Learning from Data Streams:
Synopsis and Change Detection

Ana Raquel Ferreira de Almeida Sebastião

PhD Programme in Applied Mathematics
Department of Mathematics
2013

Learning from Data Streams:
Synopsis and Change Detection

Ana Raquel Ferreira de Almeida Sebastião

Thesis submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Applied Mathematics

Department of Mathematics
Faculty of Science
University of Porto
2013

Author

Ana Raquel Ferreira de Almeida Sebastião, MSc

PhD Student
LIAAD - INESC TEC

Advisers

João Manuel Portela da Gama, PhD

Associate Professor
Faculty of Economics of the University of Porto

Researcher
LIAAD - INESC TEC

Teresa Maria de Gouveia Torres Feio Mendonça, PhD

Assistant Professor
Faculty of Science of the University of Porto

Researcher
Center for Research & Development in Mathematics and Applications, Aveiro

To my husband, my sense of direction.
To my daughter, my source of strength.

Financial Support

This work was financially supported by Fundação para a Ciência e a Tecnologia (FCT) with the PhD Grant SFRH/BD/41569/2007 (through the QREN-POPH program, partly funded by the European Union through the European Social Fund and by the Portuguese government).

This work was part-funded by FCT national funds in the context of the projects KDUDS (PTDC/EIA-EIA/098355/2008) and GALENO (PTDC/SAU-BEB/103667/2008).

This work was also part-funded by the ERDF - European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by the Portuguese Funds through the FCT within the project FCOMP - 01-0124-FEDER-022701

Acknowledgments

I owe a debt of sincere gratitude to many people and several institutions that have helped me to accomplish this PhD work.

I would like to thank Prof. João Gama, Associate Professor at the Faculty of Economics of the University of Porto, and Prof. Teresa Mendonça, Assistant Professor at the Faculty of Science of the University of Porto, for accepting to jointly supervise this thesis. Prof. João Gama guided me through machine learning and motivated my interest in change detection. Prof. Teresa Mendonça provided the opportunity for interesting application areas. Regarding the aims of my PhD, this was the perfect blend. Their expertise and guidance has been priceless. I am grateful for all the discussions, especially the tough ones, and all the knowledge shared. I thank both of you for encouraging my work and supporting my growth as a researcher.

I acknowledge the Faculty of Science of the University of Porto (FCUP), which has been my "home" ever since I started my undergraduate studies.

I am grateful for the support of LIAAD, my host institution during this PhD project. I have grown in so many aspects within this group, with its members and their expertise.

I owe profound thanks to Prof. Pavel Brazdil, founder, ex-director and fellow of LIAAD. He is an extraordinary example of a dedicated researcher; his scientific knowledge is boundless, while at the same time he is down to earth. I also thank Prof. Alípio Jorge, current director of LIAAD - INESC TEC, for his support and encouragement. I wish him all the success in this new challenge.

Kindly thanks to Rita Ribeiro and Márcia Oliveira for being my female company at LIAAD and for the many enjoyable (and long) chats.

Special thanks to Pedro Pereira Rodrigues and Margarida M. Silva for being so supportive along this path and for the collaborative research in different areas.

I thank some colleagues from the Mathematics Department, especially Sónia Gouveia and Conceição Rocha, for their assistance on mathematical questions and valuable insights.

I am also grateful to those friends that gave me the right motivation at the right time, especially during the last month of writing. I consider that names are not needed, since you know you have been there!

I am indebted to the GALENO team because the research collaboration with some of its members made this work possible.

Friends surely make my life more meaningful and fulfilled. For sincere and genuine friendship, I thank you all.

Family plays the most important role in my life and I am deeply grateful to you all for your love! I am certainly not a "man of words" and I could never express in words how grateful I am to my family for their unconditional support and constant encouragement. To my parents, Isabel and Ramiro, I thank you for everything... all the lessons I have learnt from you. But mainly, I thank you for the most important value you have given me and that has sustained me throughout the years: education.

More than my husband, Bruno you are my most truthful and kindest friend. You have stood beside me at all times and pushed me ahead with an unwavering faith, even when things seemed hopeless to me. I cannot thank you enough. I would also like to acknowledge the most important person in my life: my beloved one and a half year old daughter. Francisca, you make every second worthwhile (including those spent doing this PhD).

Finally, I express my humble apologies to those whom I might I have forgotten to thank.

Abstract

The emergence of real temporal applications under non-stationary scenarios has drastically altered the ability to generate and gather information. Nowadays, potentially unbounded and massive amounts of information are generated at high-speed rate, known as data streams. Therefore, it is unreasonable to assume that machine learning systems have sufficient memory capacity to store the complete history of the stream. Indeed, stream learning algorithms must process data promptly, discarding it immediately. Along with this, as data flows continuously for large periods of time, the process generating data is not strictly stationary and evolves over time.

This thesis embraces concerns raised when learning from data streams. Namely, concerns raised by the intrinsic characteristics of data streams and by the learning process itself. The former is addressed through the construction of synopses structures of data and change detection methods. The latter is related to the appropriate evaluation of stream learning algorithms.

Given the huge volume of data gathered, it is essential to create synopses structures of data, keeping only a small and finite representation of the received information. Such compact summaries allow the discarded data to be remembered. In this thesis, and within this context, online equi-width histograms, under mean square error constraints, are proposed to construct compact representations of data.

When dealing with data streams in evolving environments, in addition to the remembering approach, it is also necessary to forget outdated data: old observations do not describe the current state of nature and are, therefore, useless. This issue is accomplished by two approaches: a forgetting data synopsis and a windows scheme model for change detection. As the proposed fading histograms weight data examples according to their age, as well as allowing discarded data to be remembered, outdated data is gradually forgotten. Therefore, the data representation provided by these fading histograms is more up-to-date. The Cumulative Windows Model for change detection proposed in this thesis, is based on online monitoring of the distance between data distributions (provided by the histograms mentioned earlier), which is evaluated using the Kullback-Leibler divergence.

Within this approach, after the detection of a change, all past observations are abruptly forgotten.

Learning from time-changing data streams is not a straightforward task. Traditional learning algorithms, which produce models by learning in a batch mode, are not appropriate for use in such a context. Within a continuous flow of data, the learning process must be carried out sequentially. Although there is a vast number of streaming learning algorithms, the metrics and the design of experiments for assessing the quality of learning models is still an open issue. Therefore, as a main contribution of this thesis, new criteria for effectively evaluating algorithms when learning from time-changing data streams are proposed. These evaluation metrics are based on computing prequential error estimates using forgetting mechanisms: either a sliding window or fading factors. Moreover, the proofs of convergence of different error estimates to the Bayes error are also presented.

Given the recent growth of biomedical applications, the automatic detection of changes in physiological signals is a flourishing topic of research. Therefore, this thesis presents a real-time algorithm for change detection in depth of anesthesia (DoA) signals of patients undergoing surgery. The use of an effective change detection method has a high impact in clinical practice, since it alerts the clinician in advance to changes in the anesthetic state of the patient, allowing prompter actions. The remarkably encouraging results obtained when detecting changes in DoA signals, sustain the argument that the proposed change detection approach should be embedded in a real-time decision support system for routine use in clinical practice.

