM$_W$, EDDM, DDM and EWMA methods.

We observed empirically that all categories of detectors exhibited a positive relationship between missed detections and dataset dimensionality, as Allipi et al. [4] suggest, albeit to varying degrees. Evangelista et al. [42] also state that unsupervised learning in subspaces of the data is a means to address the curse of dimensionality. This is not strongly reflected in Figure 5.7, with the multivariate detectors appearing to exhibit the weakest relationship of missed detections with dimensionality. However, the ensembles had a much wider spread of results, and the better ensembles considerably outperformed the multivariate detectors.

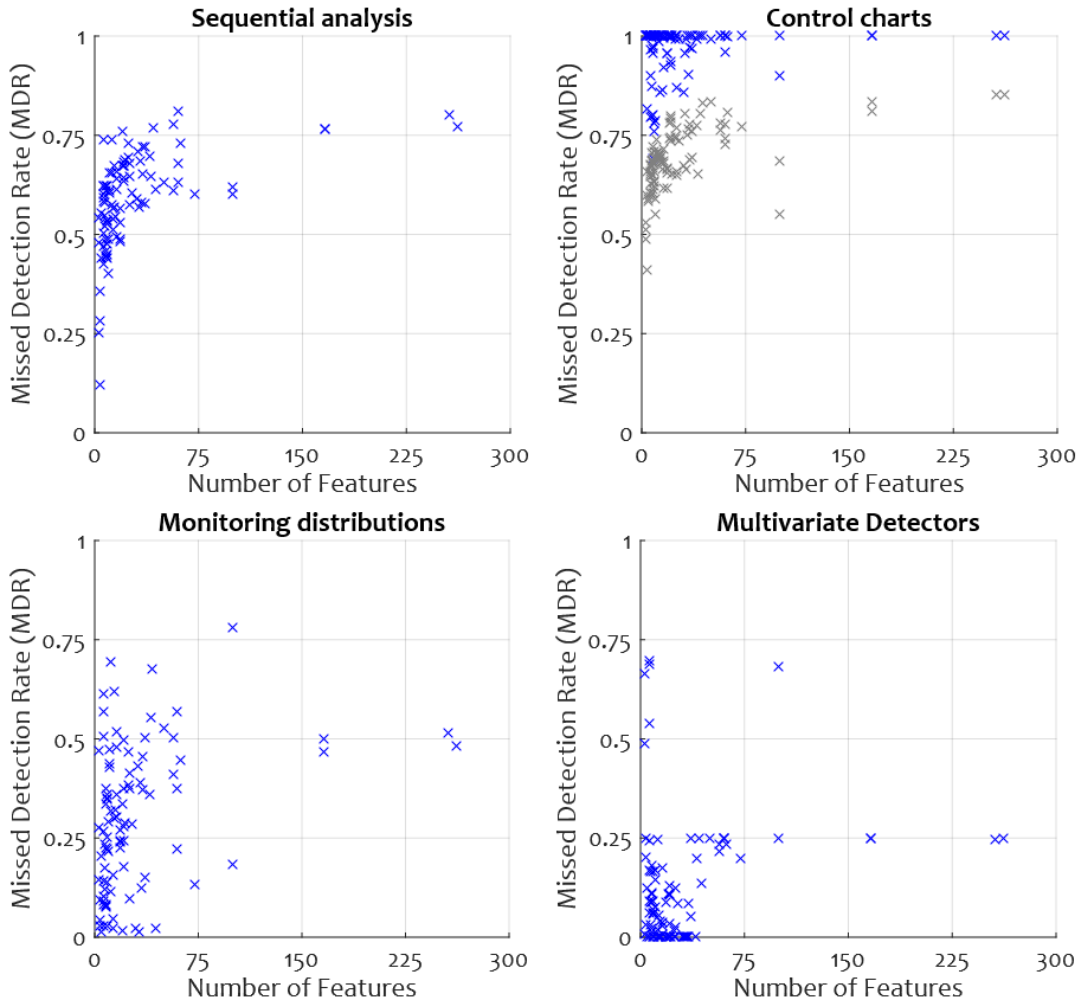



**Figure 5.7:** Scatter plots of dataset dimensionality against average missed detection rate for the 96 datasets. The plots are arranged by the category of the detectors. Data points from detectors with violated assumptions are greyed out.

The effect of differing persistence of change (abrupt, gradual 100, gradual 300) was investigated to some extent, with the effect visible in Figure 5.4. The tunability of these ensembles with regard to outlier, novelty and change detection is governed by the forgetting mechanisms of the univariate detectors which comprise them.

## Acknowledgment

This work was done under project RPG-2015-188 funded by The Leverhulme Trust, UK; Spanish Ministry of Economy and Competitiveness through project

TIN 2015-67534-P and the Spanish Ministry of Education, Culture and Sport through Mobility Grant PRX16/00495. The 96 datasets were originally curated for use in the work of Fernández-Delgado et al. [45] and accessed from the personal web page of the author[4]. The KDD Cup 1999 dataset used in the case study was accessed from the UCI Machine Learning Repository [112].

[4] http://persoal.citius.usc.es/manuel.fernandez.delgado/papers/jmlr/

# Chapter 6

# Unsupervised Endogenous Blink Detection from Streaming Video

## 6.1 Introduction

Eye blink detection from video is a small but active cross-disciplinary field of study contributing to a range of applications such as Human-Computer Interaction (HCI) [29, 131, 93, 92], drowsy driver identification [75, 106, 69], liveness detection for spoof protection [138, 182], deception detection [49] and epileptic seizure detection [152].

This problem is a good fit for unsupervised multivariate change detection for the following reasons. Firstly, it involves identifying change points in high-dimensional streaming data, where ground truth is not available. Secondly, the process of feature extraction from video may be very complex and many works include bespoke static thresholds rather than adaptive thresholds.

In this chapter, a relatively simple feature extraction process is applied to streaming video of six people's faces. This results in a fixed-size multivariate feature space in which changes can be observed over time. The video data used is an original dataset curated for this chapter. The objectives of these experiments are twofold. Firstly, to test the applicability of contributions from previous chapters. The multivariate detectors are assessed first on this dataset

as a baseline. The techniques presented in Chapters 3 and 4 are designed to improve the performance of multivariate change detection, while the technique in Chapter 5 offers an alternative to multivariate detectors. Secondly, blink detection systems involve large and complex pipelines of operations for feature extraction. The decision of what is or is not a blink is usually taken by a bespoke threshold on some aspect of these extracted features. Established change detection methods may be an alternative in lieu of bespoke thresholds.

The chapter is organised as follows. Section 6.2 reviews related work on the problem of blink detection. Section 6.3 presents an overview of the dataset, how it was collected, how it was labelled and its statistics. Section 6.4 details the feature extraction process used. The baseline experiment is presented in Section 6.5, and the experiments for the techniques in chapters 3, 4 and 5 are in Sections 6.8, 6.6 and 6.7 respectively.

## 6.2 Related Work

There are two kinds of data used for blink detection – signals from a brain-computer interface like an electroencephalogram (EEG), and facial video processed with computer vision techniques. This chapter is concerned with vision-based approaches. Two types[1] of blinks are usually discussed in blink detection literature; voluntary and endogenous (involuntary) [93]. Work on the former places a high degree of importance on blink *length*, this typically being some form of control mechanism for HCI. This chapter is concerned only with detection of endogenous blinks from video footage.

Vision-based techniques for blink detection are divided between active and passive [93]. The former uses special equipment to illuminate the eye in a

[1] Psychology literature on the subject such as Stern et al. [159] also discuss *reflex blinks* and *non-blink closures* as examples of eye closures outside these two categories.

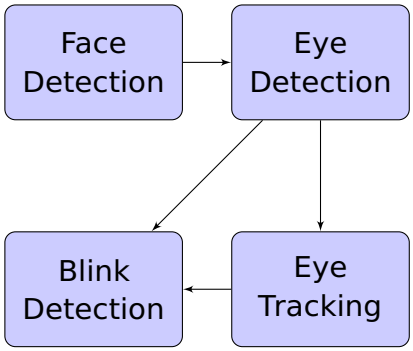



**Figure 6.1:** Blink detection is a multi-phase process, drawing on many computer vision techniques.

particular way to facilitate tracking, for example with infra-red [167]. Passive approaches use any sequence of images in normal visible light. The reviewed works in this section will be mostly from this category. However, even within this category, not all data is equal. Some approaches are tested on datasets captured from head-mounted cameras [111], which provide a reliable image of the eye with varying degrees of need for face and eye detection. However, most assume images contain a whole head [29, 106, 131, 93, 7, 35, 31], from which the face and then the eyes must first be isolated from the rest of the image. Video from a static camera such as a webcam is more difficult than a head-mounted camera, because head movements relative to the camera need to be accounted for. The advantage of static camera data and the reason for its popularity within the field is that no specialist equipment is required beyond a video camera.

As depicted in Figure 6.1, blink detection comes after an initial process of face or eye detection, or both. Once eyes have been detected, it may be less computationally expensive to track the eyes than to re-detect them in subsequent frames. Two popular algorithms for motion tracking among blink detection systems are Lukas-Kanade [116] (LK) and Starburst [111]. Blink detection systems are often very tightly related to approaches for eye detection or tracking. Common eye detection methods include Haar-feature classification [169, 25, 32, 110], template matching [69, 93, 7] and colour models [91, 30, 31].

Many systems compute a difference image between subsequent video frames to detect areas of motion, isolating the motion of the eyelids. These could be called *motion* based methods, in contrast to methods which test the openness of the eye. This motion based approach is the same frequently applied for detecting changes in satellite imagery, such as the LANDSAT data [153, 154]. Chau and Betke [29] is an example of such a system. They use streaming video from a USB webcam along with a sophisticated feature extraction process which involves frame-to-frame image differencing, thresholding and opening, eventually creating an eye template which is tracked throughout the video. Crowley and Berard [31] use a frame-to-frame difference image to identify blinks, while also using histogram normalisation to attempt to identify skin within the image. Morris et al. [131] also use a frame-to-frame motion based approach for face detection, identifying the eyes and tracking them using the LK algorithm. They detect blinks by thresholding a variance map of the eye image, signalling a blink if there is a high proportion of remaining pixels relative to the size of the eye bounding box. Lalonde et al. [106] use profile analysis to detect the eyes, and then use the GPU to compute the Scale-Invariant feature transform (SIFT) features of the eyes for tracking. To detect blinks, they compute the optical flow direction from a filtered frame-to-frame difference image and declare a blink if the optical flow direction is downwards. Bhaksar et al. [13] also use frame differencing to compute optical flow, tracking the eyes using LK and thresholding the direction and magnitude of the flow to signal blinks. Fogelton and Benesova [47] compute motion vectors for the eyes and observe downward motion relative to a threshold based on standard devation.

Although less common, there are a number of systems which use an eye openness test. Królak and Strumiłło [91] use a model based on skin colour for eye detection and blink detection. Suzuki et al. [161] use a neural network to detect the face and perform eyelid detection based on analysis of the shadows. The gap between the eyelids is monitored for blink detection. Chen et al. [30] use the Starburst algorithm [111] for eye detection and tracking. They perform

an image segmentation process designed to isolate the iris, resulting in a binary mask. The aspect ratio of the resultant iris mask is used to identify whether the eye is open or closed. Lee et al. [110] exploit a difference in the ratio of black pixels between successive closed and open eye images. The cumulative sum of the difference over successive frames is fed into a pre-trained SVM to classify the eye as open or closed.

Another approach attempts to fit active appearance models (AAM) to the eye, the same technique used by the Kinect face tracking SDK as discussed in Chapter 4. Bacivarov et al. [7] build a point-based appearance model for open and closed eyes, map it to the face and assess the point positions.

Danisman et al. [32] use a Haar cascade classifier and a neural network to locate the face and the pupils. Taking into account the angle of rotation of the head, they analyse the horizontal symmetry of the eye region to establish openness.

This chapter aims to establish a consistent multivariate lower-dimensional representation of the video in which to apply change detection techniques to this problem. Blinks in motion-based systems are characterised by a pair of 'blip' changes, for the downward and upward motion of the eyelid. This places such data closer to the domain of anomaly detection than change detection. Instead, a 'recurring concepts' problem is preferred, which is provided by an eye-openness model. The system described in this chapter uses an a Haar cascade classifier for eye detection and an eye openness model based on colour for feature extraction. The features from the eye openness model are monitored with change detectors for blink detection.

**Figure 6.2:** Sample eye bounding box crops from the three label states.

## 6.3 Data Collection

The dataset used in this chapter consists of video footage of the faces of 6 people as they watch a nature documentary whilst sat at a computer. The data is organised around 6 volunteer subjects, each providing 5 minutes of footage, recorded at 1280x720 and at approximately 30 frames per second from a standard commercial HD webcam. The subjects were asked to sit approximately 70cm from the webcam, and watch a nature documentary. The resulting footage was trimmed down to exactly 5 minutes per subject, removing the beginning of the video so that the subject is in a comfortable position when the labelled data starts. The dataset consists of 53595 frames of video, of which 50116 were labelled 'Open', 2059 were 'Transition' and 1419 were 'Closed'. Taking the definition of a blink event to be a contiguous sequence of frames in a state other than 'Open', there were 432 blink events. The dataset is made publically available[2].

**Table 6.1:** Labels, their implied states and meanings.

| State | Label | Description |
|---|---|---|
| OPEN | 0 | The eyes are open such that the entire pupil is visible |
| TRANSITION | 1 | In at least one eye, the pupil is partially or totally occluded by the eyelid, but part of the iris or white of the eye remains visible. |
| CLOSED | 2 | In at least one eye, no iris, pupil or white of the eye is visible. |

**Table 6.2:** The ideal average run length (ARL) in frames, total blinks and blinks per minute calculated for each subject.

| Subject | $ARL_I$ | Blinks | Blink Rate | Blink Duration |
|---|---|---|---|---|
| 1 | 80.76 | 55 | 11.00 | 329.09 |
| 2 | 62.63 | 71 | 14.20 | 268.54 |
| 3 | 95.08 | 46 | 9.20 | 287.68 |
| 4 | 275.03 | 15 | 3.00 | 244.44 |
| 5 | 44.96 | 99 | 19.80 | 216.50 |
| 6 | 30.57 | 146 | 29.20 | 276.94 |
| $\mu$ | 98.17 | 72.00 | 14.40 | 270.53 |
| $\sigma$ | 89.73 | 45.65 | 9.1302 | 38.39 |

### 6.3.1 Labelling

Every frame of each 5 minute segment was labelled by eye using a custom labelling tool developed for the project in Java, and using OpenCV to perform video I/O. This tool is open source and available online[3]. There are three possible states into which the frames are divided, described in detail in Table 6.1. This approach was preferred over binary labelling because there are often ambiguous frames for which even a human would have difficulty classifying as 'Open' or 'Closed', as demonstrated in Figure 6.2. If a binary classification is desired, the data may be interpreted optimistically (all transition states count as closed) or pessimistically (all transition states count as open).

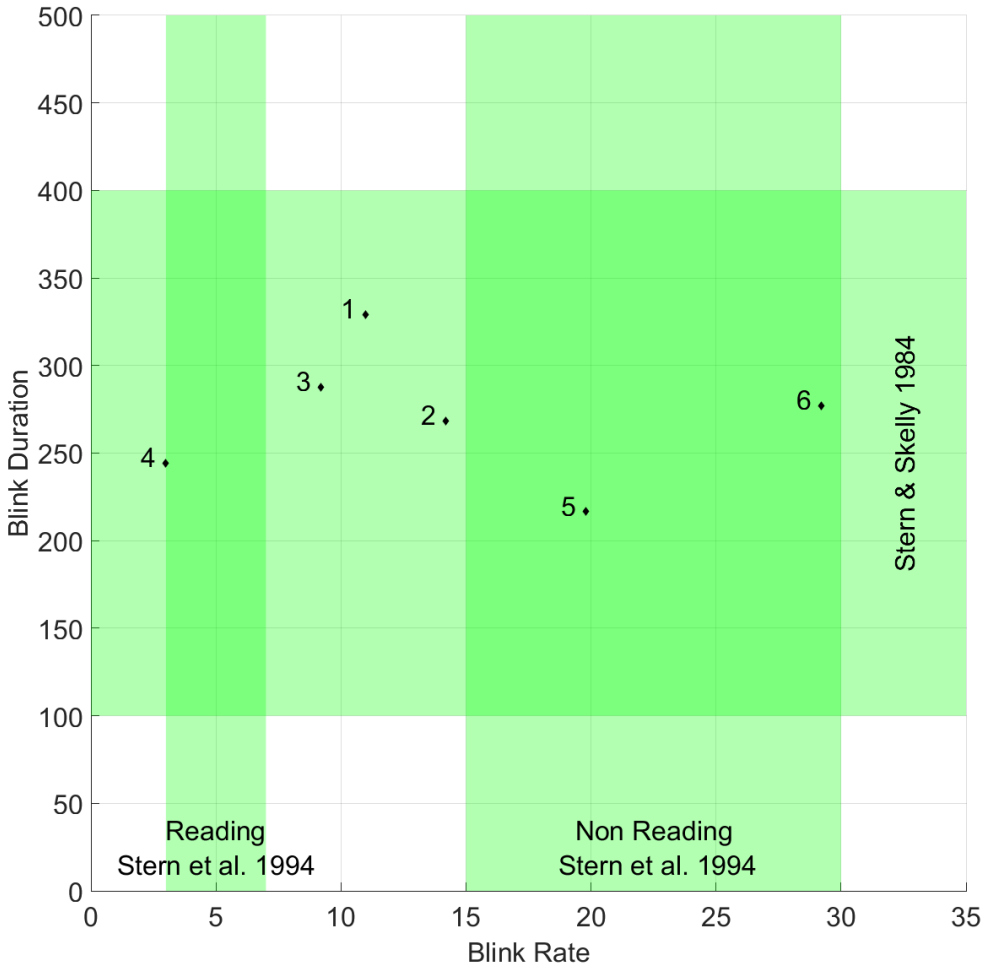



**Figure 6.3:** Mean blink rate and mean blink duration for each of the six subjects, compared to expectations from two review publications from psychology literature.

### 6.3.2 Data Discussion

In Figure 6.3 we see for all videos the mean blink rate (in blinks per minute) and mean blink duration (in milliseconds). The body of psychological literature concerned with the analysis of human blink rate and duration is helpful when considering the problem domain. Stern and Skelly [159] note that a typical endogenous, or involuntary eye blink lasts for between 100 and 400 milliseconds. Stern et al. [158] reported an average blink rate of between 3-7 blinks per minute while reading, and 15-30 blinks per minute while not reading. These findings correspond to the shaded areas in Figure 6.3. The distribution of the videos demonstrates at least anecdotally that the data lie within the reasonably

[2] https://faithfull.me/blinks-dataset/
[3] https://github.com/wfaithfull/videotag

expected range for this problem. However, Benedetto et al. [11] demonstrate that blink rate and blink duration may fluctuate due to a number of external factors, such as task difficulty and time on task. Table 6.2 lists the ideal Average Run Length (ARL), number of blinks, blink rate and blink duration.The ideal ARL is the average number of frames between blinks when the video is run in sequence, and it represents how often a perfect detector should signal.

The statistics highlight the scale of the problem. A significant diversity in ideal ARL, number of blinks and blink rate is clearly visible in both Figure 6.3 and Table 6.2. The same change detector needs to work for subject 4, who blinked only 15 times in 5 minutes and with subject 6, who blinked 146 times in the same 5 minutes. This suggests that the problem is not well suited to change detectors which are tuned to a desired ARL such as the EWMA charts of Ross et al. [144].

The subjects are anecdotally numbered in order of perceived difficulty. We expect that subjects 1 and 2 are the easiest prospects. Their eyes were well illuminated, and they made few extraneous movements throughout the video. Subjects 3 and 4 are slightly more challenging, exhibiting some head movements and a very low blink rate, respectively. Subject 5 has a considerably higher blink rate, along with less ideal illumination of the eyes and minor head movements. Finally, subject 6 is the most challenging. Problems that must be overcome include poor illumination of the eyes, high blink rate, squinting, clusters of blinks, blinking with only one eye fully closed, head movements and accidentally knocking the camera at one point in the video.

## 6.4 Feature Extraction

Video data as described in the previous section could be interpreted verbatim as a change detection problem by investigating the difference between subsequent frames – a motion based blink detection system. This is undesirable

for a number of reasons. There may be considerable unrelated movement in the video such as head or mouth movements which would appear prominently in a frame-to-frame difference image. We also know that there is a relatively small ROI (region of interest) and that this ROI is not static over time. While a sufficiently advanced feature extraction process solves these problems, the focus of this chapter is on change detection rather than feature extraction. The basic feature extraction objectives for this problem are:

1. Identify the ROI within any given frame.

2. Extract a consistently-sized set of features from the ROI which represents the change we wish to detect.

In this work, a model of eye openness based on colour histograms was chosen. This is similar to the approaches taken by Królak and Strumiłło [91] and Lee et al. [110], although the latter uses a difference image between frames.

### 6.4.1 ROI Identification

The identification of facial features within images is a well studied problem. A common approach is to employ a boosted cascade of Haar feature classifiers, also known as the Viola-Jones algorithm [169]. Haar features are filters which are convolved over an image and produce a numerical score for a given region within the image. Figure 6.4 shows a common set of Haar features used for object detection, from Lienhart and Maydt [113]. The score for the feature is calculated as the sum of the pixel intensities within the black area as a proportion of the sum for the whole filter. This results in a very large number of candidate features, so a weak classifier set is created, each consisting of a feature, a threshold and a direction. Boosting is then applied to the set to produce a strong classifier, acting as a feature selection process. Figure 6.5, from Viola and Jones [170] shows Haar features selected by AdaBoost, which

exploit the relative darkness of the brow compared to the cheeks, and the difference in intensity of the eye wells against the nose.

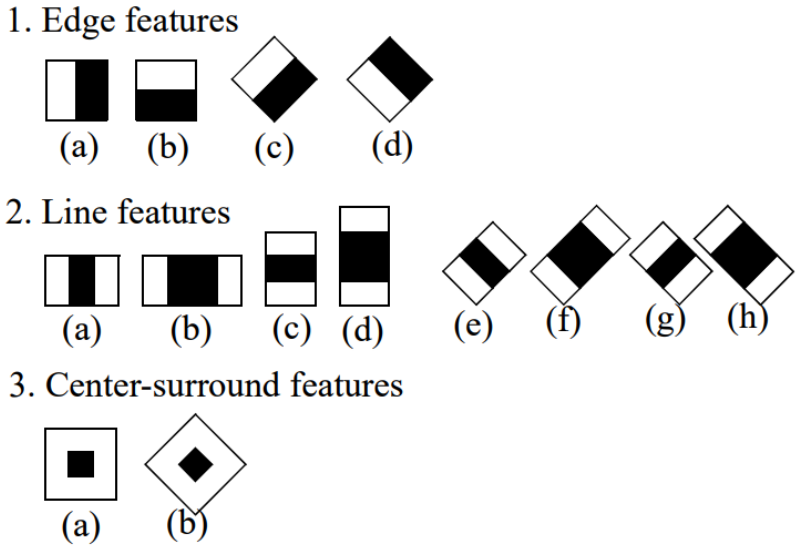



**Figure 6.4:** A example set of Haar features, from Lienhart and Maydt [113].

In order to make the above process fast enough to perform in real time, strong classifiers are combined into a special decision tree, called a cascade. Generally, a classifier which considers more features will have a higher accuracy and lower false positive rate, at the cost of more computation. As usually the vast majority of sub-windows within an image will not contain a face, it is highly inefficient to apply the best classifiers in the first instance. Therefore, the cascade is arranged such that simpler classifiers are used first to identify sub-regions of interest at which point progressively more expensive classifiers are used. The cascade described in Viola and Jones [170] is composed of 38 stages and up to 6000 features, for example.

The experiments in this chapter use the pre-trained 'EyePair' cascade classifier provided in the MATLAB Computer Vision System Toolbox [125]. For each video frame, a number of candidate eye pair bounding boxes are predicted, sorted by confidence. For some frames, no bounding boxes are returned, or the most confident bounding box is wrong. To guard against this, a number of measures are employed to ensure a good quality ROI extraction. Algorithm 8 illustrates this process in pseudocode.

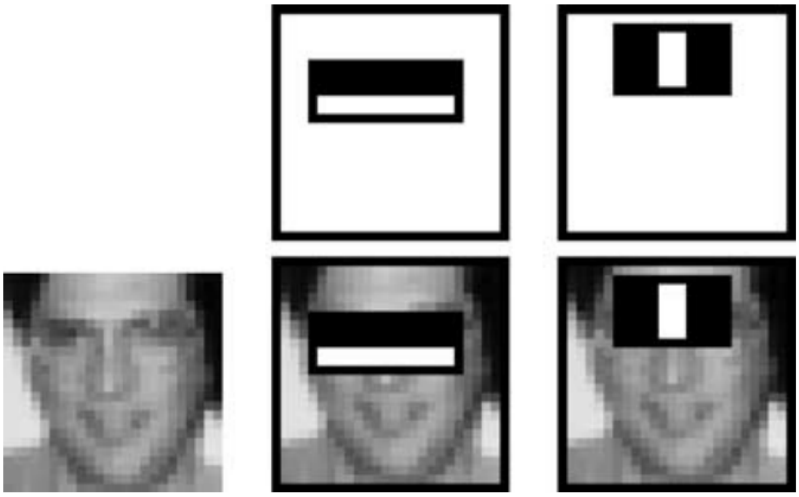


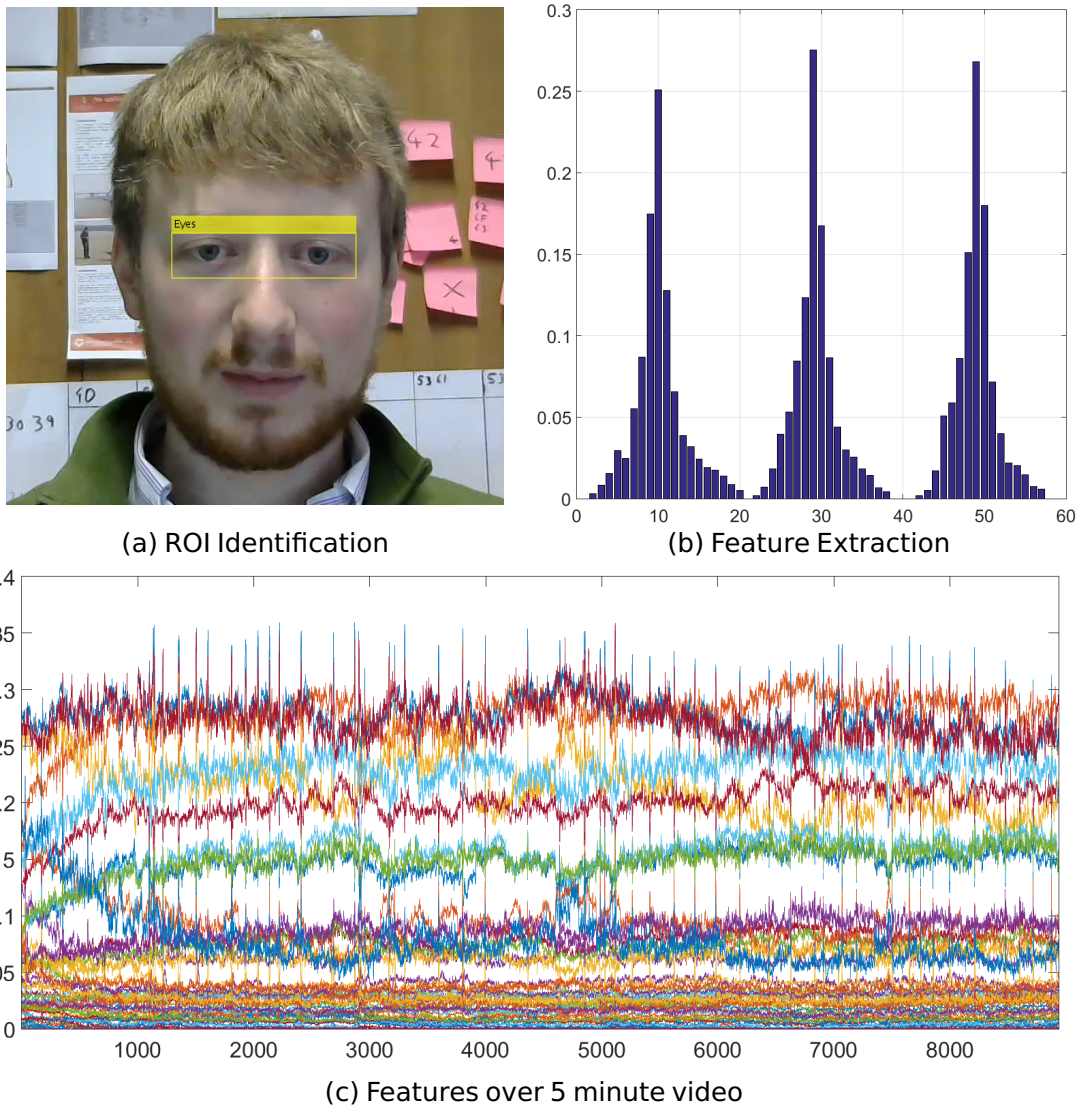
**Figure 6.5:** Haar features selected by AdaBoost, from Viola and Jones [170].

**Algorithm 8:** Feature extraction from video

```
video ← load_video();
boxes ← [];
features ← [];
n ← 0
while has_frame(video) do
    frame = read_frame(video);
    bbox = haar_cascade(frame);
    boxes = boxes ∪ bbox;
    if size(boxes) ≥ 30 then
        boxes = boxes \ boxes[0];
    end

    chosen_bbox = median(boxes);
    roi = extract_roi(frame, chosen_bbox);

    hist_red = histogram(get_channel(roi, 0), 20);
    hist_green = histogram(get_channel(roi, 1), 20);
    hist_blue = histogram(get_channel(roi, 2), 20);
    features[n] = [hist_red, hist_green, hist_blue]

    n = n + 1;
end
```

The most confident prediction is taken (if there is one) at each frame and added to an array of bounding boxes with a maximum size of 30. 30 frames was chosen as it represents a period of approximately one second in the video. A bounding box takes the form of a 4-element vector $[x,y,w,h]^T$ of position, width and height. Combining the bounding boxes into a 30 by 4 matrix, the bounding box for the ROI is taken as the columnwise median of this matrix. This combin-

ation of steps was found to provide the best balance of responsiveness to head movement and robustness against poor and non-existent bounding boxes.




(a) ROI Identification

(b) Feature Extraction

(c) Features over 5 minute video

**Figure 6.6:** (a) A haar-cascade computed eye bounding box. (b) Histogram of the bounding box in (a). (c) Progression of the 60 features over the whole video.

### 6.4.2 Histogram Calculation

The size of the extracted ROI is variable per frame, and this inconsistent number of features is problematic for most change detection algorithms. Therefore, a consistently-sized histogram was calculated and used to provide the features. The hypothesis is as follows. There is a considerable colour difference between an image of an open and closed eye. If a histogram is generated from the image pixel values, over several frames, there should be a detectably large

difference in the counts of skin colour and eye colour. Furthermore, this should generate a detectable difference in subsequent frames invariant of skin colour, eye colour or lighting conditions.

Separating the red, green and blue channels of the ROI, the pixel values of each are grouped into 20 equally sized buckets. The combination of these three histograms form a consistent 60-dimensional space in which change should be detectable. Figure 6.6 (a) depicts the ROI identification, (b) an example histogram for a single frame, and (c) the progression of the 60 features over the course of the 5 minute video. Note that the blinks are visible to the naked eye, and also that there are a considerable amount of noisy features which suggest further feature extraction would be beneficial. These noisy features are an expected byproduct of this approach. Beginning with the assumption that we will observe the difference between the colour of an open and closed eye, it stands to reason that a histogram of a static (relative to the face) ROI will have a mostly static distribution of colours over time under stable illumination. It is expected that the histogram buckets which vary significantly over time belong to the colours of the eyes. This is assumed due to the proportion of the ROI which is taken up by the eyes, and the lack of other possible facial movements within the ROI which could cause colour changes.

## 6.5 Experiment 1: Baseline

### 6.5.1 Method

The objective of the first experiment was to establish the baseline performance of the pure multivariate change detectors on the problem. The detectors were evaluated individually for each subject. Since the multivariate detectors all require sliding windows of data, a range of window sizes were evaluated to

account for the parameter choice. The process is illustrated in pseudocode in Algorithm 9.

**Algorithm 9:** Evaluation process for multivariate detectors. Each run stores the subject, detector name and window size, along with the four performance metrics; average run length (ARL), time to detection (TTD), false alarm rate (FAR) and missed detection rate (MDR).

```
subjects ← {1..6}
detectors ← { Hotelling, SPLL, KL }
window_sizes ← {5,10,15,20,25,30,35,40,45,50}
for subject ∈ subjects do
    features = load_features(subject);
    for d_name ∈ detectors do
        for wsz ∈ window_sizes do
            detector = build_detector(d_name,wsz);
            ARL,TTD,FAR,MDR = evaluate(detector,features);
            store(subject,d_name,wsz,ARL,TTD,FAR,MDR);
        end
    end
end
```

The four performance metrics discussed in Section 2.7 are recorded for each experiment – average run length (ARL), time to detection (TTD), false alarm rate (FAR) and missed detection rate (MDR).

Eye blink data presents a unique challenge when assigning true labels. The state of a frame is unavoidably subjective, open to difference of opinion through human interpretation. Whilst open and closed states are trivial to differentiate, there is a transition period between the states which is uncertain. In Section 6.3.1 two interpretations of the data were proposed; optimistic and pessimistic. In the former, transition states are interpreted as part of the blink, while in the latter they are interpreted as open eyes. An optimistic interpretation of the data is a better approximation of the underlying ground truth in which we are interested, and so was used for the experiment.

Despite applying an optimistic interpretation, some legitimate detections are marked as false positives due to eye movements that often occur immediately prior to a blink, but do not pass the thresholds described in Table 6.1

for marking a frame as transition or closed. As such, we provide a small amnesty for early detections that occur $\alpha$ frames before a true positive. For this experiment, $\alpha = 5$ was chosen, or approximately one sixth of a second at 30 frames per second. A knock on effect of this is that it becomes possible for a detector to achieve a negative TTD value. For the purposes of results figures, any negative TTD values are plotted as zero.

These experiments were run using the `meander`[4] streaming change detection framework. A number of the univariate detectors used are the implementations from `MOA`[5] adapted into this framework.

It was decided that the detectors would not be reset when a change is detected, which will almost certainly contribute to an inflated false alarm rate and would not be done in practice. This was done for the following reason. In the particular case of blink detection, the true changes are likely to occur periodically roughly according to the blink rate of the subject. Let us consider subject 5, with an ideal ARL of $44.96$. Suppose that we apply a useless detector, which signals change all the time, but uses a fixed-size window of observations. Suppose further that we evaluate this detector with a window size of 45. If the detector does not signal until its window is full, it would signal every 45 frames and achieve a very favourable result, despite its performance being entirely related to the window size. Such a detector merely arrives by accident at the ideal ARL. By not resetting the detectors, we record many more legitimate detections as false alarms after the blink has been detected, but over-eager detectors have nowhere to hide. We can be confident that rerunning the experiment with resetting of the detectors will likely significantly improve the false alarm rates. The same improvement applies to simple statistical detectors like CUSUM, which maintain rolling averages.

[4] `https://github.com/wfaithfull/meander`
[5] `https://github.com/Waikato/moa`

### 6.5.2 Results Format

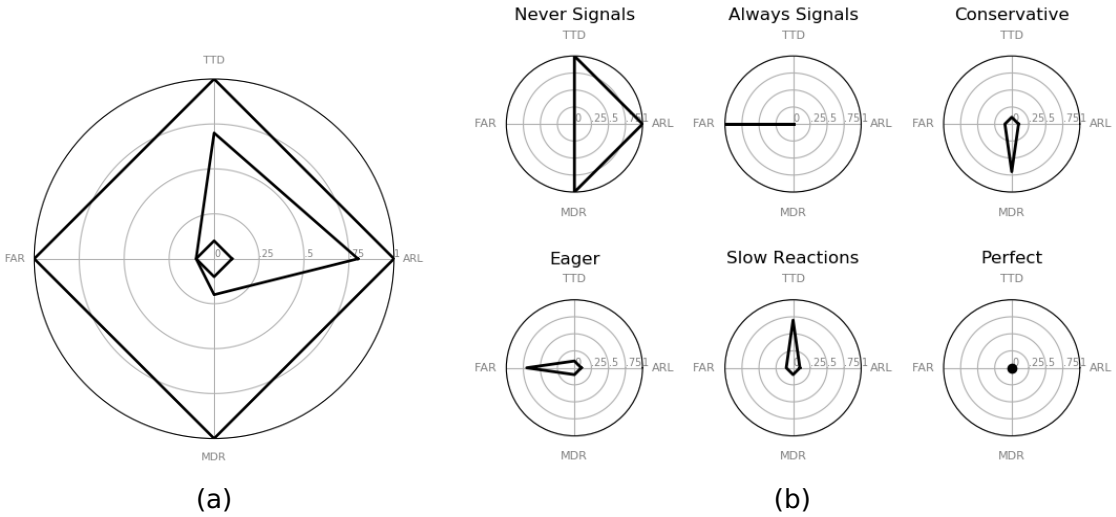



**Figure 6.7:** Results figure archetype and sample glyphs.

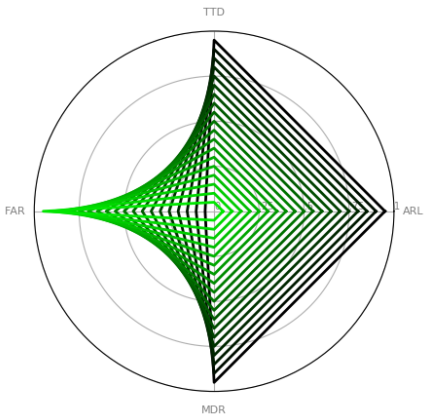



**Figure 6.8:** Progression of parameter values over subsequent runs. Low values of the parameter are in black, high values in green.

Figure 6.7 (a) demonstrates the archetypal results figure. From each individual run, the ARL, TTD, FAR and MDR are used to plot a glyph on the radar chart. FAR and MDR already reside in the $[0..1]$ interval with $0$ being ideal, but the ARL and TTD values need to be adjusted. To visually accentuate the difference in ARL and TTD at lower values, log-scales were used. To map the recorded ARL ($ARL_R$) into this space, the ideal ARL ($ARL_I$) is calculated from the data, and then the scaled distance to the ideal ARL computed.

$$ARL_{plot} = \frac{\ln(|ARL_R - ARL_I|)}{\ln(N - ARL_I)} \tag{6.1}$$

$$\max(|ARL_R - ARL_I|) = N - ARL_I \tag{6.2}$$

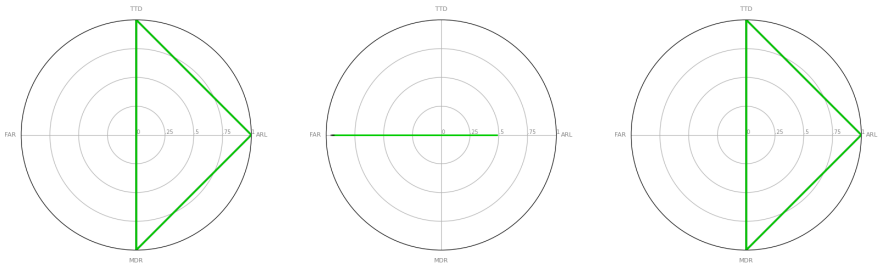
Since $N = \max(TTD)$, the TTD is plotted as

$$TTD_{plot} = \frac{ln(TTD)}{ln(N)} \tag{6.3}$$

Figure 6.7 (b) demonstrates some common 'glyphs' which provide a simple intuition about the performance of the detector. A detector which never signals will have $TTD = N, ARL = N, MDR = 1, FAR = 0$.

Figure 6.8 is how parameter progression is visualised. Regardless of which parameter is being tuned, black indicates a low value and green a high value, so that the effect of the parameter on the results is apparent. Lines are plotted at 50% opacity to visually accentuate overlaps for very small increments. Assuming the parameter value influences results, a progression should be visible in the results figures.

Each experiment will also present a ranked table of the top 20 methods in the ARL/TTD space and FAR/MDR space. This ranking will be calculated by first taking the average over all subjects and all parameter values. Then the rank is calculated as the sum of the two averages. As the ideal sum of the scaled values is 0, the methods can then be ranked in ascending order of this sum. There is no perfect ranking of these results as the importance of each metric is contextual, but the best methods should appear in the top 20 overall performers in both spaces.



(a) Hotelling (b) SPLL (c) KL

**Figure 6.9:** Radar glpyhs for the multivariate detectors averaged across all subjects and window size parameter choices.

**Table 6.3:** The top 20 performers on average in the ARL/TTD and FAR/MDR spaces.

| Detector | ARL | TTD | Detector | FAR | MDR |
|---|---|---|---|---|---|
| SPLL-W50 | 0.90 | -3.86 | SPLL-W50 | 0.77 | 0.00 |
| SPLL-W40 | 0.90 | -3.90 | SPLL-W40 | 0.77 | 0.00 |
| SPLL-W45 | 0.90 | -3.89 | SPLL-W45 | 0.77 | 0.00 |
| SPLL-W35 | 0.89 | -3.95 | SPLL-W35 | 0.77 | 0.00 |
| SPLL-W30 | 0.89 | -3.99 | SPLL-W30 | 0.78 | 0.00 |
| SPLL-W25 | 0.89 | -4.04 | SPLL-W25 | 0.78 | 0.00 |
| SPLL-W20 | 0.88 | -4.08 | SPLL-W20 | 0.78 | 0.00 |
| SPLL-W15 | 0.88 | -4.07 | SPLL-W15 | 0.79 | 0.00 |
| SPLL-W10 | 0.87 | -4.13 | SPLL-W10 | 0.79 | 0.00 |
| SPLL-W5 | 0.86 | -4.15 | SPLL-W5 | 0.80 | 0.00 |
| HOTELLING-W10 | 7443.17 | 7443.17 | HOTELLING-W10 | 0.00 | 0.83 |
| HOTELLING-W15 | 7443.17 | 7443.17 | HOTELLING-W15 | 0.00 | 0.83 |
| HOTELLING-W20 | 7443.17 | 7443.17 | HOTELLING-W20 | 0.00 | 0.83 |
| HOTELLING-W25 | 7443.17 | 7443.17 | HOTELLING-W25 | 0.00 | 0.83 |
| HOTELLING-W30 | 7443.17 | 7443.17 | HOTELLING-W30 | 0.00 | 0.83 |
| HOTELLING-W35 | 7443.17 | 7443.17 | HOTELLING-W35 | 0.00 | 0.83 |
| HOTELLING-W40 | 7443.17 | 7443.17 | HOTELLING-W40 | 0.00 | 0.83 |
| HOTELLING-W45 | 7443.17 | 7443.17 | HOTELLING-W45 | 0.00 | 0.83 |
| HOTELLING-W50 | 7443.17 | 7443.17 | HOTELLING-W50 | 0.00 | 0.83 |
| HOTELLING-W5 | 7443.17 | 7443.17 | HOTELLING-W5 | 0.00 | 0.83 |
| KL-W5 | 7443.17 | 7443.17 | KL-W5 | 0.00 | 0.83 |

### 6.5.3 Results

We can see from Figure 6.9 that the multivariate detectors fare extremely poorly at this problem across the board. Both the Hotelling and KL detectors resemble the ’Never Signals’ glyph from Figure 6.7. The SPLL plots closely resemble the ’Always Signals’ glyph. There is little to no variation in the results between subjects and across parameter values. These intuitions are reflected in the global average matrices in Table 6.4. In Table 6.3 we can see that minor

**Table 6.4:** Global average performance values for each detector.

$$\begin{pmatrix} ARL & TTD \\ FAR & MDR \end{pmatrix}$$

| | | |
|---|---|---|
| $\begin{pmatrix} 7443.17 & 7443.17 \\ 0.00 & 0.83 \end{pmatrix}$ | $\begin{pmatrix} 0.89 & -4.01 \\ 0.78 & 0.00 \end{pmatrix}$ | $\begin{pmatrix} 7443.17 & 7443.17 \\ 0.00 & 0.83 \end{pmatrix}$ |
| (a) Hotelling | (b) SPLL | (c) KL |

variations aside, the SPLL detector signalled change more often than it did not, and the other two detectors almost never signalled, having an ARL close to the number of frames and missing most changes.