Resumo

O aparecimento de aplicações reais em cenários temporais não estacionárias alterou drasticamente a capacidade de gerar e recolher informação. Hoje em dia, uma quantidade enorme, potencialmente infinita, de informação é gerada a elevada velocidade, conhecida como sendo um fluxo contínuo de dados. Portanto, é irrazoável supor que os sistemas de processamento de dados têm capacidade de memória suficiente para armazenar, por completo, estes fluxos contínuos. De facto, os algoritmos de aprendizagem em fluxos contínuos de dados têm de processá-los rapidamente e descartá-los de seguida. Juntamente com esta limitação, como os fluxos contínuos de dados são gerados durante largos períodos de tempo, o seu processo de geração não é estritamente estacionário e evolui ao longo do tempo.

Esta tese aborda questões levantadas aquando da aprendizagem a partir de fluxos contínuos de dados, nomeadamente, questões levantadas pelas características intrínsecas dos fluxos contínuos de dados e pelo próprio processo de aprendizagem. As primeiras são abordadas através da construção de estruturas sumárias de dados e através de métodos de deteção de mudanças. As últimas estão relacionadas com a avaliação adequada de algoritmos de aprendizagem a partir de fluxos contínuos de dados.

Relativamente ao grande volume de dados recolhidos, é necessária a criação de estruturas sumárias de dados, mantendo apenas uma representação pequena e finita da informação recebida. Estes resumos compactos permitem relembrar os dados descartados. Neste contexto, são propostos histogramas com classes de igual amplitude, em tempo real e com limitações no erro médio quadrático, para a construção de estruturas sumárias de dados.

Ao trabalhar com fluxos contínuos de dados em ambientes evolutivos, além da abordagem de os relembrar, torna-se necessário esquecer os dados desatualizados: observações antigas não descrevem o estado atual da natureza e, portanto, são inúteis. Nesta tese, esta questão tem duas abordagens: a construção de uma estrutura sumária de dados com propriedades de esquecimento e um modelo, baseado em janelas deslizantes, para a deteção de mudanças nos dados. Os histogramas de esquecimento propostos, pelo facto de atribuírem aos dados uma ponderação de acordo com a sua idade, além de permitirem relembrar os dados

descartados, possibilitam, também, esquecer dados desatualizados de um modo gradual. Portanto, a representação de dados obtida por estes histogramas de esquecimento é mais atualizada. O Modelo de Janelas Acumulativas para a deteção de mudanças proposto nesta tese, é baseado na monotorização em tempo real da distância entre distribuições de dados (fornecidas pelos histogramas mencionados anteriormente), avaliada através da divergência *Kullback-Leibler*. Relativamente ao modelo de deteção de mudança, depois de detetada uma mudança, todas as observações anteriores são abruptamente esquecidas.

A aprendizagem através de fluxos contínuos de dados não estacionários não é uma tarefa simples. Os algoritmos de aprendizagem tradicionais, que produzem modelos através de uma aprendizagem efetuada num conjunto de treino, não se adequam ao problema. Tratando-se de um fluxo contínuo de dados, o processo de aprendizagem tem de ser efetuado incrementalmente. De facto, atualmente, existe um grande número de algoritmos de aprendizagem incrementais, contudo as métricas e o delineamento de experiências para avaliar a qualidade dos mesmos é um tema em aberto. Portanto, como contribuição deste trabalho, são propostos novos critérios para avaliar algoritmos de aprendizagem incremental. As métricas de avaliação propostas são baseadas no cálculo de estimativas de erro, usando mecanismos de esquecimento: uma janela deslizante ou fatores de esquecimento. São também apresentadas provas da convergência destas estimativas de erro para o erro de Bayes.

Considerando o crescimento recente de aplicações em biomedicina, a deteção automática de mudanças em sinais fisiológicos é um tema de investigação em crescimento. Portanto, neste trabalho é apresentado um algoritmo para deteção, em tempo real, de mudanças em sinais fisiológicos, nomeadamente no BIS (índice bi-espectral), de pacientes submetidos a cirurgia. A utilização de um método de deteção de mudança eficaz tem um grande impacto na prática clínica, no sentido em que alerta, antecipadamente, o médico para alterações no estado de anestesia do paciente, permitindo uma ação médica mais rápida. Os resultados obtidos na deteção de mudanças nos sinais BIS sustentam que o método de deteção de mudanças proposto deve ser incorporado num sistema de apoio à decisão do controlo de anestesia a utilizar na prática clínica em bloco operatório.

Contents

List of Tables

List of Figures

- 1 -

Introduction

"All truths are easy to understand once they are discovered;
the point is to discover them."
Galileo Galilei (1564 - 1642)

This introductory chapter provides an overview of the thesis. Starting with the context of the research, research questions are proposed and contributions achieved are provided. Before concluding this chapter with the outline of the thesis, a bibliographical note of the publications resulting from the research developed during the course of the doctoral project is presented.

1.1 Context of Research

Learning is an inborn ability of human beings: to survive, man needs to learn as much as he needs to breath. Given this truth, humans have invented machine learning, which mimics human ability through computational models. As with human learning, machine learning does not happen all at once, it is an ongoing process of upgrading based on experience.

Tom M. Mitchell provided the following definition of machine learning: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E",(Mitchell, 1997).

Along with the learning skills, ever since the beginning, critical capacity and curiosity has guided humans through the search for a better understanding of the surrounding world. We are a long way of fully comprehending everything around us, but, on the endless path of progress, man has always earnestly pursued new and greater amounts of information,

knowledge and discoveries.

Therefore, data mining has appeared. Data mining is concerned with the extraction of information, through data analysis methods, and the transformation of that information in order to discover knowledge, through machine learning methods. To quote Albert Einstein, "Information is not knowledge". Indeed, data as it is collected is useless: certain techniques need to be applied so useful information can be extracted from the data, transforming the information contained in raw data into knowledge, allowing patterns underlying the data to be discovered.

In the data mining context, knowledge is an understandable model or theory which explains the information enclosed in the form of a data set. The ultimate goal of a data mining process is to create models, which, when further applied to new data, reveal new patterns underlying the data.

Data mining comes hand to hand with the process of Knowledge Discovery in Databases (KDD). Proceeded by data selection, pre-processing and transformation, data mining is the computational extraction of patterns representing knowledge implicitly stored in the database. Figure 1.1 presents a flow chart, exemplifying the KDD process, which ends with the interpretation and evaluation of the results produced by the data mining stage.

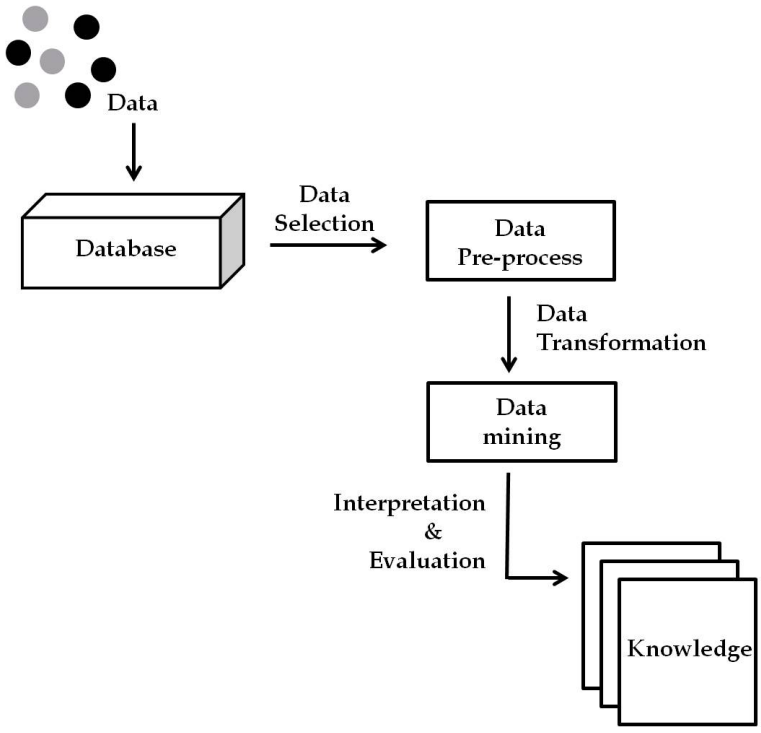

Figure 1.1: Flow chart of the KDD process.