## 6.6 Experiment 2: Chaining Detectors

Following on from the poor results of the baseline experiment, here the hypothesis from Chapter 4 is examined. It is proposed that by using univariate detectors to monitor the statistics produced by the multivariate detectors, we can achieve better results compared to the statistically justified fixed thresholds.

### 6.6.1 Motivation

The three multivariate detectors all reduce the input space to a continuous statistic. Recall their respective flowcharts from Figure 2.20. We will refer to a statistic generated by each detector as $S_{SPLL}$, $S_{KL}$ and $S_H$ for the SPLL, KL and Hotelling detectors respectively.

Continuing from the discussion in Chapter 4, we should first consider what is the expected behaviour of each statistic in response to change. The thresholds that are normally applied to $S_{SPLL}$ and $S_H$ are straightforward $95\%$ confidence intervals on the expected distribution of the statistic. That is, $S_{SPLL} \sim \chi^2_p$ and $S_H \sim T^2_{p,m}$. For KL, a log-likelihood ratio can be calculated to infer a threshold.

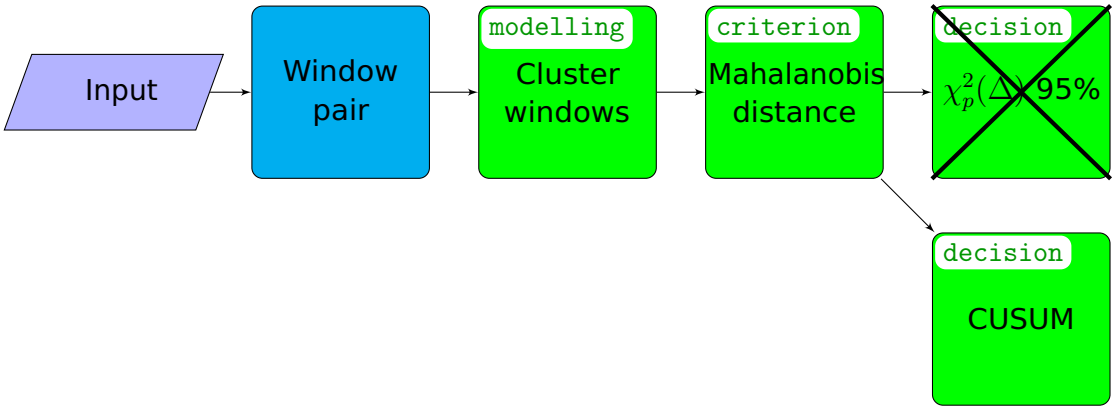



**Figure 6.10:** The normal chi squared confidence interval threshold for SPLL is replaced with a univariate change detector, such as CUSUM.

In considering whether to replace these thresholds, we will investigate what the behaviour of each statistic is likely to be at the advent of change.

SPLL calculates $S_{SPLL}$ as the average Mahalanobis distance from each leading window example to its nearest cluster. As change occurs, we expect a change in the clustering between the leading and trailing window. Whilst this statistic may reduce if the clusters become more compact over time in response to the change, there are considerably more possible movements which will result in a maximisation of the statistic with change. This is a similar argument to that expressed in Chapter 3, Figure 3.4.

If the Kullback-Leibler distance between two samples is zero, the hypothesis is that the two samples are drawn from identical distributions. As we slide windows over a change, the leading window fills with examples of the new concept. Therefore it follows that $S_{KL}$ should display a short-lived peak in response to change.

The Hotelling detector applies a multivariate $T^2$ test between the two samples represented by the pair of windows. $S_H$ is simply the $T^2$ statistic. Since the $T^2$ statistic is equivalent to the Mahalanobis distance between the samples multiplied by a constant [102], a detectable change would be expected to manifest as an increase in $S_H$.

Revisiting the common types of change Figure 2.2, we can hypothesise that it is likely to be short-lived reoccurring concepts. As discussed in Section 6.3.2, an endogenous blink is expected to last 100–400ms, or between 3–12 frames at 30FPS. An appropriately sized window will capture examples of both concepts. Therefore it can be argued that all three change detector statistics can be reasonably expected to produce localised maxima in response to change. The motivation for chaining detectors is to exploit this property. The basic idea is illustrated in Figure 6.10.

### 6.6.2 Method

Attempting to improve upon the performance of the fixed thresholds of the multivariate detectors, a selection of univariate change detectors were used to monitor the statistics and signal change. The choice of detectors are listed in Tables 6.5 and 6.6. In particular, univariate detectors were chosen that did not make strong assumptions about the distribution of the input data, and this choice was further informed by their relative performance in the main experiment of Chapter 5.

The experimental process is detailed in pseudocode in Algorithm 10. The three multivariate detectors in Table 6.6 were evaluated at a range of window sizes from $[5,50]$, as before. However, each detector was tested with its own thresholding mechanism removed and replaced with each of the six univariate detectors in Table 6.5.

### 6.6.3 Results

The glyphs in Figure 6.11 show that this approach bears fruit in particular for the SPLL statistic. Although there were minor improvements with H-ADWIN and

**Algorithm 10:** Evaluate multivariate detectors chained with univariate detectors

```
subjects ← {1..6}
mv_detectors ← { H, SPLL, KL }
uv_detectors ← {ADWIN, CUSUM, PH, GMA, MR, SEED}
window_sizes ← {5,10,15,20,25,30,35,40,45,50}
for subject ∈ subjects do
    features = load_features(subject);
    for mv_name ∈ mv_detectors do
        for wsz ∈ window_sizes do
            for uv_name ∈ uv_detectors do
                mv_detector = build_detector(mv_name,wsz);
                uv_detector = build_detector(uv_name);

                detector = chain(mv_detector,uv_detector);
                ARL,TTD,FAR,MDR = evaluate(detector,features);
                d_name = "%s-%s".format(mv_name,uv_name);

                store(subject,d_name,wsz,ARL,TTD,FAR,MDR);
            end
        end
    end
end
```

**Table 6.5:** Univariate change detectors chosen to be used as thresholds.

| Univariate methods | Acronym |
|---|---|
| ADWIN | ADWIN |
| CUSUM | CUSUM |
| Page-Hinkley | PH |
| Geometric Moving Average Chart | GMA |
| Moving Range Chart | MR |
| SEED | SEED |

**Table 6.6:** Multivariate change detectors to be rethresholded.

| Multivariate Methods | Acronym |
|---|---|
| SPLL | SPLL |
| KL | KL |
| Hotelling | H |

H-SEED, the change context does not appear to be particularly well represented in the $T^2$ or KL statistics, as both detectors tend towards a 'Never Signals' glyph. This apes the findings in the baseline experiment where these detectors also failed to signal. In the baseline, the SPLL detector tended towards an 'Always Signals' glyph, and in this instance the chosen univariate detectors appear to be better at taming the SPLL statistic into a more conservative performance.

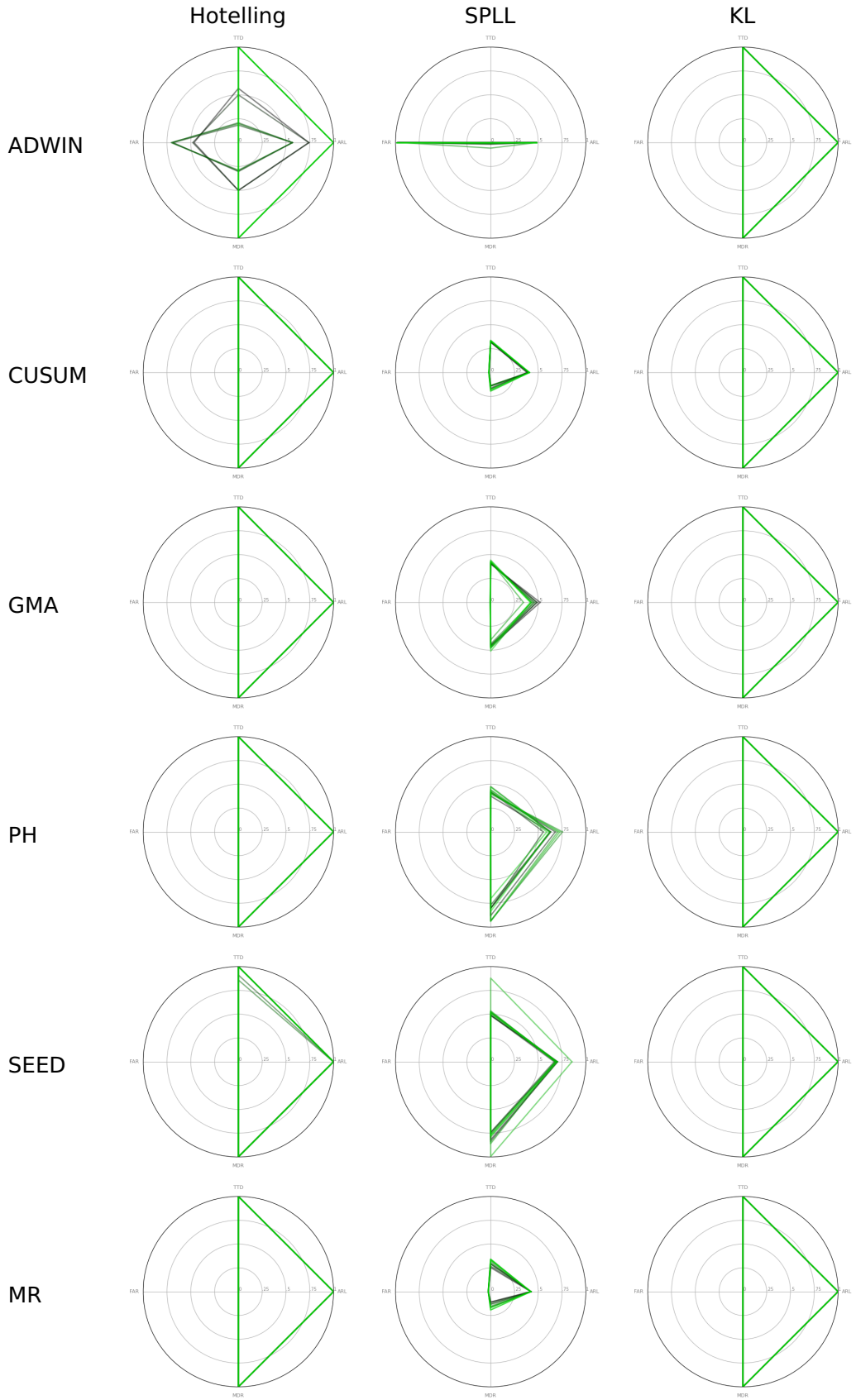



**Figure 6.11:** Radar glpyhs for Hotelling, SPLL and KL using the specified univariate detectors for thresholding. The glyphs are averages over all subjects.

**Table 6.7:** The top 20 performers in the ARL/TTD and FAR/MDR spaces.

| Detector | ARL | TTD | Detector | FAR | MDR |
|---|---|---|---|---|---|
| SPLL-ADWIN-W30 | 0.92 | -3.94 | SPLL-MR-W10 | 0.02 | 0.06 |
| SPLL-ADWIN-W10 | 0.86 | -3.86 | SPLL-MR-W5 | 0.02 | 0.06 |
| SPLL-ADWIN-W20 | 0.86 | -3.96 | SPLL-CUSUM-W25 | 0.02 | 0.08 |
| SPLL-ADWIN-W25 | 0.85 | -3.58 | SPLL-CUSUM-W10 | 0.01 | 0.08 |
| SPLL-ADWIN-W5 | 0.84 | -3.94 | SPLL-CUSUM-W30 | 0.02 | 0.08 |
| SPLL-ADWIN-W15 | 0.84 | -3.94 | SPLL-MR-W15 | 0.02 | 0.08 |
| SPLL-ADWIN-W50 | 0.84 | -3.62 | SPLL-MR-W20 | 0.02 | 0.08 |
| SPLL-ADWIN-W40 | 0.84 | -3.72 | SPLL-CUSUM-W20 | 0.02 | 0.09 |
| SPLL-ADWIN-W45 | 0.84 | -3.72 | SPLL-CUSUM-W5 | 0.01 | 0.10 |
| SPLL-ADWIN-W35 | 0.84 | -3.82 | SPLL-MR-W25 | 0.02 | 0.10 |
| SPLL-MR-W5 | 28.08 | 9.52 | SPLL-MR-W35 | 0.02 | 0.10 |
| SPLL-MR-W15 | 28.11 | 10.01 | SPLL-MR-W30 | 0.02 | 0.11 |
| SPLL-CUSUM-W5 | 39.56 | 16.54 | SPLL-CUSUM-W15 | 0.02 | 0.12 |
| SPLL-MR-W10 | 27.06 | 11.91 | SPLL-CUSUM-W35 | 0.02 | 0.11 |
| SPLL-CUSUM-W10 | 38.11 | 17.27 | SPLL-CUSUM-W50 | 0.02 | 0.12 |
| SPLL-MR-W20 | 28.01 | 13.05 | SPLL-CUSUM-W40 | 0.02 | 0.12 |
| SPLL-MR-W25 | 29.41 | 14.10 | SPLL-MR-W45 | 0.02 | 0.12 |
| SPLL-CUSUM-W20 | 35.66 | 17.99 | SPLL-CUSUM-W45 | 0.02 | 0.13 |
| SPLL-CUSUM-W15 | 35.92 | 18.34 | SPLL-MR-W40 | 0.02 | 0.13 |
| SPLL-CUSUM-W30 | 34.37 | 17.28 | SPLL-MR-W50 | 0.02 | 0.13 |

The SPLL plots show a small but clear effect from tuning the window size. For MR, CUSUM, PH and SEED, darker values tend to be associated with a lower TTD and lower MDR. This implies as expected, that a smaller window size makes the detectors less conservative. For GMA, the smaller window sizes are clearly related to a greater deviation from the ideal ARL, although this appears to be the opposite case for PH.

In Table 6.7 we see that SPLL based methods dominate the top 20 in the two categories. Of particular note are SPLL-MR and SPLL-CUSUM with low window sizes, which are well represented in the top 20 of both spaces. The global averages in Table 6.8 also reflect this, although GMA was closest on average to the ideal ARL, despite being unrepresented in the top 20. The top 20 in the ARL/TTD space is dominated by the SPLL-ADWIN detectors, due to equal weight given to ARL difference and TTD. Since the ADWIN detectors were extremely eager, signalling change almost all the time, they achieved a zero TTD. Despite this being a useless outcome, it led to a favourable rank due to very low ARL

**Table 6.8:** Global averages across the chained detectors.

| | Hotelling | SPLL | KL |
|---|---|---|---|
| MR | $\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$ | $\begin{pmatrix} 28.4774 & 15.5520 \\ 0.0210 & 0.0984 \end{pmatrix}$ | $\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$ |
| ADWIN | $\begin{pmatrix} 5061.7862 & 5061.3446 \\ 0.2223 & 0.6054 \end{pmatrix}$ | $\begin{pmatrix} 0.8533 & -3.8117 \\ 0.8115 & 0.0103 \end{pmatrix}$ | $\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$ |
| CUSUM | $\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$ | $\begin{pmatrix} 35.1398 & 17.9865 \\ 0.0156 & 0.1041 \end{pmatrix}$ | $\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$ |
| GMA | $\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$ | $\begin{pmatrix} 127.3361 & 47.7455 \\ 0.0030 & 0.3386 \end{pmatrix}$ | $\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$ |
| PH | $\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$ | $\begin{pmatrix} 475.4375 & 67.4372 \\ 0.0009 & 0.6891 \end{pmatrix}$ | $\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$ |
| SEED | $\begin{pmatrix} 7411.4500 & 7146.6000 \\ 0.0000 & 0.8328 \end{pmatrix}$ | $\begin{pmatrix} 1075.0214 & 705.3320 \\ 0.0003 & 0.6566 \end{pmatrix}$ | $\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$ |

values and the ideal ARL being much closer to 0 than to the maximum number of frames. The best overall performance therefore belongs to the SPLL-MR-W5 detector, which achieved an average 94% accuracy for a false alarm rate of only 2%, whilst being the top non-ADWIN performer in the ARL/TTD space.

## 6.7 Experiment 3: Ensembles of Univariate Detectors

In parallel with the last experiment, here ensembles of univariate change detectors are evaluated as in Chapter 5 as an alternative to the multivariate detectors.

### 6.7.1 Motivation

It is clear in Figure 6.6 that the extracted features still retain a considerable amount of noise. Any successful approach will need to discern these noisy features from the useful ones. The mechanism by which feature-wise ensembles achieve this is quite straightforward. The discussion in Section 6.4.2 notes that the noisy features are likely to be mostly static over time. As each individual feature is monitored by a member of the ensemble, if the noisy features remain stable, then these ensemble members will ideally not signal change. Assuming a well-behaved change detector, the ideal agreement percentage will coincide with the number of useful features.

### 6.7.2 Method

The same six univariate detectors were used as in the previous experiment; again because they make few or no assumptions about the distribution of the input data. The ensembles are built in the same construction as in Figure 5.2, all identical detectors, with each detector monitoring a single feature in the input space. The experimental procedure is detailed in psuedocode in Algorithm 11.

**Algorithm 11:** Evaluate ensembles of univariate detectors.

```
subjects ← {1..6}
uv_detectors ← {ADWIN, CUSUM, PH, GMA, MR, SEED}
agreements ← {.01,.05,.1,.15,.2,.25}
for subject ∈ subjects do
    features = load_features(subject);
    for agreement ∈ agreements do
        for uv_name ∈ uv_detectors do
            detector = build_ensemble(uv_name,60);
            ARL,TTD,FAR,MDR = evaluate(detector,features);
            d_name = "%s−%s".format(uv_name,agreement×100);

            store(subject,d_name,agreement,ARL,TTD,FAR,MDR);
        end
    end
end
```

The ensembles are built at a range of agreement thresholds; 1%, 5%, 10%, 15%, 20% and 25%. These were chosen from observation of the results figures in Chapter 5, where thresholds of 1%, 5%, 10%, 20%, 30%, 40%, 50% were used. It can be seen in Figures 5.5 and 5.6 that even 30% is generally well beyond the optimal point on the curve for both metrics across all the datasets. In this case, we have 60 features, which means that a 10% agreement threshold requires the agreement of six detectors to signal change.

### 6.7.3 Results

Figure 6.12 demonstrates that ADWIN and MR ensembles are the best performers on this problem. Furthermore, the progression of results in the radar glyphs demonstrates that these approaches are highly tunable. This is visible to a lesser extent for the CUSUM, PH and SEED ensembles. There is a clear relationship with the decision threshold, the detectors becoming more conservative as the threshold is increased. The fact that this is best represented in the ADWIN and MR glyphs suggest that this change context is well represented in their statistics.

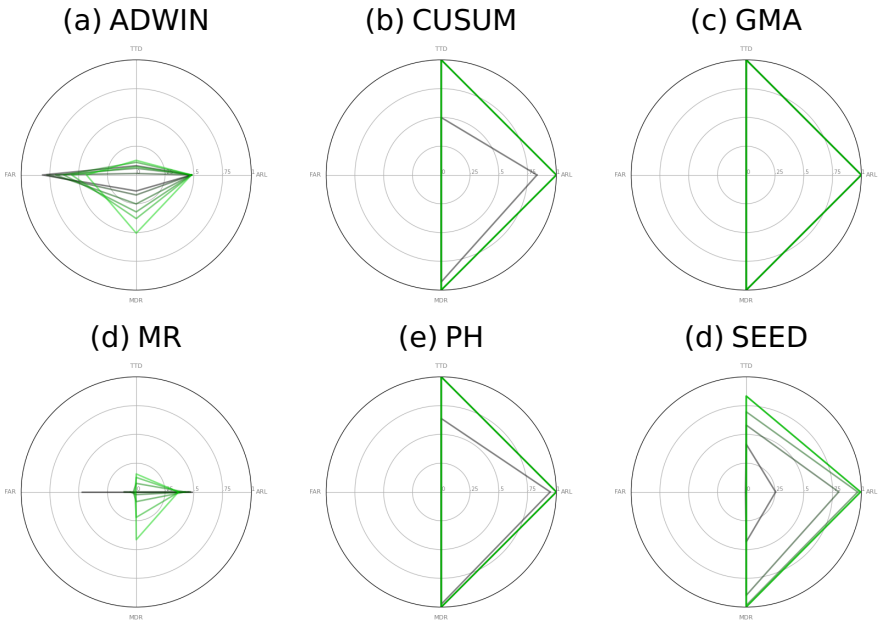



**Figure 6.12:** Radar glpyhs for the ensembles at 1%, 5%, 10%, 15%, 20% and 25%. Progression is visible from black (1%) to green (25%).

**Table 6.9:** The top 20 performers in the ARL/TTD and FAR/MDR spaces.

| Detector | ARL | TTD | Detector | FAR | MDR |
|---|---|---|---|---|---|
| MR-5 | 8.62 | -1.28 | MR-15 | 0.02 | 0.02 |
| MR-1 | 1.84 | -3.15 | MR-10 | 0.04 | 0.01 |
| MR-15 | 28.70 | 3.10 | MR-5 | 0.08 | 0.00 |
| ADWIN-5 | 1.06 | -2.14 | MR-20 | 0.01 | 0.10 |
| MR-10 | 16.01 | 0.49 | MR-25 | 0.00 | 0.27 |
| MR-20 | 58.53 | 21.27 | SEED-1 | 0.01 | 0.33 |
| ADWIN-10 | 1.16 | 1.53 | MR-1 | 0.38 | 0.00 |
| MR-25 | 160.82 | 13.25 | SEED-5 | 0.00 | 0.74 |
| ADWIN-20 | 1.53 | 6.62 | CUSUM-1 | 0.00 | 0.77 |
| ADWIN-1 | 1.00 | 9.61 | ADWIN-20 | 0.47 | 0.30 |
| ADWIN-15 | 1.29 | 4.99 | ADWIN-25 | 0.36 | 0.42 |
| ADWIN-25 | 2.23 | 6.33 | ADWIN-15 | 0.54 | 0.25 |
| SEED-1 | 82.62 | 43.27 | ADWIN-10 | 0.60 | 0.19 |
| CUSUM-1 | 1764.42 | 128.19 | ADWIN-1 | 0.69 | 0.10 |
| SEED-5 | 2397.24 | 1594.88 | ADWIN-5 | 0.66 | 0.13 |
| PH-1 | 4572.67 | 1557.08 | PH-1 | 0.00 | 0.81 |
| SEED-10 | 6640.83 | 4481.17 | SEED-10 | 0.00 | 0.82 |
| CUSUM-5 | 7443.17 | 7443.17 | CUSUM-5 | 0.00 | 0.83 |
| CUSUM-10 | 7443.17 | 7443.17 | CUSUM-10 | 0.00 | 0.83 |
| CUSUM-15 | 7443.17 | 7443.17 | CUSUM-15 | 0.00 | 0.83 |

**Table 6.10:** Global averages for the ensembles.

(a) ADWIN

$$\begin{pmatrix} 2.2346 & 6.3252 \\ 0.3638 & 0.4188 \end{pmatrix}$$

(b) CUSUM

$$\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$$

(c) GMA

$$\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$$

(d) MR

$$\begin{pmatrix} 160.8227 & 13.2456 \\ 0.0028 & 0.2720 \end{pmatrix}$$

(e) PH

$$\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$$

(d) SEED

$$\begin{pmatrix} 7443.1667 & 7443.1667 \\ 0.0000 & 0.8333 \end{pmatrix}$$

Table 6.9 clearly shows that the MR ensembles are the best represented across both categories, occupying 6 of the top 10 spots in both spaces. The MR-15 ensemble is singled out for particular praise, achieving 98% accuracy for a 2% false alarm rate, whilst on average detecting a blink one-tenth of a second after it started. A more conservative ensemble, MR-25, signalled no false positives while maintaining 73% accuracy, demonstrating the tunablity of these ensembles. The performance of SEED-1 is encouraging, with an accuracy of 67% for a 1% false alarm rate. The stark improvement in accuracy from SEED-5 suggests that the optimal threshold for SEED is any member having absolute authority.

## 6.8 Experiment 4: PCA Feature Extraction

This experiment evaluates the effect of applying PCA feature extraction as described in Chapter 3 to the previous three experiments. The data are transformed into the principal component space, and features are discarded based on their explained variance.

### 6.8.1 Method

In order to perform feature extraction, we first need to compute the principal components for the stream. The number of observations necessary to do this depends on the dimensionality of the data. For a sample of size $n$ of $p$ dimensional observations, if $n \leq p$ then there can be at most $n-1$ principal components. The rule is simply illustrated with two points in three dimensions in Figure 6.13. Since the data here has 60 dimensions, at least 60 observations are needed to compute a meaningful PCA transformation.

A scheme was suggested in Chapter 3, Section 3.3.2 where given a pair of adjacent sliding windows, $W_1$ is used to compute the principal components which are then used to transform both windows. There are a number of reasons why this scheme is undesirable in this experiment. Firstly, the largest sliding window size evaluated is 50. This means within the scheme we could only compute at most 49 principal components. Secondly, if features are discarded relative to their explained variance, then the number of features discarded may vary throughout the stream. This can be managed for detectors which look for differences between two windows of data but it is highly problematic for detectors which maintain running statistics. Finally, the ensembles used here employ many detectors which do not utilise a pair of adjacent sliding windows.

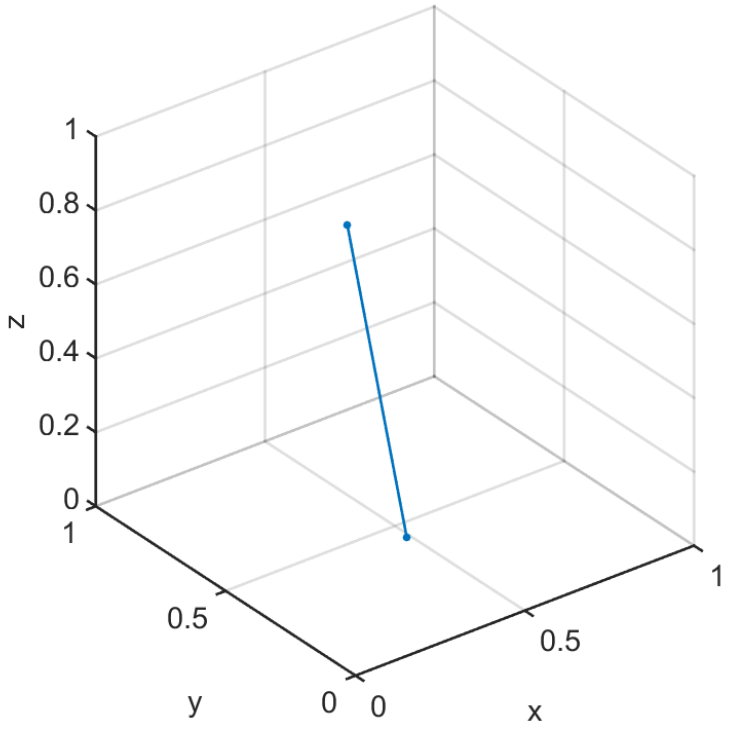



**Figure 6.13:** $n$ points in a $p$-dimensional space may only vary across $n-1$ axes if $n \leq p$. Here, $n=2$ and $p=3$.

The scheme used here is the same as in the video segmentation experiment in Section 3.4. 100 observations are sampled from the start of the stream and then used to compute principal components. These components are then used to transform the rest of the stream, and discard features based on their explained variance. The cutoff proportion of variance in this experiment was $K = 90\%$. The experiments from the previous sections were rerun twice, using both the components explaining 90% of the variance, and the components explaining the remaining 10%.

Figure 6.14 shows plots of the principal components accounting for 90% of the variance in the data for each subject. This varied between 4 and 8 principal components. It is clear from the plots that the features for subjects 4, 5 and 6 have a much weaker representation of the true change. Therefore, the components remaining 10% of the variance constituted between 56 and 52 features.

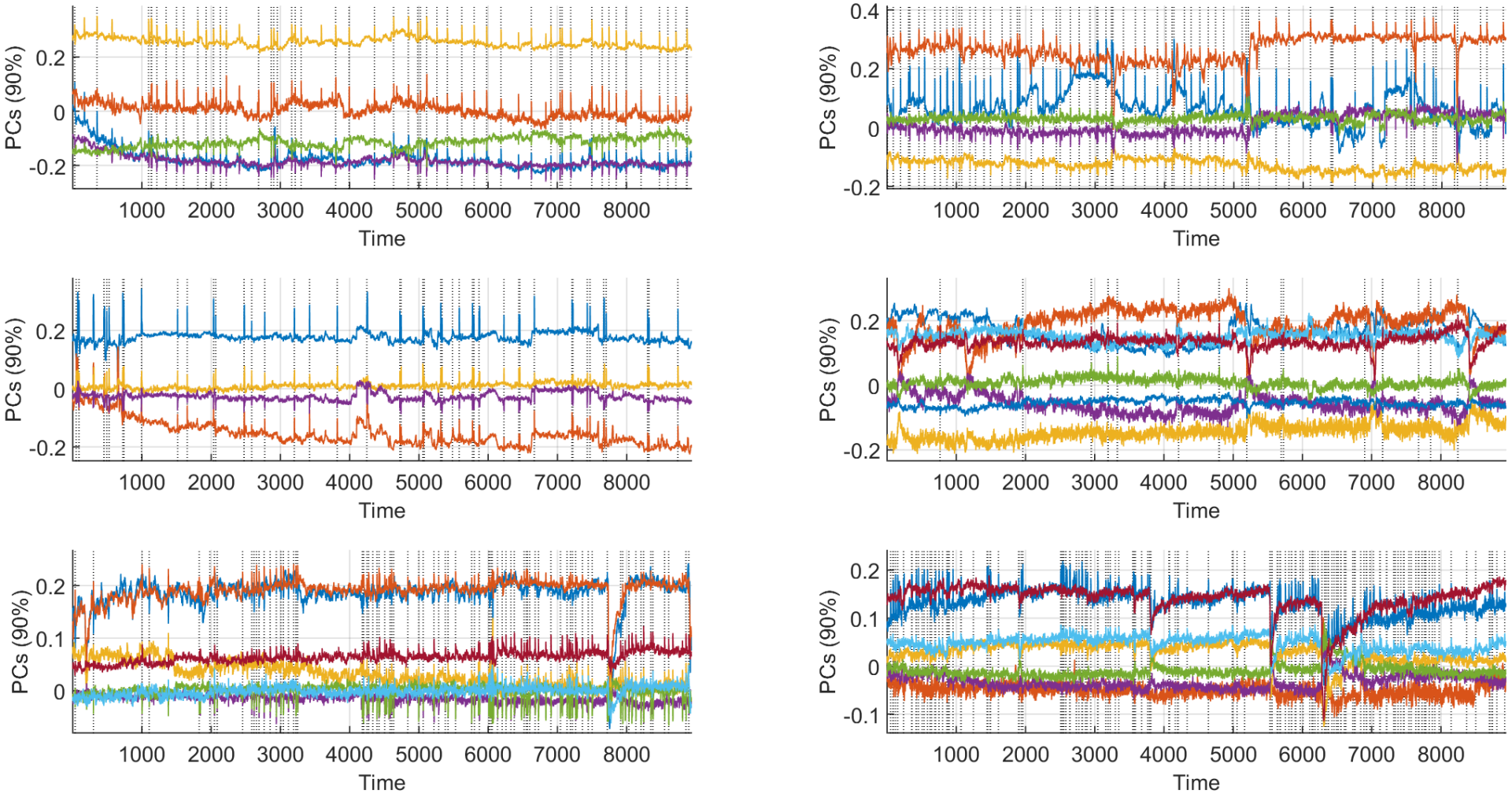



**Figure 6.14:** Plots of the features accounting for 90% of the variance in the data, transformed by PCA. Depending on the subject, this was between 4 and 8 features. The starts of the blinks are marked with dotted lines.

**Algorithm 12:** Feature extraction with PCA.

**func** $90pc, 10pc = load\_pca\_features(subject)$
- $K \leftarrow 0.9$;
- $raw\_features = load\_features(subject)$;
- $transform, eigenvalues = pca(raw\_features[0:99])$;
- $transformed\_features = raw\_features \times transform$;
- $explained = cumulative\_sum(eigenvalues) \div sum(eigenvalues)$;
- $90pc\_indices = explained[explained < K]$;
- $10pc\_indices = explained[explained >= K]$;
- $90pc = transformed\_features[90pc\_indicies]$;
- $10pc = transformed\_features[10pc\_indicies]$;

Algorithm 12 is psuedocode defining the PCA feature extraction process. Features are trimmed by variance by taking the cumulative sum of the eigenvalues, dividing through by the sum of the eigenvalues. For a vector $\vec{\lambda}$ of $n$ eigenvalues, the $j^{\text{th}}$ element of the variance explained vector $\vec{E}$ is given by

$$\vec{E}_j = \frac{\sum_{i=1}^{j} \vec{\lambda}_i}{\sum_{i=1}^{n} \vec{\lambda}_i}$$

where $\vec{E}$ is monotonic and increasing. Let $k$ be the first index where $\vec{E}_j > K$. Then features $1, ..., k$ collectively explain the amount of variance specified by $K$, and features $k, ..., n$ explain the remaining variance. The processes described in algorithms 9, 10 and 11 remain the same, except substituting $load\_features(subject)$ for $load\_pca\_features(subject)$.

### 6.8.2 Results

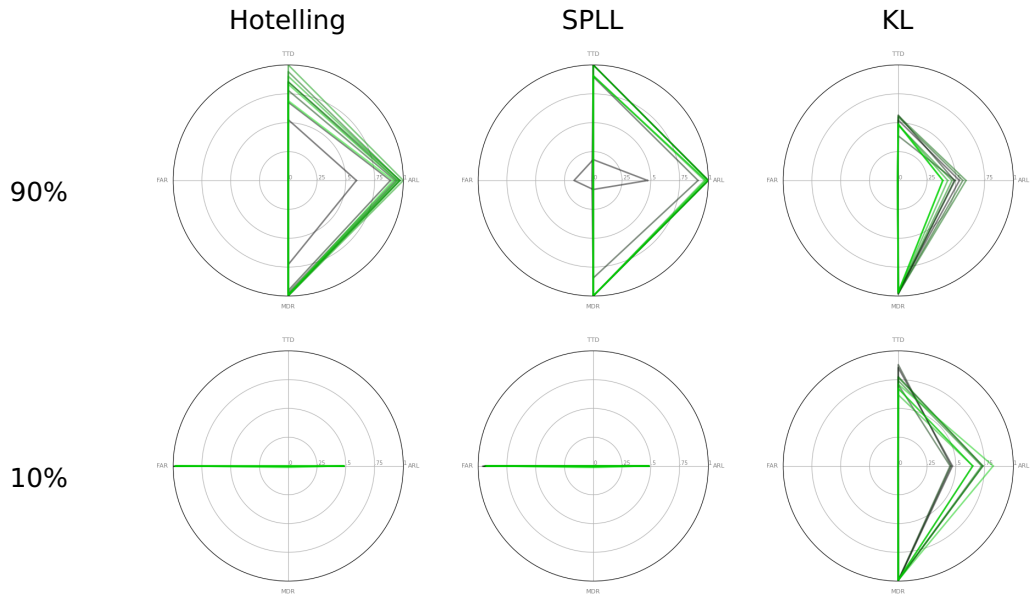



**Figure 6.15:** Radar glpyhs for Hotelling, SPLL and KL detectors with PCA feature extraction.

Figure 6.15 shows the results for the multivariate detectors with the 90% and 10% explained variance features respectively. Recalling Figure 6.9, the feature extraction process has made a clear difference. Before both Hotelling and KL resembled a 'Never Signals' glyph, with no clear progression on parameter tuning. With the 90% features, both Hotelling and KL became less conservative, while SPLL became more conservative. The reverse was true for the 10% parameters, with all detectors becoming more eager.

Figures 6.16 and 6.17 show the results for the chained detectors for 90% and 10% explained variance features respectively. Comparing to the previous results in Figure 6.11, it is clear that the 90% features in particular have enabled the Hotelling and KL statistics to be more representative of the change. This is also reflected in the 10% features, to a much lesser degree. With the 90% features, the $T^2$ statistic appears to be very competitive with the SPLL statistic, even demonstrating a better shape on the CUSUM and GMA plots. For SPLL, the 10% features demonstrate very minor changes relative to the previous results,

whilst the 90% features show greater fluctuation of results due to window size. This appears to have improved the best-case and worsened the worst-case performance.

Moving on to the ensemble results in Figure 6.18, the 10% features appear to have made an insignificant difference. Minor changes are visible in the ADWIN, PH, and SEED results, but they appear to be isolated runs rather than an overall effect. The 90% features had a more pronounced effect. The ADWIN ensemble has a more consistent performance and an improvement in MDR, in return for a reduced tunability of the FAR. We see a clear improvement in the MR and SEED ensembles. The MR ensemble delivered a considerably reduced FAR and MDR, whilst the SEED ensemble appears to be more consistently tunable along with a reduced TTD and MDR.

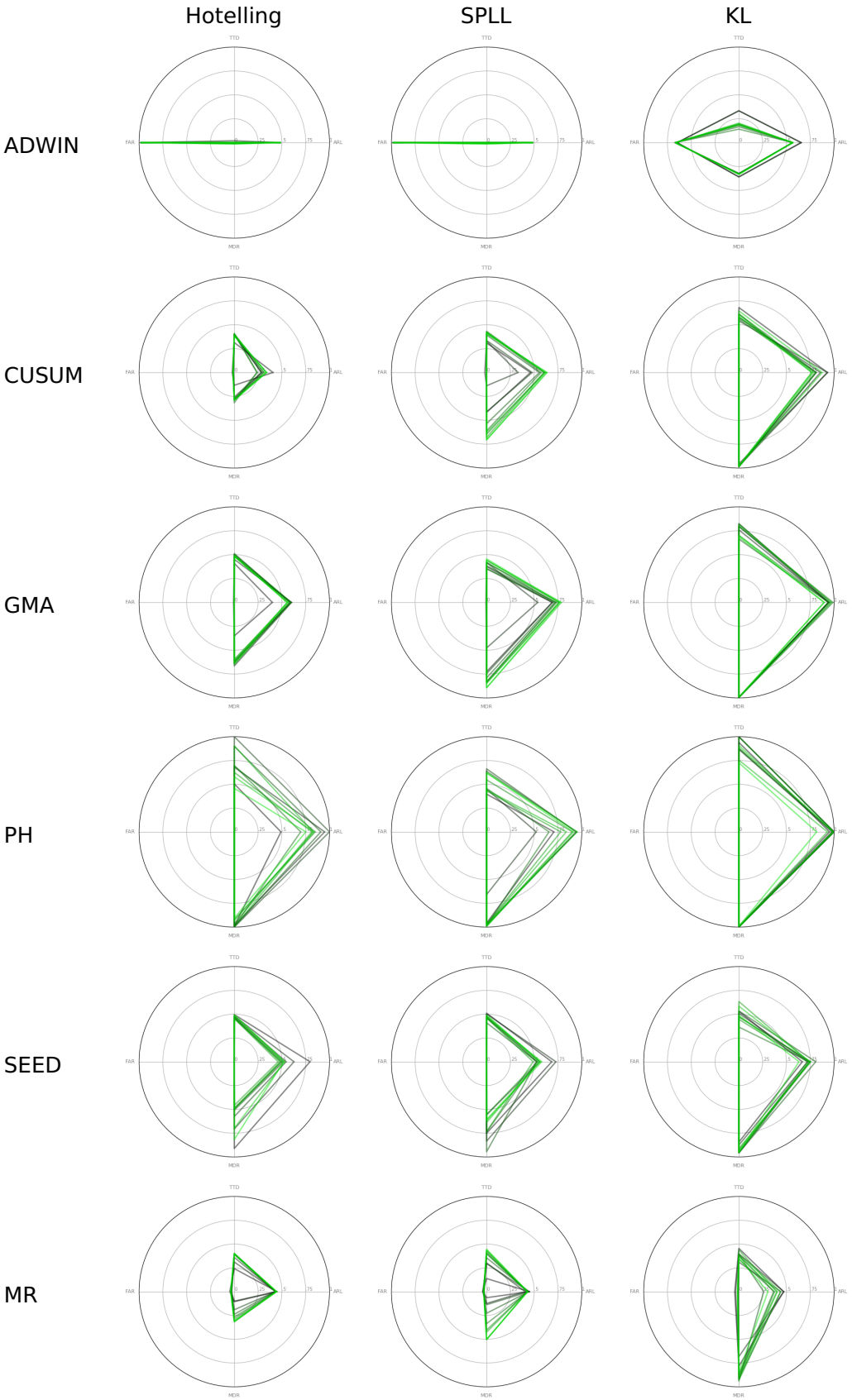



**Figure 6.16:** PCA 90% radar glpyhs for Hotelling, SPLL and KL using the specified univariate detectors for thresholding. The glyphs are averages over all subjects.