Data mining and machine learning are intimately close and are often confused, as the two areas overlap in many ways. Data mining involves several methods which come from machine learning and artificial intelligence, comprising the following tasks:

- Data summarization
- Association rules
- Clustering
- Change detection
- Classification
- Regression

The underlying motivation for this thesis was prompted by the growing number of problems and applications across research fields within data mining. Namely, the emergence of real-time applications under non-stationary scenarios, such as:

- Real-time monitoring in biomedicine (Petzold et al., 2004; Rodrigues et al., 2011; Sebastião et al., 2013; Yang et al., 1993);
- Quality control in industrial processes (Last, 2002);
- Electricity prediction domain (Bach and Maloof, 2008; Bifet and Gavaldà, 2007);
- Financial surveillance (Frisén, 2008);
- Business and stock market (Bifet and Gavaldà, 2007; Last, 2002);
- Intrusion and fraud detection in computer networks (Kim et al., 2004; Muthukrishnan et al., 2007);
- Spam filtering (Katakis et al., 2009);
- Telecommunication systems (Dasu et al., 2006);
- High-traffic monitoring (Guralnik and Srivastava, 1999);

In this evolving scenario, the evaluation of learning algorithms is a major concern. Moreover, the massive data sets produced must be addressed with feasible summarization techniques. In most of these real world applications, the detection of changes plays an important role. Consequently, in this thesis, the research efforts were also devoted to change detection methods.

1.2 Research Questions

Within the scope of this thesis, the following research questions were proposed.

Research Question 1.

1. In the context of massive data streams, which strategy should be used to remember the discarded data?
2. In the context of time-changing data streams, how can a compact representation of data forget outdated data in order to be able to keep up with the current state of evolving nature?

Research Question 2.

1. In the development of a model to detect changes through the comparison of distributions over two time windows, which is the appropriate step to perform comparisons?
2. When evaluating the distance between distributions, how do the forgetting rates of fading histograms affect the detection delay time?
3. What is the robustness against noise of the proposed change detection model?
4. What is the effect of the extension of a stationary phase in the performance of the proposed change detection model?

Research Question 3.

1. How should the performance of stream learning algorithms be evaluated?
2. Can forgetting mechanisms provide reliable error estimates?
3. How can the performance of learning algorithms in non-static environments be compared?
4. How can forgetting mechanisms be extended to cope with concept drift problems?

Research Question 4.

1. How can a change detection method contribute to a decision support system based on Depth of Anesthesia (DoA) signals?

Research question 1 is addressed in Chapter 3, Chapter 4 is devoted to research question 2, research question 3 is assigned to Chapter 5 and Chapter 6 addresses research question 4.

1.3 Thesis Contributions

The increasing computational capacity of smart devices allows massive amounts of data to be generated at high-speed rate. Subject to these two conditions, the information in the form of transient data streams is promptly processed and discarded immediately. Therefore, the use of online techniques with the ability to process continuous flows of data is essential.

Regarding the discarding of data, the first contribution of this thesis is the construction of a feasible and compact online representation of data in order to summarize huge amounts of information. With regards to the graphical representation of data, which provides useful information about the distribution of a random variable, online histograms, under error constraints, are proposed as a synopsis structure.

Along with this, forgetting out-dated data is also a major concern. In this thesis, the issue "how to forget" is addressed with two approaches:

- Within the data synopses structures, the old data is forgotten using fading factors in the construction of the online histograms. These online fading histograms provide a more up-to-date representation of data than standard ones, allowing old data to be gradually forgotten. Accomplished with fading factors, such histograms, besides allowing discarded data to be remembered, also allow outdated data to be forgotten.

- Related to the change detection problem, after the detection of a change, all past observations are discarded. This results in abrupt forgetting in order to keep up with the current state of nature.

Since data is produced in dynamic environments, the design of a method for change detection in real-time is also one of the contributions of this thesis. The change detection method proposed is based on a windowing scheme, comparing the distance between two data distributions provided by the online histograms mentioned before.

The third contribution of this thesis deals with concerns raised when learning from data streams. To overcome such problems, forgetting strategies are proposed within different assignments. With regard to the evaluation of the performance of stream learning algorithms, sliding prequential and fading prequential error estimates are advanced. Moreover, it is proved that, for consistent learners, the error estimates obtained through the prequential method, through the holdout strategy and over sliding windows converge to the Bayes error. To compare the performance of stream learning algorithms in the flow, the McNemar test applied over such error estimates is a suitable approach. Regarding the detection of concept drift, two approaches based on monitoring forgetting prequential error estimates and on the ratio of these are proposed.

As a final contribution, a real-time algorithm for changes detection in depth of anesthesia signals of patients undergoing surgery is presented.

1.4 Bibliographical Note

Most of the research developed during the course of this doctoral project was partially included in the scope of the following research projects funded by the Fundação para a Ciência e a Tecnologia: KDUDS (PTDC/EIA-EIA/098355/2008) and GALENO (PTDC/SAU-BEB/103667/2008). The enrollment in such projects allowed interactions among members from different research fields, which often leaded to the application of multidisciplinary insights and to contributions at diverse levels.

As a result, part of the work developed during the course of this PhD project has been published in peer reviewed conferences, workshops and in peer reviewed journals.

Besides sharing the research efforts in the area of discourse, publishing and attending conferences in the scope of the PhD project stimulate feedback from expert researchers, enhance interest and reinforce encouragement.

The following set presents the main publications according to the chapters addressing the related contributions:

- **Chapter 3 & Chapter 4**:

  - Sebastião, R., Gama, J., Rodrigues, P., and Bernardes, J. (2010). Monitoring Incremental Histogram Distribution for Change Detection in Data Streams. In Gaber, M. M., Vatsavai, R. R., Omitaomu, O. A., Gama, J., Chawla, N. V.,

and Ganguly, A. R., editors, *Knowledge Discovery from Sensor Data*, volume 5840, chapter 2, pages 25–42. Springer Berlin Heidelberg, Berlin, Heidelberg

– Sebastião, R. and Gama, J. (2009). A study on change detection methods. In Lopes, L. S., Lau, N., Mariano, P., and Rocha, L. M., editors, *New Trends in Artificial Intelligence, 14th Portuguese Conference on Artificial Intelligence*, EPIA'09, pages 353–364

– Sebastião, R., Gama, J., and Mendonça, T. (2008). Learning from data streams: Synopsis and change detection. In Cesta, A. and Fakotakis, N., editors, *STAIRS*, volume 179 of *Frontiers in Artificial Intelligence and Applications*, pages 163–174. IOS Press

– Sebastião, R. and Gama, J. (2007). Change detection in learning histograms from data streams. In *Proceedings of the 13th Portuguese Conference on Artificial Intelligence*, EPIA'07, pages 112–123, Berlin, Heidelberg. Springer-Verlag

- **Chapter 5** :

– Gama, J., Sebastião, R., and Rodrigues, P. P. (2013). On evaluating stream learning algorithms. *Machine Learning*, 90(3):317–346

– Gama, J., Sebastião, R., and Rodrigues, P. P. (2009). Issues in evaluation of stream learning algorithms. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 329–338, New York, NY, USA. ACM

- **Chapter 6** :

– Sebastião, R., Silva, M., Rabiço, R., Gama, J., and Mendonça, T. (2013). Real-time algorithm for changes detection in depth of anesthesia signals. *Evolving Systems*, 4(1):3–12

– Sebastião, R., Martins da Silva, M., Rabiço, R., Gama, J., and Mendonça, T. (2012). Online evaluation of a changes detection algorithm for depth of anesthesia signals. In *Proc. 8th IFAC Symposium on Biological and Medical Systems*, pages 343–348

From an academic point of view, it is worth embracing different research problems. Resulting in contributions as a co-author, the following publications addressed problems such as: regression and decision trees, concept drift detection and decision support and control systems.

- Marques de Sá, J., Sebastião, R., and Gama, J. (2011a). Tree classifiers based on minimum error entropy decisions. *Canadian Journal on Artificial Intelligence, Machine Learning & Pattern Recognition*, 2(3):41–55

- Marques de Sá, J., Sebastião, R., Gama, J., and Fontes, T. (2011b). New results on minimum error entropy decision trees. In *CIARP*, pages 355–362

- Ikonomovska, E., Gama, J., Sebastião, R., and Gjorgjevik, D. (2009). Regression trees from data streams with drift detection. In *Discovery Science*, pages 121–135

- Kosina, P., Gama, J., and Sebastião, R. (2010). Drift severity metric. In *ECAI 2010 - Proceedings of the 19th European Conference on Artificial Intelligence*, pages 1119–1120

- Rodrigues, P., Gama, J., and Sebastião, R. (2010). Memoryless fading windows in ubiquitous settings. In *Proceedings of Ubiquitous Data Mining (UDM) Workshop, in conjunction with the 19th European Conference on Artificial Intelligence - ECAI 2010*, pages 27–32

- Rodrigues, P. P., Sebastião, R., and Santos., C. C. (2011). Improving cardiotocography monitoring: a memory-less stream learning approach. In *Workshop on Learning from Medical Data Streams*, volume 765 - paper 7

- Silva, M. M., Sousa, C., Sebastião, R., Gama, J., Mendonça, T., Rocha, P., and Esteves, S. (2009). Total mass TCI driven by parametric estimation. In *Proceedings of the 2009 17th Mediterranean Conference on Control and Automation*, MED '09, pages 1149–1154, Thessaloniki, Greece. IEEE Computer Society

1.5 Thesis Outline

The thesis is organized as follows:

Chapter 1 - Introduction

The present chapter introduces the context of research and advances the major questions proposed within the doctoral project. It also provides the achieved contributions and a bibliographical note of the publications produced during these years of research.

Chapter 2 - Fundamentals on Evolving Data Streams

The next chapter is devoted to the fundamentals on evolving data streams. This chapter starts with a background on data streams and the standards for learning from data streams are introduced. The problems of distribution change and concept change are exposed and data management methods and adaptation strategies to cope with evolving data are discussed. A broader overview of the existing methods to cope with these problems is presented and the chapter concludes with an evaluation methodology for change detection methods.

Chapter 3 - Histograms over Data Streams

This chapter addresses the problem of constructing histogram representations from data streams. Online histograms, under error constraints, are proposed to create a compact representation of huge amounts of data, allowing properties of discarded data to be remembered. The problem of forgetting outdated data is also addressed and two strategies to cope with this problem are advanced. Therefore, research question 1 is assigned to this chapter.

Chapter 4 - Monitoring Data over Sliding Windows

Chapter 4 proposes the Cumulative Windows Model (CWM), an approach to cope with distribution change and concept drift from time-changing data streams. The performance of this model is evaluated on artificial data, on real data and on a public data set. This change detection model is also compared with state-of-the-art concept drift detection methods. Research question 2 is evaluated under the scope of this chapter.

Chapter 5 - New Criteria for Learning from Data Streams

In this chapter, the problems faced in learning scenarios are addressed. A framework based on forgetting mechanisms is proposed for computing error estimates, to compare, in the flow, the performance of two algorithms and for concept drift detection. This chapter answers proposed research question 3.

Chapter 6 - Application in a Clinical Environment

Chapter 6 extends the problem of online detecting changes in depth of anesthesia signals of patients undergoing surgery. The change detection method is embedded in a real-time software and its performance is evaluated online in the operating room. This chapter is devoted to research question 4.

Chapter 7 - Concluding Remarks

Chapter 7 concludes the thesis, summarizing the contributions achieved and advancing further research directions.

- 2 -

Fundamentals on Evolving Data Streams

"No man ever steps in the same river twice,
for it's not the same river and he's not the same man."
Heraclitus (535 BCE - 475 BCE)

This chapter is devoted to time-changing data streams. After an introduction to the problems raised by evolving data streams, Section 2.2 presents a background on data streams and Section 2.3 introduces concerns when learning from data streams. Next, in Section 2.4 and in Section 2.5, the problems of distribution changes and concept changes are defined and overviewed, respectively. Thereafter, solutions are presented to cope with such data, namely data management and unsighted and sighted strategies to accommodate evolving data, as well as a literature review on methods for change detection. Before concluding the chapter, evaluation metrics to assess the performance of methods for change detection are presented.

2.1 Introduction

The most recent developments in science and information technology have led to the wide spread of the computational capacity of smart devices, which are capable to produce massive amounts of information at high-speed rate, known as data streams. A data stream is a sequence of information in the form of transient data that arrives continuously (possibly at varying times) and is potentially infinite. Along with this, as data flows for long periods of time, the process generating data is not strictly stationary and evolves over time.

The growth of high-speed rate data streams, provided by a wide range of applications,

requires online analysis of the gathered signals. The ability to process examples once at the rate they arrive is essential. It is also of utmost importance to maintain a stream learning model consistent with the most recent data, forgetting outdated data. The dynamics of environments faced nowadays, raise the need for performing online change detection tests. Moreover, changes must be detected as soon as they occur, minimizing the delay between the occurrence of a change and its detection.

2.2 Data Streams Background

The compelling human hunger for research has led man down a path of innovation and, therefore, technology. Indeed, advances in technology have contributed to reducing the costs of computational power, making it available almost everywhere. The improvements in the computational capacity of smart devices, allow them to produce overwhelming amounts of information at high-speed rate. This kind of data is referred to as data streams, and can be viewed as a transient sequence of data that arrives continuously (possibly at varying times) and potentially unlimited in size.