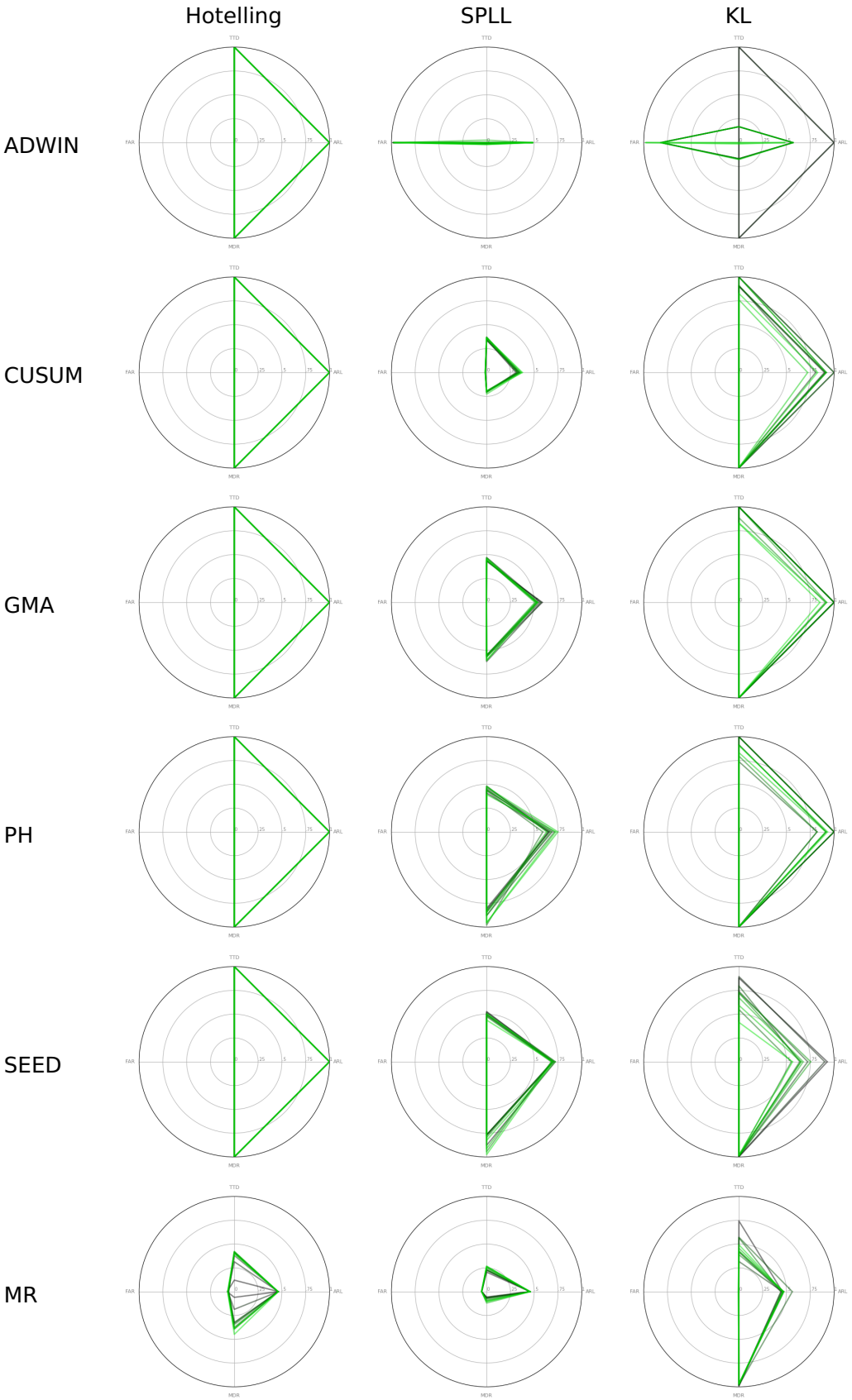



**Figure 6.17:** PCA 10% radar glpyhs for Hotelling, SPLL and KL using the specified univariate detectors for thresholding. The glyphs are averages over all subjects.

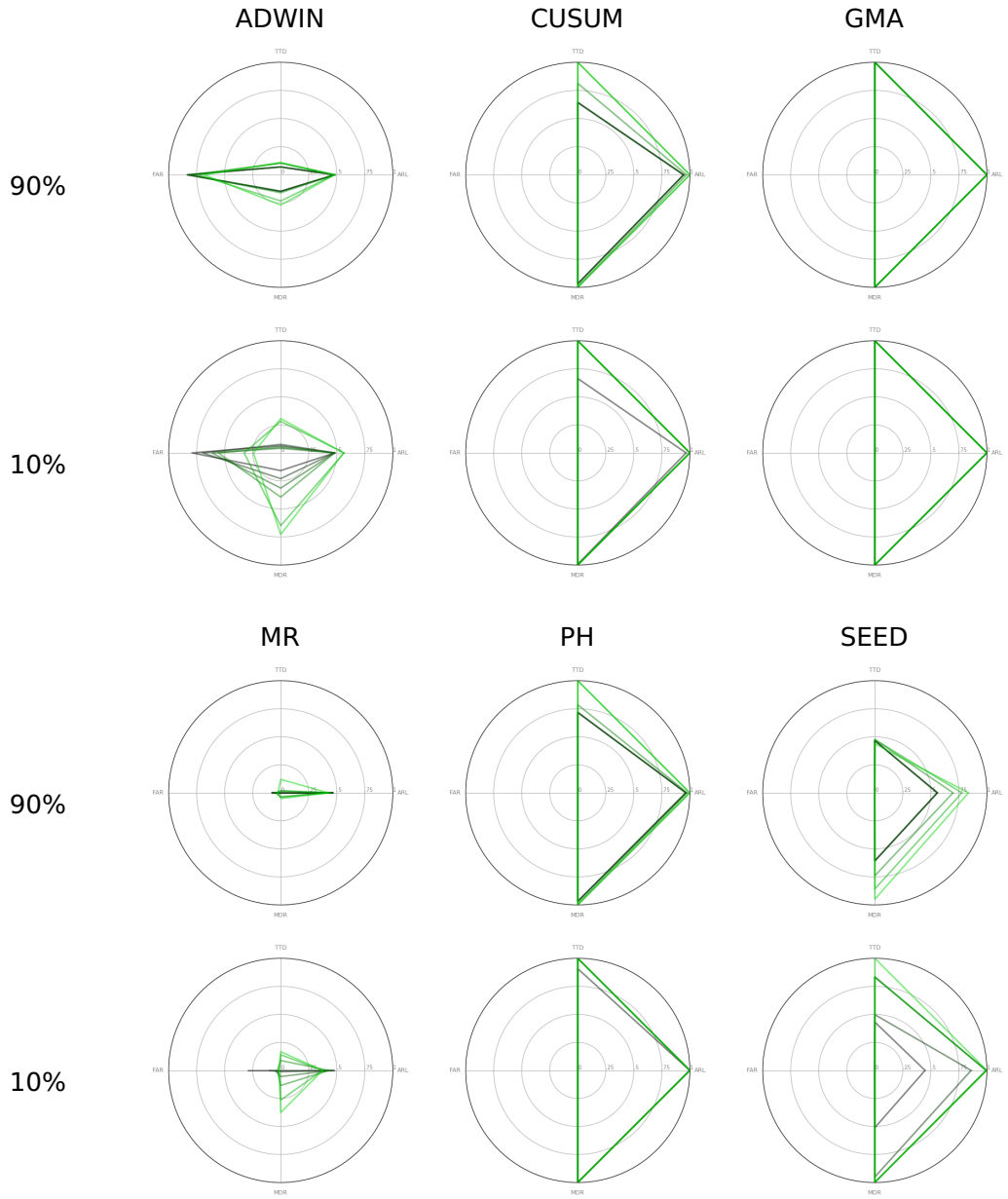



**Figure 6.18:** Radar glpyhs for Hotelling, SPLL and KL detectors with PCA feature extraction.

The global averages for the PCA experiments are in Table 6.11. The left column contains averages for the 90% variance components, the right column contains the averages for the 10% variance components. These can be directly compared with the pre-PCA averages in Tables 6.4, 6.8 and 6.10. Table 6.12 summarises the difference. For example, $\Delta_{ARL}$ is the average difference in ARL with the PCA features, compared to the baseline. The individual values are averages for a detector over all 6 subjects. Improvements are marked as •,

worse performances with ∘ and no difference with -. The improvement markers for ARL are plotted based on a reduction of the distance to the ideal ARL. The last row of the table for each experiment is the number of improvements minus the number of worse performances. A positive value indicates that PCA was generally beneficial to this metric. The mean and standard deviation of the differences, denoted as $\mu$ and $\sigma$, provide an indication of the strength of the effect.

Summing the improvement scores over all metrics and experiments, we see that the 90% features were more beneficial overall with a total sum of 28, compared to a total sum of 0 for the 10% features. The only metric which it was not an improvement for was false alarms. However, the higher false alarm rates were mostly for the chained detectors and the mean and standard deviation shows that the effect was fairly modest.

Table 6.13 shows an updated global top 20 performers across all four experiments. The moving range charts are particularly well represented, both as ensembles and chained to the multivariate detectors. We also see that a majority of the top performers use PCA features. In particular, the moving range ensembles appear to have benefited from PCA, improving their accuracy considerably for a relatively small increase in false alarms.

## 6.9 Conclusion

All the techniques applied resulted in a considerable improvement over the baseline experiment. From experiment 2 it is clear that the SPLL statistic is well representative of the change, but the $\chi^2$ threshold is inadequate to extract it. The relative success of both MR and CUSUM in this instance could have interesting applications as a quick-fix for over-eager multivariate detectors, where the threshold is holding back the performance, rather than the statistic.

**Table 6.11:** Global averages for the rerun experiments with PCA. Averages for 90% variance components are on the left, 10% variance on the right.

| | **90%** | | | **10%** | | | |
|---|---|---|---|---|---|---|---|
| | Hotelling | SPLL | KL | Hotelling | SPLL | KL | |
| | $\begin{pmatrix} 5557.40 & 4545.74 \\ 0.00 & 0.96 \end{pmatrix}$ | $\begin{pmatrix} 7741.71 & 7591.53 \\ 0.01 & 0.90 \end{pmatrix}$ | $\begin{pmatrix} 110.77 & 52.94 \\ 0.00 & 0.97 \end{pmatrix}$ | $\begin{pmatrix} 1.01 & -4.61 \\ 0.99 & 0.00 \end{pmatrix}$ | $\begin{pmatrix} 1.05 & -4.54 \\ 0.94 & 0.00 \end{pmatrix}$ | $\begin{pmatrix} 2249.81 & 7157.25 \\ 0.00 & 1.00 \end{pmatrix}$ | |
| | Hotelling | SPLL | KL | Hotelling | SPLL | KL | |
| MR | $\begin{pmatrix} 25.27 & 35.68 \\ 0.03 & 0.18 \end{pmatrix}$ | $\begin{pmatrix} 28.13 & 35.29 \\ 0.03 & 0.27 \end{pmatrix}$ | $\begin{pmatrix} 86.23 & 44.75 \\ 0.01 & 0.86 \end{pmatrix}$ | $\begin{pmatrix} 14.22 & 39.82 \\ 0.06 & 0.29 \end{pmatrix}$ | $\begin{pmatrix} 17.38 & 9.75 \\ 0.05 & 0.04 \end{pmatrix}$ | $\begin{pmatrix} 14.74 & 936.64 \\ 0.00 & 0.97 \end{pmatrix}$ | MR |
| ADWIN | $\begin{pmatrix} 1.01 & -4.61 \\ 0.98 & 0.00 \end{pmatrix}$ | $\begin{pmatrix} 1.01 & -4.64 \\ 0.98 & 0.00 \end{pmatrix}$ | $\begin{pmatrix} 1.01 & -2.06 \\ 0.52 & 0.02 \end{pmatrix}$ | $\begin{pmatrix} 8933.50 & 8933.50 \\ 0.00 & 1.00 \end{pmatrix}$ | $\begin{pmatrix} 1.01 & -4.61 \\ 0.98 & 0.00 \end{pmatrix}$ | $\begin{pmatrix} 1787.51 & 1782.89 \\ 0.78 & 0.20 \end{pmatrix}$ | ADWIN |
| CUSUM | $\begin{pmatrix} 56.66 & 37.97 \\ 0.01 & 0.18 \end{pmatrix}$ | $\begin{pmatrix} 273.13 & 32.80 \\ 0.00 & 0.52 \end{pmatrix}$ | $\begin{pmatrix} 3501.02 & 532.61 \\ 0.00 & 0.98 \end{pmatrix}$ | $\begin{pmatrix} 8933.50 & 8933.50 \\ 0.00 & 1.00 \end{pmatrix}$ | $\begin{pmatrix} 51.01 & 24.04 \\ 0.01 & 0.10 \end{pmatrix}$ | $\begin{pmatrix} 7193.20 & 8494.60 \\ 0.00 & 1.00 \end{pmatrix}$ | CUSUM |
| GMA | $\begin{pmatrix} 263.74 & 81.62 \\ 0.00 & 0.57 \end{pmatrix}$ | $\begin{pmatrix} 836.01 & 38.77 \\ 0.00 & 0.80 \end{pmatrix}$ | $\begin{pmatrix} 7909.90 & 5857.27 \\ 0.00 & 1.00 \end{pmatrix}$ | $\begin{pmatrix} 8933.50 & 8933.50 \\ 0.00 & 1.00 \end{pmatrix}$ | $\begin{pmatrix} 223.17 & 53.00 \\ 0.00 & 0.53 \end{pmatrix}$ | $\begin{pmatrix} 8498.15 & 8933.50 \\ 0.00 & 1.00 \end{pmatrix}$ | GMA |
| PH | $\begin{pmatrix} 4326.30 & 4056.73 \\ 0.00 & 0.97 \end{pmatrix}$ | $\begin{pmatrix} 4111.15 & 1834.51 \\ 0.00 & 0.94 \end{pmatrix}$ | $\begin{pmatrix} 8185.12 & 7593.98 \\ 0.00 & 1.00 \end{pmatrix}$ | $\begin{pmatrix} 8933.50 & 8933.50 \\ 0.00 & 1.00 \end{pmatrix}$ | $\begin{pmatrix} 649.86 & 65.24 \\ 0.00 & 0.88 \end{pmatrix}$ | $\begin{pmatrix} 8933.50 & 8933.50 \\ 0.00 & 1.00 \end{pmatrix}$ | PH |
| SEED | $\begin{pmatrix} 302.75 & 68.16 \\ 0.00 & 0.56 \end{pmatrix}$ | $\begin{pmatrix} 334.67 & 71.53 \\ 0.00 & 0.67 \end{pmatrix}$ | $\begin{pmatrix} 1359.02 & 512.10 \\ 0.00 & 0.92 \end{pmatrix}$ | $\begin{pmatrix} 8933.50 & 8933.50 \\ 0.00 & 1.00 \end{pmatrix}$ | $\begin{pmatrix} 646.84 & 116.10 \\ 0.00 & 0.84 \end{pmatrix}$ | $\begin{pmatrix} 4146.32 & 4064.88 \\ 0.00 & 0.99 \end{pmatrix}$ | SEED |
| | ADWIN | CUSUM | GMA | ADWIN | CUSUM | GMA | |
| | $\begin{pmatrix} 1.29 & -0.98 \\ 0.77 & 0.13 \end{pmatrix}$ | $\begin{pmatrix} 6800.12 & 3745.08 \\ 0.00 & 0.98 \end{pmatrix}$ | $\begin{pmatrix} 8933.50 & 8933.50 \\ 0.00 & 1.00 \end{pmatrix}$ | $\begin{pmatrix} 2.68 & 8.89 \\ 0.48 & 0.46 \end{pmatrix}$ | $\begin{pmatrix} 8898.50 & 8192.42 \\ 0.00 & 1.00 \end{pmatrix}$ | $\begin{pmatrix} 8933.50 & 8933.50 \\ 0.00 & 1.00 \end{pmatrix}$ | |
| | MR | PH | SEED | MR | PH | SEED | |
| | $\begin{pmatrix} 17.32 & -0.82 \\ 0.06 & 0.00 \end{pmatrix}$ | $\begin{pmatrix} 7693.29 & 3784.62 \\ 0.00 & 0.98 \end{pmatrix}$ | $\begin{pmatrix} 679.15 & 89.41 \\ 0.00 & 0.71 \end{pmatrix}$ | $\begin{pmatrix} 28.29 & 0.64 \\ 0.07 & 0.05 \end{pmatrix}$ | $\begin{pmatrix} 8926.50 & 8926.50 \\ 0.00 & 1.00 \end{pmatrix}$ | $\begin{pmatrix} 6339.61 & 5973.96 \\ 0.00 & 0.91 \end{pmatrix}$ | |

**Table 6.12:** The average difference made to each metric by applying the 90% / 10% PCA feature extraction on the three experiments. The values are averaged over all subjects and all parameter choices.

| | 90% | | | | 10% | | | |
|---|---|---|---|---|---|---|---|---|
| Detector | $\Delta_{ARL}$ | $\Delta_{TTD}$ | $\Delta_{FAR}$ | $\Delta_{MDR}$ | $\Delta_{ARL}$ | $\Delta_{TTD}$ | $\Delta_{FAR}$ | $\Delta_{MDR}$ |
| H | -2321.76● | -2925.28● | 0.00○ | -0.04● | -7442.33● | -7447.09● | 0.82○ | -0.83● |
| KL | -7047.52● | -5331.56● | 0.00○ | -0.02● | -4606.02● | -2075.70● | 0.00○ | -0.01● |
| SPLL | 6452.13○ | 6407.32○ | -0.76● | 0.74○ | -0.01○ | 0.13- | 0.01○ | 0.00○ |
| $\mu$ | -972.38 | -616.51 | -0.25 | 0.23 | -4016.12 | -3174.22 | 0.28 | -0.28 |
| $\sigma$ | 6218.08 | 5969.55 | 0.41 | 0.41 | 4049.79 | 4148.51 | 0.44 | 0.45 |
| ●−○ | 1 | 1 | -1 | 1 | 1 | 2 | -3 | 1 |

| | 90% | | | | 10% | | | |
|---|---|---|---|---|---|---|---|---|
| Detector | $\Delta_{ARL}$ | $\Delta_{TTD}$ | $\Delta_{FAR}$ | $\Delta_{MDR}$ | $\Delta_{ARL}$ | $\Delta_{TTD}$ | $\Delta_{FAR}$ | $\Delta_{MDR}$ |
| H-MR | -7421.57● | -7412.53● | 0.03○ | -0.66● | -7430.15● | -7404.63● | 0.05○ | -0.59● |
| H-ADWIN | -5060.94● | -5065.02● | 0.60○ | -0.60● | 2381.38○ | 2381.82○ | -0.22● | 0.23○ |
| H-CUSUM | -7394.90● | -7410.51● | 0.01○ | -0.65● | 0.00- | 0.00- | 0.00- | 0.00- |
| H-GMA | -7220.40● | -7355.65● | 0.00○ | -0.37● | 0.00- | 0.00- | 0.00- | 0.00- |
| H-PH | -3733.23● | -4116.79● | 0.00○ | -0.03● | 0.00- | 0.00- | 0.00- | 0.00- |
| H-SEED | -7109.37● | -7067.13● | 0.00○ | -0.35● | 31.72○ | 296.57○ | 0.00- | 0.00○ |
| KL-MR | -7348.61● | -7207.01● | 0.01○ | -0.12● | -7283.27● | -5335.17● | 0.00○ | -0.02● |
| KL-ADWIN | -5654.22● | -5658.41● | 0.50○ | -0.50● | -5060.36● | -5063.57● | 0.56○ | -0.56● |
| KL-CUSUM | -3486.22● | -4281.29● | 0.00○ | -0.02● | -1462.53● | -1033.37● | 0.00○ | -0.00● |
| KL-GMA | -1183.89● | -2058.00● | 0.00○ | -0.00● | -436.55● | -295.85● | 0.00○ | -0.00● |
| KL-PH | -672.71● | -882.28● | 0.00○ | -0.00● | -583.80● | -589.80● | 0.00- | -0.00● |
| KL-SEED | -6304.75● | -6624.18● | 0.00○ | -0.07● | -3788.52● | -3250.26● | 0.00○ | -0.01● |
| SPLL-MR | -3.72○ | 19.03○ | 0.01○ | 0.13○ | -14.00○ | -7.26● | 0.02○ | -0.05● |
| SPLL-ADWIN | -0.01○ | -0.13- | 0.01○ | -0.00● | -0.00○ | 0.19○ | 0.00○ | -0.00● |
| SPLL-CUSUM | 196.57○ | 31.73○ | -0.01● | 0.31○ | 11.17● | 7.92○ | -0.00● | 0.02○ |
| SPLL-GMA | 677.71○ | 26.48● | -0.00● | 0.30○ | 71.34○ | 17.74○ | -0.00● | 0.11○ |
| SPLL-PH | 3112.82○ | 1914.93○ | -0.00● | 0.10○ | 201.11○ | 167.18○ | -0.00● | 0.03○ |
| SPLL-SEED | -791.66● | -619.66● | 0.00○ | -0.10● | -384.56● | -455.09● | -0.00● | 0.03○ |
| $\mu$ | -3299.95 | -3542.58 | 0.06 | -0.15 | -1319.28 | -1142.42 | 0.02 | -0.05 |
| $\sigma$ | 4158.88 | 4121.59 | 0.22 | 0.36 | 3067.87 | 2922.09 | 0.17 | 0.23 |
| ●−○ | 8 | 11 | -12 | 10 | 3 | 3 | -3 | 3 |

| | 90% | | | | 10% | | | |
|---|---|---|---|---|---|---|---|---|
| Detector | $\Delta_{ARL}$ | $\Delta_{TTD}$ | $\Delta_{FAR}$ | $\Delta_{MDR}$ | $\Delta_{ARL}$ | $\Delta_{TTD}$ | $\Delta_{FAR}$ | $\Delta_{MDR}$ |
| MR | -28.35○ | -4.12● | -0.04● | -0.06● | -17.86● | -1.47● | -0.02● | 0.01○ |
| ADWIN | -0.30○ | -0.95○ | 0.10○ | -0.08● | 497.18○ | 499.21○ | -0.12● | 0.13○ |
| CUSUM | -658.39● | -2170.84● | 0.00○ | -0.01● | 876.96○ | 725.08○ | -0.00● | 0.01○ |
| GMA | 0.00- | 0.00- | 0.00- | 0.00- | 0.00- | 0.00- | 0.00- | 0.00- |
| PH | -609.23● | -1724.04● | 0.00- | -0.01● | 460.86○ | 734.40○ | 0.00- | 0.00○ |
| SEED | -4390.57● | -4658.43● | -0.00● | -0.15● | 282.18○ | 484.87○ | -0.00● | 0.02○ |
| $\mu$ | -947.81 | -1426.40 | 0.01 | -0.05 | 349.89 | 407.02 | -0.02 | 0.03 |
| $\sigma$ | 2019.63 | 2441.30 | 0.06 | 0.08 | 609.96 | 745.88 | 0.06 | 0.06 |
| ●−○ | 1 | 3 | 0 | 5 | -3 | -3 | 4 | -5 |

Experiment 3 demonstrates that a simple and constant-time ensemble of moving range charts performs very strongly compared to the multivariate detectors. We also see that MR, ADWIN and SEED appear to be highly tunable in these ensembles.

Experiment 4 extracted sets of features which were the principal components accounting for 90% and 10% of the variance respectively. In Chapter 3

**Table 6.13:** The global top 20 performers in the ARL/TTD and FAR/MDR spaces.

| Detector | ARL | TTD | Detector | FAR | MDR |
|---|---|---|---|---|---|
| PCA10-MR-10 | 15.80 | 0.87 | PCA90-MR-20 | 0.03 | 0.00 |
| PCA90-MR-20 | 20.80 | -0.65 | PCA90-MR-25 | 0.02 | 0.02 |
| PCA90-MR-15 | 16.16 | -0.81 | PCA90-MR-15 | 0.04 | 0.00 |
| PCA90-MR-1 | 10.50 | -1.20 | PCA10-MR-10 | 0.04 | 0.01 |
| PCA90-MR-5 | 10.50 | -1.20 | PCA90-SPLL-MR-W5 | 0.04 | 0.02 |
| PCA90-MR-10 | 10.50 | -1.20 | PCA10-SPLL-MR-W10 | 0.04 | 0.02 |
| PCA10-MR-5 | 8.65 | -0.73 | PCA10-SPLL-MR-W5 | 0.04 | 0.02 |
| PCA10-MR-1 | 2.91 | -2.09 | PCA90-MR-1 | 0.07 | 0.00 |
| SPLL-ADWIN-W30 | 0.92 | -3.94 | PCA90-MR-5 | 0.07 | 0.00 |
| SPLL-W50 | 0.90 | -3.86 | PCA90-MR-10 | 0.07 | 0.00 |
| SPLL-W40 | 0.90 | -3.90 | PCA10-SPLL-MR-W25 | 0.04 | 0.03 |
| SPLL-W45 | 0.90 | -3.89 | PCA10-MR-5 | 0.08 | 0.00 |
| SPLL-W35 | 0.89 | -3.95 | PCA10-SPLL-MR-W20 | 0.04 | 0.04 |
| SPLL-W30 | 0.89 | -3.99 | PCA10-SPLL-MR-W15 | 0.04 | 0.04 |
| PCA10-SPLL-W50 | 0.89 | -3.71 | PCA10-HOTELLING-MR-W5 | 0.06 | 0.02 |
| SPLL-W25 | 0.89 | -4.04 | PCA10-MR-15 | 0.02 | 0.06 |
| PCA10-SPLL-W45 | 0.88 | -3.81 | PCA10-SPLL-MR-W45 | 0.04 | 0.05 |
| PCA10-SPLL-W40 | 0.88 | -3.60 | PCA10-SPLL-MR-W35 | 0.04 | 0.05 |
| SPLL-W20 | 0.88 | -4.08 | SPLL-CUSUM-W25 | 0.02 | 0.08 |
| PCA10-SPLL-W35 | 0.88 | -3.74 | PCA90-HOTELLING-MR-W10 | 0.03 | 0.07 |
| SPLL-W15 | 0.88 | -4.07 | PCA10-SPLL-MR-W40 | 0.04 | 0.05 |

it was suggested that in the case of random change, the least variant components should be retained. In this case, the most variant components appear to be the better choice. There are several explanations for this. Firstly, the changes are not random noise, but stable reoccurring concepts. Secondly, given that we hypothesise the largest changes in colour should be indicative of blinks, this means that the most useful features are also likely to have high variance relative to the background. Thirdly, counts in the histogram are not independent. Assuming constant illumination and no significant changes other than blinks in the ROI, changes in the histogram over time will be a roughly zero-sum game. A reduction in one bucket's colour count will be reflected with an increase in others, creating a higher proportion of useful variant features. This relationship is visible to a varying extent in Figure 6.14.

The 90% features were particularly successful in improving the relevance of the Hotelling and KL statistics, as can be seen in Figures 6.11 and 6.16 respectively. The initial poor performance of Hotelling and KL, the relatively

small number of features in this set and the above arguments regarding the features suggest that SPLL is better at coping with noisy data. The features were also highly successful in decreasing missed detections, especially for the ensembles where the effect on false alarm rates was insignificant compared with the chained detectors.

As discussed in Section 6.5.1, the reported false alarm rates for all detectors are higher than necessary because in all these experiments the detectors were not reset when a change was detected. However, this allowed the results to clearly reflect which detectors were suitable.

A selection of the results compare favourably to other examples in the blink detection literature. Chau and Betke [29] report an average accuracy of 96%. Królak and Strumiłło [93] report an accuracy of 95.3% in good lighting condition. Bacivarov et al. [7] report an accuracy of 91%, Bhaskar et al. [13] 97% and Chen et al. [30] 96.88%. Of the reviwed papers, only Chau and Betke report their false positive rate[6]. They report 173 false positives for 2288 blinks analysed, better than was achieved here. For a better comparison, Table 6.14 shows a global top 20 methods in the FAR/MDR space only considering those with less than 1% FAR. As an example of the gap, PCA10-SPLL-CUSUM-W10, posted on average 99 false positives per run, with 88% accuracy. However, this is likely significantly contributed to by repeat detections as discussed earlier. A repeat of the experiment with resetting would hopefully bring the best performers up to a comparable rate to the other approaches in the literature.

The feature extraction process as discussed in Section 6.4.2 is clearly quite naïve compared to alternative vision-based pipelines in the literature such as Chen et al. [30]. It is very encouraging that several of the methods tested here approach or exceed their reported accuracy of 96.88% despite a very

[6]To prevent misconception, the reader is reminded of the difference between the false alarm rate and false positive rate. The former is false alarms as a proportion of all observations, i.e. video frames. The latter is false alarms as a proportion of true positives, i.e. blinks.

**Table 6.14:** Top 20 detectors with a less than 1% false alarm rate.

| Detector | FAR | MDR |
|---|---|---|
| PCA10-SPLL-CUSUM-W10 | 0.0099 | 0.1177 |
| PCA10-SPLL-CUSUM-W5 | 0.0091 | 0.1213 |
| PCA90-H-CUSUM-W15 | 0.0091 | 0.1706 |
| PCA90-H-CUSUM-W10 | 0.0095 | 0.1917 |
| PCA90-H-CUSUM-W30 | 0.0092 | 0.1952 |
| PCA90-H-CUSUM-W25 | 0.0090 | 0.2273 |
| PCA10-MR-25 | 0.0081 | 0.2355 |
| PCA90-H-GMA-W5 | 0.0070 | 0.2620 |
| PCA90-SPLL-CUSUM-W15 | 0.0021 | 0.2823 |
| SPLL-GMA-W35 | 0.0040 | 0.2836 |
| PCA90-SPLL-CUSUM-W5 | 0.0024 | 0.2986 |
| SPLL-GMA-W25 | 0.0030 | 0.3116 |
| SPLL-GMA-W10 | 0.0021 | 0.3219 |
| SPLL-GMA-W20 | 0.0028 | 0.3325 |
| SEED-1 | 0.0057 | 0.3333 |
| SPLL-GMA-W30 | 0.0032 | 0.3402 |
| SPLL-GMA-W5 | 0.0015 | 0.3412 |
| PCA90-H-SEED-W35 | 0.0026 | 0.3448 |
| SPLL-GMA-W40 | 0.0034 | 0.3477 |
| SPLL-GMA-W15 | 0.0025 | 0.3541 |
| SPLL-GMA-W50 | 0.0038 | 0.3606 |

simplistic feature extraction process. It is highly likely that further work into a more advanced feature extraction process would yield a better end result.

There is a remarkable diversity of approaches to this problem within the reviewed literature in Section 6.2. Many of the reviewed works used bespoke thresholding schemes for their complex feature extraction pipelines. The results here appear to lend weight to the hypothesis that change detection methods are a feasible alternative solution to these bespoke thresholding schemes. Secondly, the poor performance of the multivariate detectors in the baseline experiment, and the subsequent improvement seen in Sections 6.6, 6.7 and 6.8 demonstrate the feasibility of the techniques presented in Chapters 3, 4 and 5.

The experiments in this chapter were designed to fill the experimental gaps in the hypotheses left by the previous chapters. Continuing from the conclusion of Chapter 3, in Section 6.8 the value of PCA is demonstrated in the fully unsupervised context, emphasising the context-free nature of the technique. From

the conclusion of Chapter 4, Section 6.6 presents a comprehensive before-and-after study of 3 multivariate detectors chained with 6 univariate detectors, and compared to the baseline established in Section 6.5. The performance of the ensembles in Sections 6.7 and 6.8 lend further support to hypothesis (5).

It should be noted that the assumption from hypothesis (1) that data are *i.i.d* are probably not met in this chapter. The data is not sampled, and streaming blink data is likely to exhibit a high degree of temporal dependence. However, experiments from previous chapters where sampling was used have met this qualification.

# Chapter 7

# Meander: A Java Library for Change Detection Pipelines and Change Stream Generation

Recall Figure 2.10. The observation that many change detectors are complex pipelines of operations along with a necessity to speed up experiments for this thesis led to the development of a Java library. The objectives of this library were as follows:

- Break down a number of change detection approaches into their fundamental components.
- Provide a type-safe framework for the combination of these components.
- Components should be thread-safe and significantly faster than the equivalent MATLAB implementations.

It was named ’Meander’ as a synonym for ’Changing Stream’. The library is built on Java 8 to take advantage of the streams API. It is divided into two modules.

`meander-core` Change stream generation, evaluation and interfaces.

`meander-detectors` Components for functional change detection.

This was based on the recognition that generating changing data streams and evaluation of change detection approaches may be valuable on its own for people wishing to evaluate their own change detectors.

## 7.1 Motivation

Consider a change detector as a function $CD(\vec{x}_t) \to \{0,1\}$ (or $CD(W_t) \to \{0,1\}$ if it takes a window) which maps an input to a binary space. We see this function as a black box – examples or batches are taken from the data stream, passed into our change detector, and it indicates whether there has been change in the context of those examples. In fact, we can decompose this black box into a pipeline of operations, as depicted in Figure 2.10. As a minimal example, there must be some threshold to arrive at a binary output, so the change detection function is at least a composition $CD(\vec{x}_t) = D \circ C(\vec{x}_t)$ of:

$$\text{A criterion function: } \begin{cases} C(\vec{x}_t) \to \psi_t \\ C(W_t) \to \psi_t \\ C(W_{1,t}, W_{2,t}) \to \psi_t \end{cases} \tag{7.1}$$

$$\text{A decision function: } D(\psi_t) \to \{0,1\} \tag{7.2}$$

where $\psi_t$ represents the information required to make a decision at index $t$. This state is typically a real number change detection criterion $\psi_t \in \mathbb{R}$. Depending on whether we have data management, forgetting, preprocessing or modelling steps, this may be a composition of arbitrarily more functions.

To see why this perspective is important, we will consider how change detectors are contributed to MOA [19, 21]. The focus of MOA is data stream mining rather than change detection, although it offers a `ChangeDetector` interface

contract for use in adaptive learning. The important methods are shown in Figure 7.1.

```
public interface ChangeDetector extends OptionHandler {
        // Other methods redacted...

        public void input(double inputValue);

        public boolean getChange();
}
```

**Figure 7.1:** Important methods in MOA `ChangeDetector` interface contract from `moa.classifiers.core.driftdetection`.

Users of the interface provide the next value in the stream to `void input(double inputValue)`, and find out whether change was detected by calling `boolean getChange()`. This pattern presents two impediments. Firstly, it is implicitly assumed that input will be `double` valued, i.e. univariate. Secondly, this interface encapsulates the whole pipeline from input to boolean output. Each implementer must wholly manage their data management, forgetting, preprocessing and so on. This makes it very cumbersome to investigate the effects of changing thresholds, adding preprocessing steps, or replacing modules. In effect, this is the 'black box' approach that was mentioned at the beginning of this section.

The motivation of Meander is to encourage component-first development, such that these components can be replaced, reordered or appended to without having to rewrite any code. This in turn should reduce the time taken to prototype ideas in unsupervised change detection.

## 7.2 Change Detection

The fundamental component is called a `Pipe`. A pipe encapsulates a function mapping from an input type to an output type. Using Java's generics, example signatures for univariate and multivariate detectors respectively are:

- `Pipe<Double,Boolean>`
- `Pipe<Double[],Boolean>`

However in principle a detector could map from any input type to a boolean.

Pipes are composable as demonstrated in Figure 7.2. A pipe that maps `Double[]` to `Boolean` might be composed of multiple pipes, the output of each connected to the input of the next.

```
1 Pipe<Double[],Double> criterion = ...
2 Pipe<Double, Boolean> decision  = ...
3
4 Pipe<Double[], Boolean> detector = criterion.then(decision);
5
6 Double[] example = ...
7
8 Boolean wasChangeDetected = detector.execute(example);
```

**Figure 7.2:** Type-constrained functional composition of `Pipe` objects.

Each execution of a pipe is intended to map a single stream example to a detection or non-detection of change. Stateful components like sliding windows block further execution of the pipe until they have collected enough examples to pass on.

Figures 7.3 and 7.4 demonstrate a typical use case. In Figure 7.3, we create the canonical Hotelling detector, as described earlier in this chapter. It takes a pair of windows, calculates the $T^2$ statistic between them, and takes the complementary probability on an F-distribution cdf with the appropriate degrees of freedom. If there were simply a class implementing the whole process for the Hotelling detector, it would most likely require cumbersome modifications or duplication of code if we wanted to investigate the effects of PCA, for example. By breaking detectors down into pipelines of operations, we are free to insert or replace steps as required with minimal re-engineering of existing steps.

In Figure 7.4, we insert a step which performs a PCA transformation on the sliding windows, and we have replaced the static threshold with a moving

```
Pipe<Double[], Boolean> tsq = new WindowPairPipe(100)
        .then(new TSquared())
        .then(new FWithDF().complementary())
        .then(Threshold.lessThan(0.05));
```

**Figure 7.3:** The pipeline for the Hotelling $T^2$ detector.

```
Pipe<Double[], Boolean> tsqPCAMR = new WindowPairPipe(100)
        .then(new PCAWindowPairTransform())
        .then(new TSquared())
        .then(new FWithDF().complementary())
        .then(new MovingRange())
        .then(new MovingRangeThreshold());
```

**Figure 7.4:** A pipeline with PCA for the Hotelling $T^2$ detector.

range control chart. We can do the latter because the second to last step in the pipeline, the cdf, produces a `Double` and we need to arrive at a `Boolean`. An important observation is that since any univariate detector fulfils a mapping from `Double` to `Boolean`, we could insert it in lieu of a threshold.

## 7.3 Ensembles

Ensembles can be created through the combination of pipes. Three types of ensembles are currently supported.

`UnivariateEnsemble` Aggregates the votes of multiple detectors matching the signature `Pipe<Double,Boolean>`. Combines univariate detectors over univariate data.

`MultivariateEnsemble` Aggregates the votes of multiple detectors matching the signature `Pipe<Double[],Boolean>`. Combines multivariate detectors over multivariate data.

`SubspaceEnsemble` Aggregates the votes of multiple detectors matching the signature `Pipe<Double,Boolean>`, but expects multivariate data. Com-

bines univariate detectors over multivariate data. Each detector is assigned a feature (or subspace) of the input data to monitor.

An ensemble in Meander satisfies the interface `Pipe<T,Boolean[]>`, mapping from an input to an array of votes. The fusion of these decisions is delegated to the next step in the pipeline. Currently there are two fusion types supported.

`SimpleMajority` Given a threshold between 0 and 1, signals true if at least that percentage of the detectors signal. Implements `Pipe<Boolean[], Boolean>`.

`DecayingMajority` Takes a threshold between 0 and 1. Takes a sum of the votes at each time point, incrementally downweighting old votes according to a decay function. Designed to account for detectors whose votes do not exactly synchronise over time. Implements `Pipe<Boolean[], Double>`.

`DecayingMajority` maps to a statistic, so this can subsequently be thresholded to a desired agreement level, or by any univariate detector.

Figure 7.5 demonstrates the process of creating an ensemble.

```
1  Pipe<Double,Boolean> mr = new MovingRange()
2          .then(new MovingRangeThreshold());
3  Pipe<Double,Boolean> cusum = new CUSUM()
4          .then(Threshold.greaterThan(3));
5  Pipe<Double,Boolean> imr = new MovingRange()
6          .then(new IndividualsMovingRangeThreshold());
7
8  Pipe<Double,Boolean> ensemble =
9          new UnivariateEnsemble(mr, cusum, imr)
10         .then(new SimpleMajority(0.5))
```

**Figure 7.5:** Creating a univariate simple majority ensemble from univariate detectors.

## 7.4 Change Stream Generation

The library also includes a framework for the evaluation of these detector pipelines on data streams. WEKA-style `.arff` files can be streamed verbatim, or sampled to produce arbitrary length streams with specific change points.

```
1  // Stream of examples verbatim from file
2  Stream<Example> verbatim = ArffStream.of("abalone.arff");
3
4  // Stream of sampled examples with artificial change
5  // between specified prior probabilities.
6  ChangeStreamBuilder builder = ChangeStreamBuilder
7          .fromArff("abalone.arff")
8          .withUniformPriors().fromStart()
9          .withPriors(1.0, 0.0, 0.0)
10         .transition(new AbruptTransition(2500))
11         .withPriors(0.0, 1.0, 0.0)
12         .transition(new LogisticTransition(5000,5100))
13         .withPriors(0.0, 0.0, 1.0)
14         .transition(new AbruptTransition(7500));
15
16 // Artificial streams sample from the dataset, so
17 // we can draw an arbitrary number of examples.
18 Stream<Example> artificial = builder.build().limit(10000);
```

**Figure 7.6:** Fluent API to provide Java 8 streams from `.arff` files both verbatim and with artificial change.

Given an `.arff` file to sample, we can produce a stream with artificial changes using the framework described by Narasimhamurthy and Kuncheva [134] and Bifet et al. [20]. Three types of transition are supported.

`AbruptTransition` Transitions between data sources instantly.

`LinearTransition` Gradual transition as in Figure 2.3.

`LogisticTransition` Gradual transition using logistic function as suggested by Bifet et al. [20].

The second example of Figure 7.6 shows how transitions can be created between defined prior probabilities. In this instance, we transition cleanly from one class to another, but any mixture of priors is supported.

## 7.5 Evaluation

With a suitable data stream and a change detector, the evaluation API can be invoked as in Figure 7.7. This invokes the detector on each example in the change stream, and records the positions of positive detections.

```
1  Pipe<Double[], Boolean> detector = ...
2  Stream<Example> stream = ...
3
4  Evaluator evaluator = new SequenceEvaluator();
5  Evaluation results = evaluator.evaluate(detector, stream);
6
7  double arl,ttd,far,mdr;
8  arl = results.getArl();
9  ttd = results.getTtd();
10 far = results.getFar();
11 mdr = results.getMdr();
```

**Figure 7.7:** The evaluation API providing the four performance metrics.

Two evaluators are offered. Each choice of evaluator reflects a differing change *context*.

`SequenceEvaluator` Interprets transition start and end points as changes that must be detected.

`ShortConceptsEvaluator` Only interprets transition start points as changes.

The `ShortConceptsEvaluator` is geared towards very short lived changes, where the objective is closer to anomaly detection.

## 7.6 Summary

This framework allows a technical user to quickly prototype new components and assess the impact of new pipeline steps. It was used to run the experiments in Chapter 6 and an early iteration was used to run the experiments in Chapter 5. The code is open source, hosted on GitHub[1] at the time of writing. The code is made available permissively, under the Apache License 2.0.

[1] `https://github.com/wfaithfull/meander`

# Chapter 8

# Conclusion

## 8.1 Summary of Work

Change Detection and related fields have seen considerable advances in the last few decades, and there exists now a suite of approaches that can be applied to the unsupervised problem, univariate and multivariate. The objective of this thesis has been to develop general techniques that can leverage these existing approaches to improve multivariate change detection performance.