In many modern applications (such as telecommunications, web applications, networking monitoring) information is no longer gathered as finite stored data sets, but as a continuous flow of data. This has imposed constraints on the standards of data mining: as it is impractical to permanently store data streams, it is impossible to apply conventional methods, which require storing the full historic data in the main memory. In the data stream context, the data elements are continuously received, treated and discarded. Moreover, in most cases, the environment evolves over time and data is generated by non-stationary distributions.

The following problems are raised by the main characteristics of the data streams:

- *High-speed rate* - the arrival speed of data implies that each data element needs to be processed in the flow. Therefore, the time for processing each data element is limited, and data stream algorithms must run in real-time.
- *Huge amount of data* - the volume of information is massive and so storing all data is unsuitable. This is overcome by constructing compact summaries of data, which are stored, and discarding the remaining information.
- *Time-changing data* - as data flows over time, changes in the distribution generating examples are expected. Therefore, in order to keep up with the current state of

Table 2.1: Differences between batch and stream learning systems.

		Batch	Streaming
Data	**Data size**	finite data set	open-ended
	Data evolution	static	evolving
	Order of evolution	independent	dependent
System	**Nr. of passes**	multiple	single
	Processing time	unlimited	restricted
	Available memory	unlimited	restricted
	Results accuracy	accurate	approximate

nature, approaches for coping with evolving data are of the utmost importance.

The use of online learning systems is devoted to the first two characteristics, while the third implies data management or adaptation strategies.

2.3 Standards in Learning from Data Streams

The emergence of data streams has posed difficulties for traditional machine learning systems. Considering the high-speed rate and the huge amount of information that characterizes a data stream, traditional models are not suitable for use in this context. Therefore, online learning systems, also known as stream learning systems, must be used to accommodate crucial constraints such as processing time, memory availability and sample size.

Hence, learning systems are roughly divided into two categories: batch learning and stream learning. Within the first category, a large set of examples is provided to the batch learning system and it learns them all at once. In a stream learning system, the examples are provided sequentially, learning them one by one, updating the model as new examples are processed. Table 2.1 presents a summary of the differences between batch and stream learning systems and the data characteristics used in each of them (Gama and Rodrigues, 2007; Gama et al., 2013).

On the assumption that examples are finite, i.i.d. and generated from a stationary distribution, batch learners build static models. Contrary, stream learning systems build models that evolve over time, therefore being dependent on the order of examples generated from a continuous non-stationary flow of data.

Without being concerned about memory availability or the time required to process examples, the accuracy of traditional batch decision models relies on the fact that the whole data set is used for training, allowing multiple passes through the data set to build the final model. On the other hand, stream learning is based on processing examples sequentially, learning in an incrementally manner. Regarding the limitations of the computational resources, stream approaches make only a single pass through the data, processing each example in a short time regardless of the number of examples processed so far. Obviously, as in the stream learning context, the decision model is not rebuilt every time a new observation comes in, the accuracy of results will be close to those of a batch model. Whereas, in a static environment, the stream model is nearly as accurate as the model trained at once on the whole data set (Domingos and Hulten, 2001).

Figure 2.1 depicts an overview of the workflow of a stream learning classifier. At each time an instance arrives, the class prediction is outputted by the learning model. During the classifier training, immediately after the prediction output and before the arrival of the next instance, the true label of the target instance is known. At this stage, the classifier is updated to accommodate the new training example. Such classifiers should be as accurate as necessary, approximating the results obtained with a batch classifier trained in one go on the data seen so far.

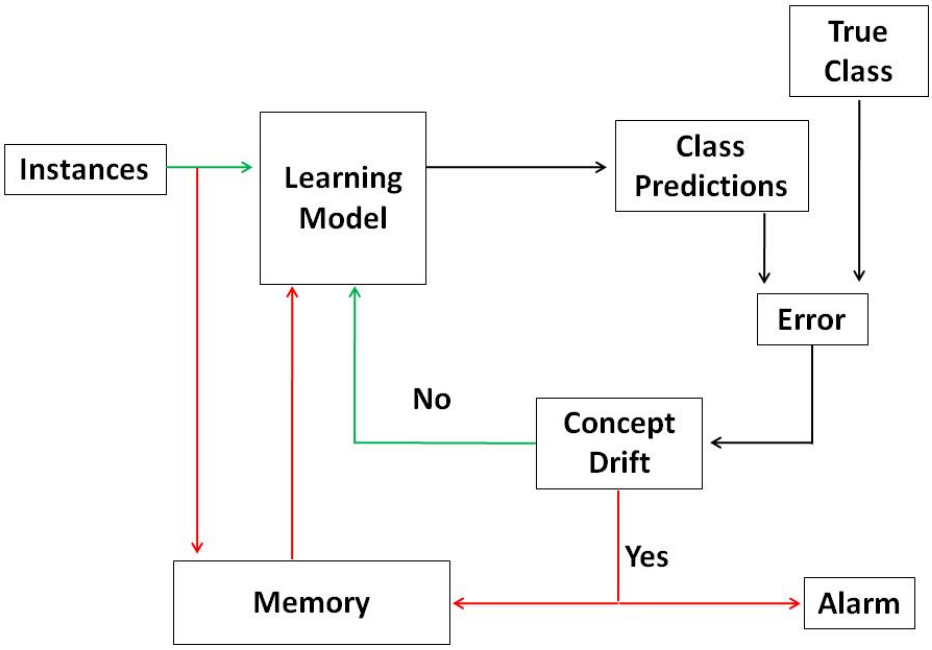

Figure 2.1: Workflow of an online learning system.

Moreover, the knowledge of the true label, also allows the prediction error to be computed.

Through the monitoring of the error rate, concept drifts are assessed. If the concept remains stable, the model learns and predicts the next target instance. Otherwise, if the concept changes, the current learning model is forgotten and a new one is learnt using the most recent examples in the short-term memory.

Hulten et al. (2001) discusses several desirable properties of online learning systems to mine huge continuous and unbounded time-changing data streams. To embrace such data characteristics, they should process examples at once and in short constant time, ideally at the rate of arrival, and using a fixed amount of memory. Moreover, the decision model must be maintained at all times and be adaptable to dynamic scenarios. Frequently in real-world applications, the data is order-dependent and generated according to non-stationary distributions, stressing the need to adapt online learning systems to the status of environment. Accommodating online learning systems in order to cope with evolving data streams, can be done either by data management techniques or adaptation strategies.

In recent years, several learning algorithms have been proposed for handling concept drift, maintaining a model that is consistent with the current state of nature:

- Clustering algorithms (Cormode and Garofalakis, 2007; Rodrigues et al., 2008);
- Incremental decision trees (Gama et al., 2003; Hulten et al., 2001; Street and Kim, 2001);
- Support Vector Machines (Klinkenberg, 2004);
- Rule-based learning (Gama and Kosina, 2011; Widmer and Kubat, 1996);
- Change detection (Bifet and Gavaldà, 2007; Gama et al., 2004);

Along with the flourishing of learning algorithms to cope with evolving data, there are public available systems for mining high-speed time-changing and open-ended data streams, such as: VFML toolkit (Hulten and Domingos, 2003), MOA framework (Bifet et al., 2010) and Rapid-Miner system (Mierswa et al., 2006).

2.4 Distribution Changes

In the dynamic scenarios faced nowadays, it is fundamental to bring out the question: "Is the recent received information from the same distribution observed in the past?"