It has been demonstrated that in the case of multivariate unsupervised change detection, general purpose steps can be appended to the pipeline, offering improved outcomes without any modification of the underlying approaches. In Chapter 3, we take the least variant principal components as the features for change detection, discuss in what context this is preferable and show that this improves performance on most of the datasets tested. Chapter 4 suggests an alternative means of thresholding the statistics generated by multivariate change detectors, chaining the statistic to a univariate change detector (in this case a control chart). This was tested on a challenging dataset of facial expressions. The results demonstrate the approach relative to bootstrapping. Chapter 5 takes existing univariate detectors and builds them into subspace ensembles which can act as multivariate detectors. Over a selection of 96 datasets, these ensembles frequently outperformed established

multivariate change detectors. Chapter 6 performed a baseline experiment in endogenous eye blink detection, and subsequently demonstrated the viability of the techniques from the previous chapters.

## 8.2 Future Work

Chapter 3 raises several avenues of future work; What is the relationship of window size to the benefits of PCA for change detection? Is there a "middle part" of principal components which are both relatively important and relatively sensitive to change? How much computational complexity is added by the feature extraction step? Chapter 3 also has relevance to Chapter 5. It can be noted that a stated limitation of the novel subspace ensemble technique was that it assumes independence of the features and as such cannot take into account correlations. Firstly, it would be beneficial to establish the context and extent of the effect of feature independence on multivariate change detection. Secondly, assuming correlation is significant, one would expect that the ensembles are likely to benefit from PCA transformed data. The principal components are by necessity linearly uncorrelated. Important contextual information contained in the covariance of the original data is partially preserved as it will affect the parameters of the transformation. Empirically, this combination was effective in Chapter 6.

The ensembles in Chapters 5 and 6 were fused by simple majority. Although techniques such as bagging and boosting are not possible in the unsupervised setting, there is an active area of research into unsupervised ensemble combination techniques [185, 78]. Within this thesis ensembles were consistently among the best performers, so applying more sophisticated combination techniques is an interesting research direction.

Finally, there is increasing interest in deep learning across a multitude of disciplines at present. This includes a small selection of work in change detection [178, 133]. Recent work on adversarial autoencoders [120] appears promising as a means to model even very complex distributions from minimal samples using adversarial training.

## 8.3 Publications Relating to the Thesis

- Kuncheva, L.I. and Faithfull, W.J., 2012, November. Pca feature extraction for change detection in multidimensional unlabelled streaming data. In Pattern Recognition (ICPR), 2012 21st International Conference on (pp. 1140-1143). IEEE.
- Kuncheva, L.I. and Faithfull, W.J., 2014. PCA feature extraction for change detection in multidimensional unlabeled data. IEEE transactions on neural networks and learning systems, 25(1), pp.69-80.
- Faithfull, W.J. and Kuncheva, L.I., 2014. On Optimum Thresholding of Multivariate Change Detectors. In Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR) (pp. 364-373). Springer, Berlin, Heidelberg.
- Faithfull, W.J., Rodríguez, J.J. and Kuncheva, L.I., 2019. Combining univariate approaches for ensemble change detection in multivariate data. Information Fusion, 45, pp.202-214.

# References


[1] R. P. Adams and D. J. MacKay, ‘Bayesian online changepoint detection’, *arXiv preprint arXiv:0710.3742*, 2007 (p. 19).

[2] D. Agarwal, ‘An Empirical Bayes approach to detect anomalies in dynamic multidimensional arrays’, in *Proceedings - IEEE International Conference on Data Mining, ICDM*, 2005, pp. 26–33 (pp. 1, 42, 92).

[3] J. Ahlberg, ‘Candide-3-an updated parameterised face’, 2001 (p. 84).

[4] C. Alippi, G. Boracchi, D. Carrera and M. Roveri, ‘Change detection in multivariate datastreams: Likelihood and detectability loss’, in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, AAAI Press, 2016, pp. 1368–1374 (pp. 93, 109, 112, 114).

[5] C. Alippi, G. Boracchi and M. Roveri, ‘Hierarchical change-detection tests’, *IEEE transactions on neural networks and learning systems*, vol. 28, no. 2, pp. 246–258, 2017 (p. 94).

[6] S. Aminikhanghahi and D. J. Cook, ‘A survey of methods for time series change point detection’, *Knowledge and information systems*, vol. 51, no. 2, pp. 339–367, 2017 (pp. 16, 22–24).

[7] I. Bacivarov, M. Ionita and P. Corcoran, ‘Statistical models of appearance for eye tracking and eye-blink detection and measurement’, *IEEE transactions on consumer electronics*, vol. 54, no. 3, 2008 (pp. 119, 121, 159).

[8] M. Baena-García, J. del Campo Ávila, R. Fidalgo, A. Bifet, R. Gavalda and R. Morales-Bueno, ‘Early Drift Detection Method’, *Fourth international workshop on knowledge discovery from data streams*, vol. 6, pp. 77–86, 2006 (pp. 37, 48, 94, 95).

[9] M. Basseville and I. V. Nikiforov, *Detection of abrupt changes: theory*

*and application*. Prentice Hall Englewood Cliffs, 1993, vol. 104 (pp. 7, 15, 19, 21, 22, 24, 47).

[10] C. Beaulieu, J. Chen and J. L. Sarmiento, 'Change-point analysis as a tool to detect abrupt climate variations', *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 370, no. 1962, 2012 (p. 15).

[11] S. Benedetto, M. Pedrotti, L. Minin, T. Baccino, A. Re and R. Montanari, 'Driver workload and eye blink duration', *Transportation research part F: traffic psychology and behaviour*, vol. 14, no. 3, pp. 199–208, 2011 (p. 125).

[12] I. Ben-Gal, 'Outlier detection', in *Data mining and knowledge discovery handbook*, Springer, 2009, pp. 117–130 (p. 16).

[13] T. N. Bhaskar, F. T. Keat, S. Ranganath and Y. V. Venkatesh, 'Blink detection and eye tracking for eye localization', in *TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region*, vol. 2, Oct. 2003, 821–824 Vol.2. DOI: 10.1109/TENCON.2003.1273293 (pp. 120, 159).

[14] A. Bifet, 'Adaptive learning and mining for data streams and frequent patterns', *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 55–56, 2009 (p. 14).

[15] A. Bifet, E. Frank, G. Holmes and B. Pfahringer, 'Ensembles of restricted hoeffding trees', *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 3, no. 2, p. 30, 2012 (p. 94).

[16] A. Bifet, E. Frank, G. Holmes and B. Pfahringer, 'Accurate ensembles for data streams: Combining restricted hoeffding trees using stacking', in *Proceedings of 2nd Asian Conference on Machine Learning*, 2010, pp. 225–240 (p. 94).

[17] A. Bifet and R. Gavalda, 'Learning from time-changing data with adaptive windowing', in *Proceedings of the 2007 SIAM International Conference on Data Mining*, SIAM, 2007, pp. 443–448 (pp. 1, 19, 20, 33, 39, 86, 95).

[18] A. Bifet and R. Gavaldà, ‘Adaptive learning from evolving data streams’, in *International Symposium on Intelligent Data Analysis*, Springer, 2009, pp. 249–260 (pp. 14, 49).

[19] A. Bifet, G. Holmes, R. Kirkby and B. Pfahringer, ‘MOA Massive Online Analysis’, *Journal of Machine Learning Research*, vol. 11, pp. 1601–1604, 2011 (pp. 91, 95, 163).

[20] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby and R. Gavaldà, ‘New ensemble methods for evolving data streams’, in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2009, pp. 139–148 (pp. 9, 48–50, 94, 168).

[21] A. Bifet, J. Read, B. Pfahringer, G. Holmes and I. Žliobaitė, ‘CD-MOA: Change detection framework for massive online analysis’, *Advances in Intelligent Data Analysis XII*, pp. 92–103, 2013 (pp. 91, 95, 163).

[22] R. B. Blazek, H. Kim, B. Rozovskii and A. Tartakovsky, ‘A novel approach to detection of “denial–of–service” attacks via adaptive sequential and batch–sequential change–point detection methods’, in *Workshop on Information Assurance and Security*, vol. 1, 2001, p. 0930 (p. 15).

[23] D. A. Blythe, P. Von Bunau, F. C. Meinecke and K.-R. Muller, ‘Feature extraction for change-point detection using stationary subspace analysis’, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 4, pp. 631–643, 2012 (p. 55).

[24] A. Bouchachia, ‘On the scarcity of labeled data’, in *Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on*, IEEE, vol. 1, 2005, pp. 402–407 (p. 10).

[25] G. Bradski, A. Kaehler and V. Pisarevsky, ‘Learning-based computer vision with intel’s open source computer vision library.’, *Intel Technology Journal*, vol. 9, no. 2, 2005 (p. 119).

[26] D. Brauckhoff, K. Salamatian and M. May, ‘Applying pca for traffic

anomaly detection: Problems and solutions', in *INFOCOM 2009, IEEE*, IEEE, 2009, pp. 2866–2870 (pp. 55, 56).

[27] D. Brzezinski and J. Stefanowski, 'Ensemble diversity in evolving data streams', in *International Conference on Discovery Science*, Springer, 2016, pp. 229–244 (p. 97).

[28] V. Chandola, A. Banerjee and V. Kumar, 'Anomaly detection: A survey', *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009 (pp. 16, 17, 22–24).

[29] M. Chau and M. Betke, 'Real time eye tracking and blink detection with usb cameras', Boston University Computer Science Department, Tech. Rep., 2005 (pp. 117, 119, 120, 159).

[30] B.-C. Chen, P.-C. Wu and S.-Y. Chien, 'Real-time eye localization, blink detection, and gaze estimation system without infrared illumination', in *Image Processing (ICIP), 2015 IEEE International Conference on*, IEEE, 2015, pp. 715–719 (pp. 119, 120, 159).

[31] J. L. Crowley and F. Berard, 'Multi-modal tracking of faces for video communications', in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, IEEE, 1997, pp. 640–645 (pp. 119, 120).

[32] T. Danisman, I. M. Bilasco, C. Djeraba and N. Ihaddadene, 'Drowsy driver detection system using eye blink patterns', in *2010 International Conference on Machine and Web Intelligence*, Oct. 2010, pp. 230–233. DOI: 10.1109/ICMWI.2010.5648121 (pp. 119, 121).

[33] T. Dasu, S. Krishnan, S. Venkatasubramanian and K. Yi, 'An information-theoretic approach to detecting changes in multi-dimensional data streams', in *Proc. Symp. on the Interface of Statistics, Computing Science, and Applications*, Citeseer, 2006 (pp. 30, 41, 45, 48, 60, 76, 79, 80, 82).

[34] S. Delany, P. Cunningham and A. Tsymbal, 'A case-based technique for tracking concept drift in spam filtering', *Knowledge-based systems*,

2005 (pp. 13, 14, 48).

[35] M. Divjak and H. Bischof, ‘Eye blink based fatigue detection for prevention of computer vision syndrome.’, in *IAPR Conference on Machine Vision Applications*, 2009, pp. 350–353 (p. 119).

[36] P. Domingos and G. Hulten, ‘Mining high-speed data streams’, in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2000, pp. 71–80 (p. 37).

[37] A. Dries and U. Rückert, ‘Adaptive Concept Drift Detection’, *Statistical Analysis and Data Mining*, vol. 2, no. 5-6, pp. 311–327, 2009 (p. 48).

[38] L. Du, Q. Song, L. Zhu and X. Zhu, ‘A selective detector ensemble for concept drift detection’, *The Computer Journal*, vol. 58, no. 3, pp. 457–471, 2014 (pp. 20, 94).

[39] R. O. Duda, P. E. Hart and D. G. Stork, *Pattern classification*. John Wiley & Sons, 2012 (p. 11).

[40] G. J. Edwards, T. F. Cootes and C. J. Taylor, ‘Face recognition using active appearance models’, in *European conference on computer vision*, Springer, 1998, pp. 581–595 (p. 84).

[41] R. Elwell and R. Polikar, ‘Incremental learning of concept drift in non-stationary environments’, *IEEE Transactions on Neural Networks*, 2011 (pp. 7, 13, 46, 48, 49).

[42] P. Evangelista and M. Embrechts, ‘Taming the curse of dimensionality in kernels and novelty detection’, *Applied soft computing*, 2006 (pp. 92, 93, 95, 97, 112–114).

[43] W. Fan, ‘Systematic data selection to mine concept-drifting data streams’, in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2004, pp. 128–137 (p. 48).

[44] T. Fawcett and F. Provost, ‘Adaptive fraud detection’, *Data mining and knowledge discovery*, 1997 (p. 15).

[45] M. Fernández-Delgado, E. Cernadas, S. Barro and D. Amorim, ‘Do we need hundreds of classifiers to solve real world classification problems’,

*Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3133–3181, 2014 (p. 116).

[46] C. Ferri, J. Hernández-Orallo and P. A. Flach, 'A coherent interpretation of auc as a measure of aggregated classification performance', in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 657–664 (p. 68).

[47] A. Fogelton and W. Benesova, 'Eye blink detection based on motion vectors analysis', *Computer Vision and Image Understanding*, vol. 148, pp. 23–33, 2016 (p. 120).

[48] I. Frias-Blanco, J. del Campo-Avila, G. Ramos-Jimenez, R. Morales-Bueno, A. Ortiz-Diaz and Y. Caballero-Mota, 'Online and non-parametric drift detection methods based on Hoeffding's bounds', *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 3, pp. 810–823, 2015 (pp. 37, 94, 95).

[49] K. Fukuda, 'Eye blinks: New indices for the detection of deception', *International Journal of Psychophysiology*, vol. 40, no. 3, pp. 239–245, 2001 (p. 117).

[50] M. Gaber and P. Yu, 'Classification of changes in evolving data streams using online clustering result deviation', 2006 (p. 41).

[51] J. Gama and P. Rodrigues, 'Data stream processing', *Learning from Data Streams*, 2007 (p. 6).

[52] J. Gama, R. Sebastião and P. Rodrigues, 'On evaluating stream learning algorithms', *Machine learning*, 2013 (pp. 17, 47).

[53] J. Gama, *Knowledge discovery from data streams*. Chapman & Hall/CRC, 2010, p. 237, ISBN: 1439826129 (pp. 1, 7, 8, 16, 19, 31, 36, 38, 45).

[54] J. Gama, P. Medas, G. Castillo and P. Rodrigues, 'Learning with drift detection', *Advances in Artificial Intelligence – SBIA 2004*, pp. 286–295, 2004 (pp. 6, 14, 33, 36, 37, 46, 48, 86, 94, 95).

[55] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy and A. Bouchachia, 'A survey on concept drift adaptation', *ACM Computing Surveys*, vol. 46,

no. 4, pp. 1–37, 2014 (pp. 7, 8, 11, 13, 17, 20, 22, 24, 26, 27, 34, 36, 106).

[56] J. Gao, W. Fan, J. Han and P. Yu, 'A general framework for mining concept-drifting data streams with skewed distributions', *Proceedings of the 2007 SIAM International*, 2007 (pp. 12, 13).

[57] J. W. Gardner, 'Detection of vapours and odours from a multisensor array using pattern recognition part 1. principal component and cluster analysis', *Sensors and Actuators B: Chemical*, vol. 4, no. 1-2, pp. 109–115, 1991 (pp. 55, 56).

[58] B. K. Ghosh and P. K. Sen, *Handbook of sequential analysis*. CRC Press, 1991 (pp. 16, 19).

[59] M. Girshick and H. Rubin, 'A Bayes approach to a quality control model', *The Annals of mathematical statistics*, 1952 (pp. 18, 19).

[60] H. M. Gomes, J. P. Barddal, F. Enembreck and A. Bifet, 'A survey on ensemble learning for data stream classification', *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, p. 23, 2017 (pp. 20, 94, 97).

[61] P. R. Goulding, B. Lennox, D. J. Sandoz, K. J. Smith and O. Marjanovic, 'Fault detection in continuous processes using multivariate statistical methods', *International journal of systems science*, vol. 31, no. 11, pp. 1459–1471, 2000 (pp. 55, 56).

[62] J. Gupchup, A. Terzis, R. Burns and A. Szalay, 'Model-based event detection in wireless sensor networks', *arXiv preprint arXiv:0901.3923*, 2009 (pp. 55, 56).

[63] M. Gupta, J. Gao, C. C. Aggarwal and J. Han, 'Outlier detection for temporal data: A survey', *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 9, pp. 2250–2267, 2014 (pp. 17, 20, 23).

[64] S. Haghtalab, P. Xanthopoulos and K. Madani, 'A robust unsupervised consensus control chart pattern recognition framework', *Expert Systems with Applications*, vol. 42, no. 19, pp. 6767–6776, 2015 (p. 20).

[65] D. J. Hand, 'Measuring classifier performance: A coherent alternative to

the area under the roc curve', *Machine learning*, vol. 77, no. 1, pp. 103–123, 2009 (p. 68).

[66] R. Hickey and M. Black, 'Refined time stamps for concept drift detection during mining for classification rules', *Temporal, Spatial, and Spatio-Temporal Data*, 2001 (pp. 7, 15).

[67] W. Hoeffding, 'Probability inequalities for sums of bounded random variables', *Journal of the American statistical association*, vol. 58, no. 301, pp. 13–30, 1963 (p. 37).

[68] A. Hoover, A. Singh, S. Fishel-Brown and E. Muth, 'Real-time detection of workload changes using heart rate variability', *Biomedical Signal Processing*, 2012 (p. 15).

[69] W.-B. Horng, C.-Y. Chen, Y. Chang and C.-H. Fan, 'Driver fatigue detection based on eye tracking and dynamk, template matching', in *Networking, Sensing and Control, 2004 IEEE International Conference on*, IEEE, vol. 1, 2004, pp. 7–12 (pp. 117, 119).

[70] H. Hotelling, 'The generalization of student's ratio', *The Annals of Mathematical Statistics*, vol. 2, no. 3, pp. 360–378, 1931 (pp. 18, 19, 30, 42).

[71] ——, *Multivariate quality control illustrated by the air testing of sample bomb sights, techniques of statistical analysis, ch. ii*, 1947 (p. 78).

[72] D. T. J. Huang, Y. S. Koh, G. Dobbie and R. Pears, 'Detecting volatility shift in data streams', in *Data Mining (ICDM), 2014 IEEE International Conference on*, IEEE, 2014, pp. 863–868 (pp. 41, 95).

[73] L. Huang, X. Nguyen, M. Garofalakis, M. I. Jordan, A. Joseph and N. Taft, 'In-network pca and anomaly detection', in *Advances in Neural Information Processing Systems*, 2007, pp. 617–624 (pp. 55, 56).

[74] G. Hulten, L. Spencer and P. Domingos, 'Mining time-changing data streams', in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2001, pp. 97–106 (pp. 1, 19).

[75] T. Ito, S. Mita, K. Kozuka, T. Nakano and S. Yamamoto, 'Driver blink measurement by the motion picture processing and its application to drowsiness detection', in *Intelligent Transportation Systems, 2002. Proceedings. The IEEE 5th International Conference on*, IEEE, 2002, pp. 168–173 (p. 117).

[76] J. E. Jackson, 'Quality control methods for several related variables', *Technometrics*, vol. 1, no. 4, pp. 359–377, 1959 (p. 18).

[77] J. E. Jackson and R. A. Bradley, 'Sequential $\chi^2$-and $T^2$-tests', *The Annals of Mathematical Statistics*, pp. 1063–1077, 1961 (p. 19).

[78] A. Jaffe, E. Fetaya, B. Nadler, T. Jiang and Y. Kluger, 'Unsupervised ensemble learning with dependent classifiers', in *Artificial Intelligence and Statistics*, 2016, pp. 351–360 (p. 172).

[79] A. Jain and B. Chandrasekaran, *ªdimensionality and sample size considerations in pattern recognition practice, º handbook of statistics. pr krishnaiah and ln kanal, eds., vol. 2*, 1982 (p. 73).

[80] Y. Kawahara and M. Sugiyama, 'Change-point detection in time-series data by direct density-ratio estimation', *Proceedings of the 2009 SIAM International*, 2009 (p. 48).

[81] *KDD Cup 1999 Data*. [Online]. Available: `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html` (p. 49).

[82] M. G. Kelly, D. J. Hand and N. M. Adams, 'The impact of changing populations on classifier performance', in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 1999, pp. 367–371 (p. 12).

[83] D. Kifer, S. Ben-David and J. Gehrke, 'Detecting change in data streams', in *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, VLDB Endowment, 2004, pp. 180–191 (pp. 20, 30, 38, 80).

[84] R. Klinkenberg and T. Joachims, 'Detecting concept drift with support vector machines', *Proceedings of ICML-00, 17th International Confer-*

*ence on Machine Learning*, pp. 487–494, 2000 (pp. 7, 14, 33, 39, 46, 48).

[85] R. Klinkenberg and I. Renz, 'Adaptive information filtering: Learning in the presence of concept drifts', *Learning for Text Categorization*, pp. 33–40, 1998 (pp. 1, 33).

[86] J. Z. Kolter and M. A. Maloof, 'Dynamic weighted majority: A new ensemble method for tracking concept drift', in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, IEEE, 2003, pp. 123–130 (pp. 48, 49).

[87] P. Kosina, J. Gama and R. Sebastiao, 'Drift severity metric.', in *ECAI*, 2010, pp. 1119–1120 (pp. 7, 9, 49).