When data flows over time and for large periods of time, it is an unlikely the assumption that the observations are generated, at random, according to a stationary probability distribution (Basseville and Nikiforov, 1993). Changes in the distribution of the data are expected. As the underlying distribution of data may change over time, it is of utmost importance to perceive if and when there is a change.

2.4.1 Problem Definition

The distribution change detection problem is concerned with the identification of the time of occurrence of a change (or several changes) in the probability distribution of a data sequence. Figure 2.2 illustrates this problem. In this example, P_0 is the probability distribution of the observations seen in the past and P_1 is the probability distribution of the most recent observed data.

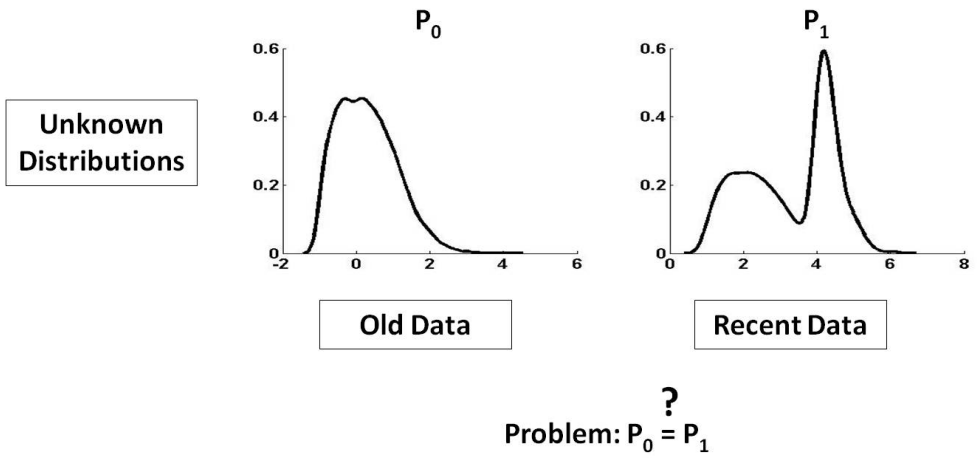

Figure 2.2: Illustration of a distribution change.

Consider that $x_1, x_2, \ldots$ is a sequence of random observations, such that $x_t \in \mathbb{R}, t = 1, 2, \ldots$ (unidimensional data stream). Consider that there is a change point at time t^* with $t^* \geq 1$, such that the subsequence $x_1, x_2, \ldots, x_{t^*-1}$ is generated from a distribution P_0 and the subsequence $x_{t^*}, x_{t^*+1}, \ldots$ is generated from a distribution P_1.

A change is assigned if the distribution P_0 differs significantly from the distribution P_1. In this context, it means that the distance between both distributions is greater than a given threshold.

The change detection problem relies on testing the hypothesis that the observations are generated from the same distribution and the alternative hypothesis that they are generated

from different distributions: $H_0 : P_0 \equiv P_1$ versus $H_1 : P_0 \cong P_1$. The goal of a change detection method is to decide whether or not to reject H_0.

A windows-based change detection method considers two time windows and the data distribution on both windows is monitored and compared to detect any change. It assumes that the observations in the first window of length L_0 are generated according to a stationary distribution P_0 and that the observations in the second window of length L_1 are generated according to a distribution P_1. The null hypothesis is rejected at time t^* when:

$D_{t^*}(P_0||P_1) = \max_{L_0<t} D_t(P_0||P_1) > \lambda$, where λ is known as the detection threshold.

The method outputs that distribution changes at the change point estimate t^*.

Whenever the alternative hypothesis is verified, the change detection method reports an alarm. The correct detection of a change is a hit; a non-detection of an occurred change is a miss or a false positive. Incorrectly detecting a change that does not occur is a false alarm or false negative. An effective change detection method must present few false events and detect changes with a short delay time.

2.4.2 Essence of Distribution Changes

The essence of a distribution change can be categorized according to three main characteristics: rate, magnitude and source.

The rate of a change (also known as speed) is extremely important in a change detection problem, describing whether a signal changes between distributions suddenly, incrementally, gradually or recurrently.

Figure 2.3 presents the mentioned rates of change in the distribution of a single variable. The first plot illustrates a sudden change that takes place when the distribution changes immediately. This is typically exemplified by the buying preferences of customers that change with the season. The second and the third plots show changes that happen slowly over time. In an incremental change, the distribution of the signal changes smoothly along time. Small dysfunctions in parts of an industrial process, which can modify the quality of the final product, are an example of incremental changes. The detection of such changes allows for the correct maintenance of the equipment and the identification of default products. A gradual change is characterized by a time period during which the distributions of the signal change at varying times and for varying periods of time. A typical example of such changes is the slow variation of interests of the users, in a web monitoring process. The last plot represents a recurrent change. Such change happens when a signal

changes from one distribution to another, remaining for a while, and returning again to the former distribution. In this kind of changes, the distribution is expected to reappear at irregular time intervals, without any periodicity attached.

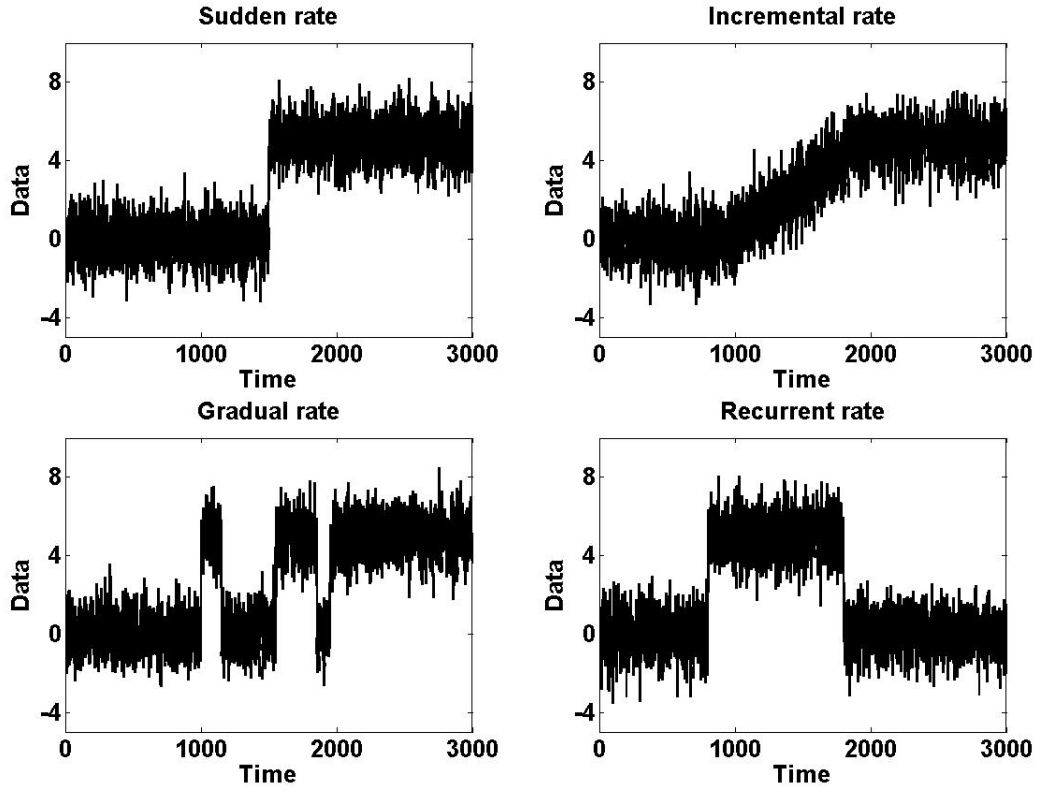

Figure 2.3: Examples of the rate of a change in the distribution of a single variable.