[88] I. Koychev and R. Lothian, 'Tracking drifting concepts by time window optimisation', *Research and Development in Intelligent Systems XXII*, pp. 46–59, 2006 (p. 33).

[89] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski and M. Woźniak, 'Ensemble learning for data stream analysis: A survey', *Information Fusion*, vol. 37, pp. 132–156, 2017 (pp. 20, 94).

[90] G. Krempl, Z. F. Siddiqui and M. Spiliopoulou, 'Online clustering of high-dimensional trajectories under concept drift', in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, 2011, pp. 261–276 (p. 41).

[91] A. Królak and P. Strumiłło, 'Fatigue monitoring by means of eye blink analysis in image sequences', *ICSES*, vol. 1, pp. 219–222, 2006 (pp. 119, 120, 126).

[92] A. Krolak and P. Strumillo, 'Vision-based eye blink monitoring system for human-computer interfacing', in *Human System Interactions, 2008 Conference on*, IEEE, 2008, pp. 994–998 (p. 117).

[93] A. Królak and P. Strumiłło, 'Eye-blink detection system for human–computer interaction', *Universal Access in the Information Society*, vol. 11, no. 4, pp. 409–419, 2012 (pp. 117–119, 159).

[94] C. Kruegel and G. Vigna, 'Anomaly detection of web-based attacks', in *Proceedings of the 10th ACM conference on Computer and communications security*, ACM, 2003, pp. 251–261 (p. 48).

[95] S. Kullback and R. A. Leibler, 'On information and sufficiency', *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951 (pp. 1, 29, 45).

[96] L. Kuncheva, 'Classifier ensembles for detecting concept change in streaming data: Overview and perspectives', *Proceedings of the Second Workshop SUEMA, ECAI 2008*, no. July, pp. 5–9, 2008 (pp. 48, 76, 92, 95).

[97] L. Kuncheva, 'Using control charts for detecting concept change in streaming data', *Bangor University*, 2009. [Online]. Available: https://www.bangor.ac.uk/cs/Documents/CS-TR1-2009%20-%20Using%20Control%20Charts%20for%20Detecting%20Concept.pdf (p. 48).

[98] L. I. Kuncheva, 'That elusive diversity in classifier ensembles', in *IbPRIA*, Springer, vol. 2652, 2003, pp. 1126–1138 (p. 97).

[99] ——, 'Classifier ensembles for changing environments', *Multiple classifier systems*, vol. 3077, pp. 1–15, 2004 (p. 48).

[100] ——, 'A stability index for feature selection.', in *Artificial intelligence and applications*, 2007, pp. 421–427 (p. 73).

[101] ——, 'Classifier Ensembles for Detecting Concept Change in Streaming Data : Overview and Perspectives', *Computer*, no. July, pp. 5–9, 2008 (p. 46).

[102] ——, 'Change detection in streaming Multivariate data using likelihood detectors', *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 5, pp. 1175–1180, 2013, ISSN: 1041-4347. DOI: 10.1109/tkde.2011.226 (pp. 20, 41–43, 48, 60, 76, 79, 92, 93, 95, 137).

[103] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms: Second Edition*. Wiley Blackwell, Sep. 2014, pp. 1–357 (p. 97).

[104] L. I. Kuncheva and W. J. Faithfull, 'PCA feature extraction for change

detection in multidimensional unlabeled data', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 69–80, Jan. 2014 (pp. 11, 57, 79).

[105] T. L. Lai, 'Sequential Changepoint Detection in Quality Control and Dynamical Systems', *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 57, pp. 613–658, 1995 (p. 14).

[106] M. Lalonde, D. Byrns, L. Gagnon, N. Teasdale and D. Laurendeau, 'Real-time eye blink detection with gpu-based sift tracking', in *Computer and Robot Vision, 2007. CRV'07. Fourth Canadian Conference on*, IEEE, 2007, pp. 481–487 (pp. 117, 119, 120).

[107] M. Lavielle and G. Teyssiere, 'Detection of multiple change-points in multivariate time series', *Lithuanian Mathematical Journal*, 2006 (p. 15).

[108] M. Lazarescu and S. Venkatesh, 'Using multiple windows to track concept drift', *Intelligent data analysis*, 2004 (pp. 9, 13).

[109] D. D. Lee and H. S. Seung, 'Algorithms for non-negative matrix factorization', in *Advances in neural information processing systems*, 2001, pp. 556–562 (p. 55).

[110] W. O. Lee, E. C. Lee and K. R. Park, 'Blink detection robust to various facial poses', *Journal of neuroscience methods*, vol. 193, no. 2, pp. 356–372, 2010 (pp. 119, 121, 126).

[111] D. Li, D. Winfield and D. J. Parkhurst, 'Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches', in *Computer Vision and Pattern Recognition-Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, IEEE, 2005, pp. 79–79 (pp. 119, 120).

[112] M. Lichman, *{UCI} Machine Learning Repository*, 2013. [Online]. Available: `http://archive.ics.uci.edu/ml` (pp. 47, 65, 92, 104, 116).

[113] R. Lienhart and J. Maydt, 'An extended set of haar-like features for rapid object detection', in *Proceedings. International Conference on Image Processing*, vol. 1, 2002, I-900-I-903 vol.1. DOI: `10.1109/ICIP.2002.`

1038171 (pp. 126, 127).

[114] C. A. Lowry and D. C. Montgomery, ‘A review of multivariate control charts’, *IIE transactions*, vol. 27, no. 6, pp. 800–810, 1995 (p. 78).

[115] C. A. Lowry, W. H. Woodall, C. W. Champ and S. E. Rigdon, ‘A multivariate exponentially weighted moving average control chart’, *Technometrics*, vol. 34, no. 1, pp. 46–53, 1992 (p. 76).

[116] B. D. Lucas, T. Kanade *et al.*, ‘An iterative image registration technique with an application to stereo vision’, 1981 (p. 119).

[117] A. Lung-Yut-Fong, C. Lévy-Leduc and O. Cappé, ‘Robust changepoint detection based on multivariate rank statistics’, in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, IEEE, 2011, pp. 3608–3611 (pp. 30, 60, 61).

[118] J. F. MacGregor and T. Kourti, ‘Statistical process control of multivariate processes’, *Control Engineering Practice*, vol. 3, no. 3, pp. 403–414, 1995 (pp. 18, 43).

[119] B. I. F. Maciel, S. G. T. C. Santos and R. S. M. Barros, ‘A Lightweight Concept Drift Detection Ensemble’, in *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, IEEE, Nov. 2015, pp. 1061–1068 (pp. 20, 94).

[120] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow and B. Frey, ‘Adversarial autoencoders’, *arXiv preprint arXiv:1511.05644*, 2015 (p. 173).

[121] 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*, pp. 50–60, 1947 (p. 29).

[122] M. Markou and S. Singh, *Novelty detection: A review - Part 1: Statistical approaches*, Dec. 2003 (p. 16).

[123] M. M. Masud, Q. Chen, L. Khan, C. Aggarwal, J. Gao, J. Han and B. Thuraisingham, ‘Addressing concept-evolution in concept-drifting data streams’, in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, IEEE, 2010, pp. 929–934 (p. 13).

[124] M. M. Masud, C. Woolam, J. Gao, L. Khan, J. Han, K. W. Hamlen and N. C. Oza, 'Facing the reality of data stream classification: Coping with scarcity of labeled data', *Knowledge and information systems*, vol. 33, no. 1, pp. 213–244, 2012 (p. 10).

[125] *Matlab 2015a*, The MathWorks Inc., Natick, MA, USA, 2015 (pp. 34, 35, 127).

[126] P. Micó, M. Mora, D. Cuesta-Frau and M. Aboy, 'Automatic segmentation of long-term ECG signals corrupted with broadband noise based on sample entropy', *Computer Methods and Programs*, 2010 (p. 15).

[127] L. Minku, A. White and X. Yao, 'The impact of diversity on online ensemble learning in the presence of concept drift', *IEEE Transactions on Knowledge*, 2010 (pp. 8, 9, 49).

[128] S. Mohamad, A. Bouchachia and M. Sayed-Mouchaweh, 'A bi-criteria active learning algorithm for dynamic data streams', *IEEE transactions on neural networks and learning systems*, vol. 29, no. 1, pp. 74–86, 2018 (p. 13).

[129] S. Mohamad, M. Sayed-Mouchaweh and A. Bouchachia, 'Active learning for classifying data streams with unknown number of classes', *Neural Networks*, vol. 98, pp. 1–15, 2018 (p. 13).

[130] D. Montgomery, *Statistical quality control*. 2009 (pp. 14, 16).

[131] T. Morris, P. Blenkhorn and F. Zaidi, 'Blink detection for real-time eye tracking', *Journal of Network and Computer Applications*, vol. 25, no. 2, pp. 129–143, 2002 (pp. 117, 119, 120).

[132] H. Mouss, D. Mouss, N. Mouss and L. Sefouhi, 'Test of page-hinckley, an approach for fault detection in an agro-alimentary production system', *Proceedings of the 5th Asian Control Conference*, pp. 815–818, 2004 (pp. 15, 36).

[133] A. Nanduri and L. Sherry, 'Anomaly detection in aircraft data using recurrent neural networks (rnn)', in *Integrated Communications Navigation and Surveillance (ICNS), 2016*, IEEE, 2016, pp. 5C2–1 (pp. 20,

173).

[134] A. M. Narasimhamurthy and L. I. Kuncheva, 'A framework for generating data to simulate changing environments.', in *Artificial Intelligence and Applications*, 2007, pp. 415–420 (pp. 7–9, 11, 49, 50, 168).

[135] 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, pp. 1822–1829, 2014 (pp. 1, 42, 92).

[136] S. Oh, M. S. Lee and B.-T. Zhang, 'Ensemble learning with active example selection for imbalanced biomedical data classification.', *IEEE/ACM transactions on computational biology and bioinformatics / IEEE, ACM*, vol. 8, no. 2, pp. 316–25, 2010 (p. 15).

[137] E. S. Page, 'Continuous Inspection Schemes', *Biometrika*, vol. 41, no. 1/2, p. 100, 1954 (pp. 14, 18, 19, 35, 36, 95).

[138] G. Pan, L. Sun, Z. Wu and S. Lao, 'Eyeblink-based anti-spoofing in face recognition from a generic webcamera', in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, IEEE, 2007, pp. 1–8 (p. 117).

[139] M. A. F. Pimentel, D. A. Clifton, L. Clifton and L. Tarassenko, *A review of novelty detection*, Jun. 2014 (pp. 16, 20–22, 24).

[140] J. Portelo, M. Bugalho, I. Trancoso, J. Neto, A. Abad and A. Serralheiro, 'Non-speech audio event detection', in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, IEEE, 2009, pp. 1973–1976 (pp. 55, 56).

[141] J. Reeves, J. Chen, X. L. Wang, R. Lund, Q. Q. Lu, J. Reeves, J. Chen, X. L. Wang, R. Lund and Q. Q. Lu, 'A Review and Comparison of Changepoint Detection Techniques for Climate Data', *Journal of Applied Meteorology and Climatology*, vol. 46, no. 6, pp. 900–915, Jun. 2007 (pp. 15, 16).

[142] H. Ringberg, A. Soule, J. Rexford and C. Diot, 'Sensitivity of pca for traffic anomaly detection', *ACM SIGMETRICS Performance Evaluation Review*,

vol. 35, no. 1, pp. 109–120, 2007 (pp. 55, 56).

[143] S. Roberts, ‘Control chart tests based on geometric moving averages’, *Technometrics*, 1959. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/00401706.1959.10489860 (pp. 14, 18, 38, 95).

[144] G. J. Ross, N. M. Adams, D. K. Tasoulis and D. J. Hand, ‘Exponentially weighted moving average charts for detecting concept drift’, *Pattern Recognition Letters*, vol. 33, no. 2, pp. 191–198, 2012 (pp. 38, 48, 95, 125).

[145] G. C. Runger, T. R. Wlllemain and S. Prabhu, ‘Average run lengths for cusum control charts applied to residuals’, *Communications in Statistics-Theory and Methods*, vol. 24, no. 1, pp. 273–282, 1995 (p. 34).

[146] S. Sakthithasan, R. Pears and Y. S. Koh, ‘One pass concept change detection for data streams’, *Advances in Knowledge Discovery and Data Mining*, pp. 461–472, 2013 (pp. 40, 95).

[147] M. Salganicoff, ‘Tolerating concept and sampling shift in lazy learning using prediction error context switching’, *Artificial Intelligence Review*, 1997 (p. 13).

[148] J. C. Schlimmer and R. H. Granger, ‘Beyond incremental processing: Tracking concept drift.’, in *AAAI*, 1986, pp. 502–507 (p. 48).

[149] ——, ‘Incremental learning from noisy data’, *Machine learning*, vol. 1, no. 3, pp. 317–354, 1986 (p. 19).

[150] W. Shewhart, *Economic control of quality of manufactured product*. 1931 (pp. 14, 18, 21, 34).

[151] A. Shiryaev, ‘The problem of the most rapid detection of a disturbance in a stationary process’, *Soviet Math. Dokl*, 1961 (p. 19).

[152] A. H. Shoeb and J. V. Guttag, ‘Application of machine learning to epileptic seizure detection’, in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 975–982 (p. 117).

[153] A. Singh, ‘Review article digital change detection techniques using remotely-sensed data’, *International journal of remote sensing*, vol. 10,

no. 6, pp. 989–1003, 1989 (pp. 55, 120).

[154] C. Song, C. E. Woodcock, K. C. Seto, M. P. Lenney and S. A. Macomber, 'Classification and change detection using landsat tm data: When and how to correct atmospheric effects?', *Remote sensing of Environment*, vol. 75, no. 2, pp. 230–244, 2001 (p. 120).

[155] X. Song, M. Wu, C. Jermaine and S. Ranka, 'Statistical change detection for multi-dimensional data', *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '07*, vol. V, p. 667, 2007 (pp. 41, 48, 79, 80).

[156] E. J. Spinosa, A. P. de Leon F de Carvalho and J. Gama, 'Cluster-based novel concept detection in data streams applied to intrusion detection in computer networks', in *Proceedings of the 2008 ACM Symposium on Applied computing*, ACM, 2008, pp. 976–980 (p. 15).

[157] M. Staudacher, S. Telser, A. Amann, H. Hinterhuber and M. Ritsch-Marte, 'A new method for change-point detection developed for on-line analysis of the heart beat variability during sleep', *Physica A: Statistical Mechanics and its Applications*, vol. 349, no. 3, pp. 582–596, 2005 (p. 15).

[158] J. A. Stern, D. Boyer and D. Schroeder, 'Blink rate: A possible measure of fatigue', *Human factors*, vol. 36, no. 2, pp. 285–297, 1994 (p. 124).

[159] J. A. Stern and J. J. Skelly, 'The eye blink and workload considerations', in *Proceedings of the Human Factors Society Annual Meeting*, Sage Publications Sage CA: Los Angeles, CA, vol. 28, 1984, pp. 942–944 (pp. 118, 124).

[160] W. N. Street and Y. Kim, 'A streaming ensemble algorithm (sea) for large-scale classification', in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2001, pp. 377–382 (pp. 7, 19, 49, 94).

[161] M. Suzuki, N. Yamamoto, O. Yamamoto, T. Nakano and S. Yamamoto, 'Measurement of driver's consciousness by image processing-a method

for presuming driver's drowsiness by eye-blinks coping with individual differences', in *Systems, Man and Cybernetics, 2006. SMC'06. IEEE International Conference on*, IEEE, vol. 4, 2006, pp. 2891–2896 (p. 120).

[162] L. Tarassenko, A. Hann and D. Young, 'Integrated monitoring and analysis for early warning of patient deterioration.', *British journal of anaesthesia*, vol. 97, no. 1, pp. 64–8, Jul. 2006 (pp. 1, 41).

[163] A. Tartakovsky and G. Moustakides, 'State-of-the-art in Bayesian change-point detection', *Sequential Analysis*, 2010 (p. 48).

[164] A. G. Tartakovsky, B. L. Rozovskii, R. B. Blažek and H. Kim, 'Detection of intrusions in information systems by sequential change-point methods', *Statistical Methodology*, vol. 3, no. 3, pp. 252–293, Jul. 2006 (pp. 15, 48, 92, 94, 114).

[165] M. Tavallaee, E. Bagheri, W. Lu and A. A. Ghorbani, 'A Detailed Analysis of the KDD CUP 99 Data Set', (pp. 49, 105).

[166] A. Theissler, 'Detecting known and unknown faults in automotive systems using ensemble-based anomaly detection', *Knowledge-Based Systems*, vol. 123, pp. 163–173, 2017 (p. 20).

[167] P. Thoumie, J. Charlier, M. Alecki, D. d'Erceville, A. Heurtin, J. Mathe, G. Nadeau and L. Wiart, 'Clinical and functional evaluation of a gaze controlled system for the severely handicapped', *Spinal Cord*, vol. 36, no. 2, p. 104, 1998 (p. 119).

[168] A. Tsymbal, 'The problem of concept drift: definitions and related work', *Computer Science Department, Trinity College Dublin*, vol. 4, no. C, pp. 2004–15, 2004 (pp. 7, 13).

[169] P. Viola and M. Jones, 'Rapid object detection using a boosted cascade of simple features', in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, IEEE, vol. 1, 2001, pp. I–I (pp. 119, 126).

[170] P. Viola and M. J. Jones, 'Robust real-time face detection', *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004 (pp. 126–

128).

[171] A. Wald, ‘Sequential Tests of Statistical Hypotheses’, *The Annals of Mathematical Statistics*, vol. 16, no. 2, pp. 117–186, Jun. 1945 (pp. 17–19, 21).

[172] A. Wald, *Sequential Analysis*. Wiley, 1947 (pp. 19, 34).

[173] X. Wang, U. Kruger, G. W. Irwin, G. McCullough and N. McDowell, ‘Nonlinear pca with the local approach for diesel engine fault detection and diagnosis’, *IEEE Transactions on Control Systems Technology*, vol. 16, no. 1, pp. 122–129, 2008 (p. 55).

[174] G. Widmer and M. Kubat, ‘Learning flexible concepts from streams of examples: Flora2’, in *Proceedings of the 10th European conference on Artificial intelligence*, John Wiley & Sons, Inc., 1992, pp. 463–467 (p. 33).

[175] ——, ‘Effective learning in dynamic environments by explicit context tracking’, in *Machine learning: ECML-93*, Springer, 1993, pp. 227–243 (p. 13).

[176] ——, ‘Learning in the presence of concept drift and hidden contexts’, *Machine Learning*, vol. 23, no. 1, pp. 69–101, 1996 (pp. 1, 7, 14, 19, 33, 39, 46, 48, 86).

[177] M. Woźniak, P. Ksieniewicz, B. Cyganek and K. Walkowiak, ‘Ensembles of heterogeneous concept drift detectors-experimental study’, in *IFIP International Conference on Computer Information Systems and Industrial Management*, Springer, 2016, pp. 538–549 (pp. 20, 94).

[178] W.-A. Yang and W. Zhou, ‘Autoregressive coefficient-invariant control chart pattern recognition in autocorrelated manufacturing processes using neural network ensemble’, *Journal of Intelligent Manufacturing*, vol. 26, no. 6, pp. 1161–1180, 2015 (pp. 20, 173).

[179] Z. R. Zaidi, S. Hakami, T. Moors and B. LANDFELDT, ‘Detection and identification of anomalies in wireless mesh networks using principal component analysis (pca)’, *Journal of Interconnection Networks*, vol. 10,

no. 04, pp. 517–534, 2009 (p. 55).

[180] K. Zamba and D. M. Hawkins, ‘A multivariate change-point model for statistical process control’, *Technometrics*, vol. 48, no. 4, pp. 539–549, 2006 (p. 79).

[181] Y. Zhang, N. Meratnia and P. Havinga, ‘A taxonomy framework for unsupervised outlier detection techniques for multi-type data sets’, Centre for Telematics and Information Technology, University of Twente, Tech. Rep., 2007 (p. 20).

[182] Z. Zhang, D. Yi, Z. Lei and S. Z. Li, ‘Face liveness detection by learning multispectral reflectance distributions’, in *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, IEEE, 2011, pp. 436–441 (p. 117).

[183] C. Zhou, C. Zou, Y. Zhang, Z. Wang *et al.*, ‘Nonparametric control chart based on change-point model’, *Statistical Papers*, vol. 50, no. 1, pp. 13–28, 2009 (p. 48).

[184] Z.-H. Zhou, N. V. Chawla, Y. Jin and G. J. Williams, ‘Big Data Opportunities and Challenges: Discussions from Data Analytics Perspectives [Discussion Forum]’, *IEEE Computational Intelligence Magazine*, vol. 9, no. 4, pp. 62–74, Nov. 2014 (p. 6).

[185] A. Zimek, R. J. Campello and J. Sander, ‘Ensembles for unsupervised outlier detection: Challenges and research questions a position paper’, *ACM Sigkdd Explorations Newsletter*, vol. 15, no. 1, pp. 11–22, 2014 (p. 172).

[186] I. Žliobaitė, ‘Learning under concept drift: An overview’, *arXiv preprint arXiv:1010.4784*, 2010 (pp. 7–9, 12, 20, 23).

[187] F. Zorriassatine, A. Al-Habaibeh, R. M. Parkin, M. R. Jackson and J. Coy, ‘Novelty detection for practical pattern recognition in condition monitoring of multivariate processes: A case study’, *International Journal of Advanced Manufacturing Technology*, vol. 25, no. 9-10, pp. 954–963, May 2005 (pp. 1, 41, 92).