Besides the intrinsic difficulties that each of these kinds of rates impose to change detection methods, real data streams often present several combinations of different rates of change. This is even more challenging since data streams are potentially infinite.

Along with the rate of change, the magnitude of change (also known as severity) is also a characteristic of paramount importance. In the presence of a change, the difference between distributions of the signal can be abrupt or smooth.

Figure 2.4 illustrates the magnitude of change in the distribution of a single variable, as well as the rate of change. The magnitude of the change is drawn alongside the rate of change in order to illustrate that these characteristics describe different patterns, despite being closely related.

The top plots present changes with an abrupt magnitude, where the signal changes from one distribution to another with a great difference. The bottom plots show changes with

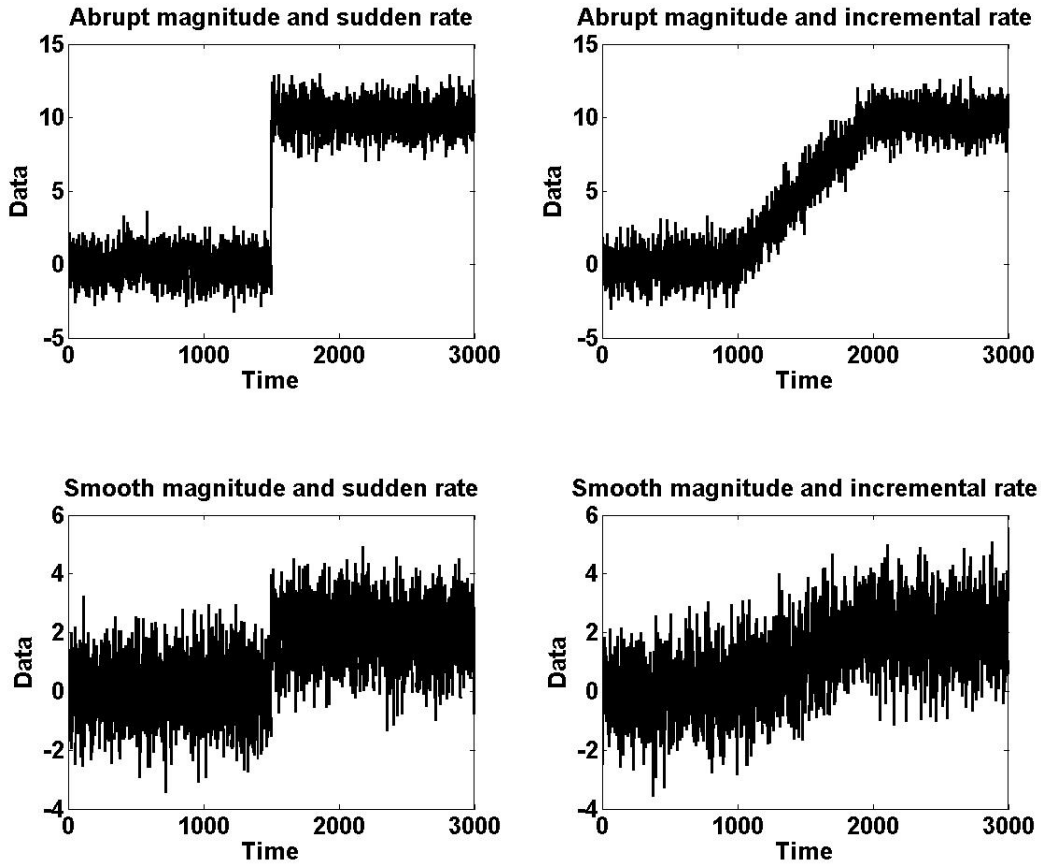

Figure 2.4: Examples of the magnitude and the rate of a change in the distribution of a single variable.

smooth magnitudes, illustrating signals where the distributions before and after the change are quite close.

Abrupt changes are easily observed and detected. Hence, in most cases, they do not pose great difficulties to change detection methods. What is more, these changes are the most critical ones because the distribution of the signal changes abruptly. However, smooth changes are more difficult to be identified. At least in the initial phases, smooth changes can easily be confused with noise (Gama, 2010). Since noise and examples from another distribution are differentiated by permanence, the detection of a smooth change in an early phase, tough to accomplish, is of foremost interest.

The third characteristic of the essence of a distribution change is its source. Besides other features that also describe a distribution of a data set (such as skewness, kurtosis, median, mode), in most cases, a distribution is characterized by the mean and variance. In this sense, a change in data distribution can be translated by a change in the mean or by a

change in variance. Figure 2.5 shows examples of changes in the mean and in the variance occurred in the distribution of a single variable. While a change in the mean do not pose great challenges to a change detection method, a change in the variance tends to be more difficult to detect (considering that both presented similar rate and magnitude).

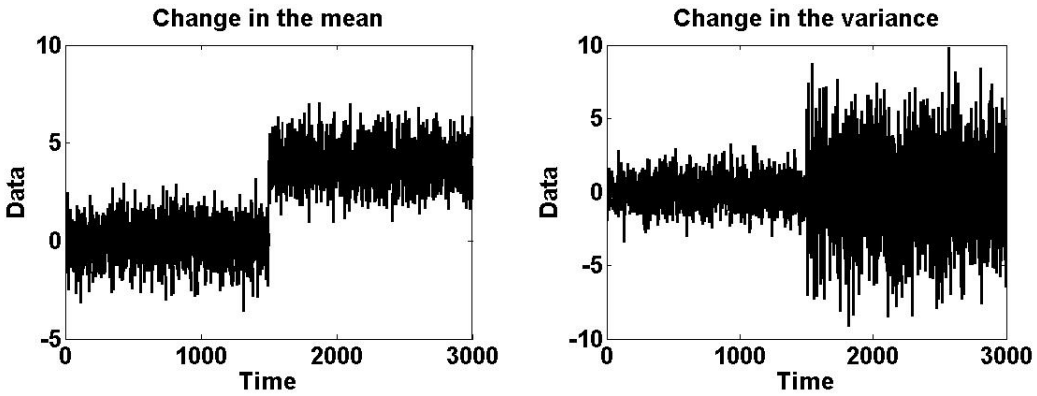

Figure 2.5: Examples of the source of a change in the distribution of a single variable.

2.5 Concept Changes

The concept change problem is found in the field of machine learning and is closely related to the distribution change problem. A change in the concept means that the underlying distribution of the target concept may change over time (Widmer and Kubat, 1996). In this context, concept change describes changes that occur in a learned structure.

2.5.1 Problem Definition

Consider a learning scenario, where a sequence of instances $\vec{X}_1, \vec{X}_2, \ldots$ is being observed (one at a time and possibly at varying times), such that $\vec{X}_t \in \mathbb{R}^p, t = 1, 2, \ldots$ is an instance p-dimensional feature vector and y_t is the corresponding label, $y_t \in \{C_1, C_2, ..., C_k\}$. Each example $(\vec{X}_t, y_t), t = 1, 2, \ldots$ is independently drawn from the distribution that generates the data $P(\vec{X}_t, y_t)$. The goal of a stream learning model is to output the label y_{t+1} of the target instance $\vec{X}_{t+1}$, minimizing the cumulative prediction errors during the learning process. This is remarkably challenging in environments where the distribution that is generating the examples changes: $P(\vec{X}_{t+1}, y_{t+1})$ may be different from $P(\vec{X}_t, y_t)$.

Figure 2.6 illustrates a learning scenario: the top picture shows a static sequence of

instances, while the bottom picture presents an evolving sequence of instants, representing the concept change problem (image design based on Zliobaite (2009)).

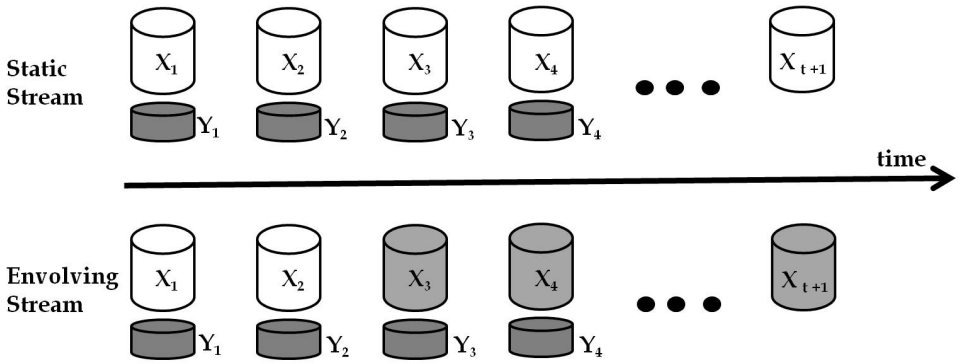

Figure 2.6: Illustration of a concept change.

For evolving data streams, some properties of the problem might change over time, namely the target concept on which data is obtained may shift from time to time, on each occasion after some minimum of permanence (Gama, 2010). This time of permanence is known by context and represents a set of examples from the data stream where the underlying distribution is stationary. In learning scenarios, changes may occur due to modifications in the context of learning (caused by changes in hidden variables) or in the intrinsic properties of the observed variables. Concept change can be formalized as a change in the joint probability distribution $P(\vec{X}, y)$:

$$P(\vec{X}, y) = P(y|\vec{X}) \times P(\vec{X})$$

Therefore, a concept change can be explained through a change in the class conditional probability (conditional change) and/or in the feature probability (feature change) (Gao et al., 2007).

By applying the Bayes theorem, it comes:

$$P(\vec{X}, y) = P(y|\vec{X}) \times P(\vec{X}) = \frac{P(\vec{X}|y) \times P(y)}{P(\vec{X})} \times P(\vec{X}) = P(\vec{X}|y) \times P(y)$$

Therefore, considering the above terminology, due to Kelly et al. (1999) it is possible to identify three ways in which concept change might occur:

1. The class of prior probabilities $P(C_m), m = 1, 2, \ldots, k$ could change over time.

2. The feature conditional probability distributions $p(\vec{X}|C_m), m = 1, 2, \ldots, k$ can change.

3. The class conditional probability distributions $p(C_m|\vec{X}), m = 1, 2, \ldots, k$ may change.

Literature often describes the second case as virtual drift (Widmer and Kubat, 1996), since a change in the class-conditional probability distributions can occur while the target concept remains the same (Tsymbal, 2004), while the last case is identified as real drift.

As well as for the distribution changes, concept change can also be categorized according to the rate, magnitude and source of concept change. Although, regarding the source, a change in the concept can be translated as a change in the mean, variance and correlation of the feature value distribution. Moreover, literature categorizes concept changes into concept drift and concept shift according to the rate and magnitude of the change. A concept drift occurs when the change presents a sudden rate and an abrupt magnitude, whilst a concept shift designates a change with gradual rate and smooth magnitude.

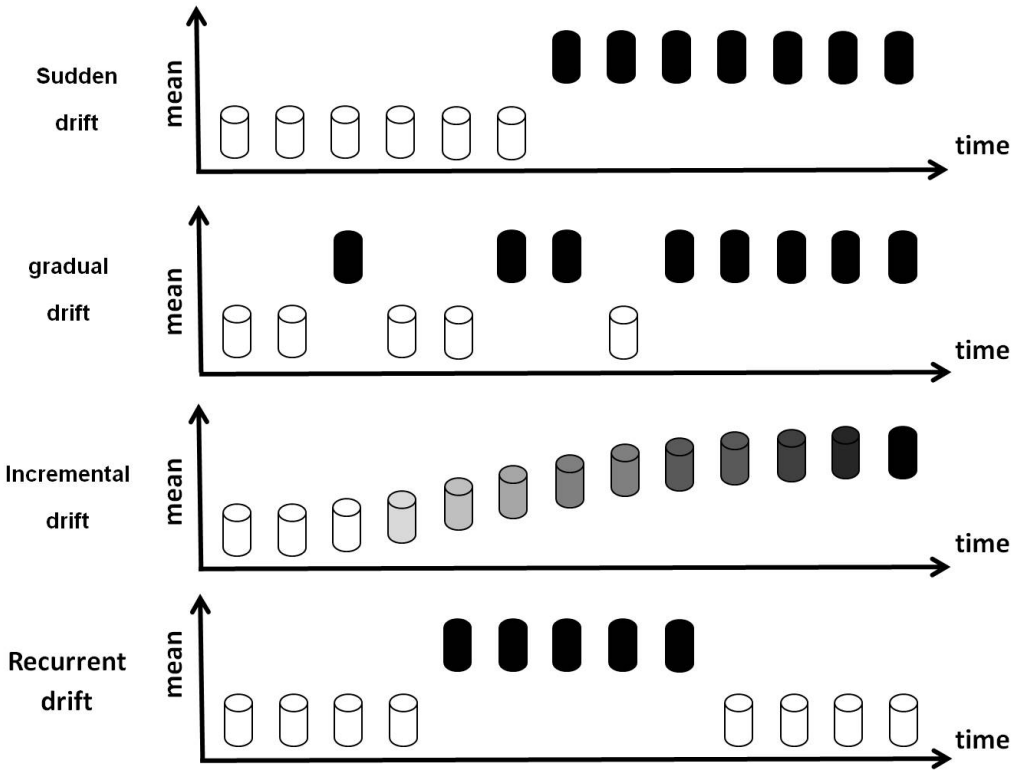

Figure 2.7: Examples of concept changes for an unidimensional data stream.

Figure 2.7 illustrates four kinds of concept change: sudden, incremental, gradual and recurrent (Zliobaite, 2009).

Concept changes can be addressed by assessing changes in the probability distribution (class-conditional distributions or prior probabilities for the classes), changes due to different feature relevance patterns, modifications in the learning model complexity and increases in the classification accuracy (Kuncheva, 2008).

In a supervised learning problem, at each time stamp t, the class prediction $\hat{y}_t$ of the instance $\vec{X}_t$ is outputted. After checking the class y_t the error of the algorithm is computed. For consistent learners, according to the *Probability Approximately Correct* (PAC) learning model (Mitchell, 1997) if the distribution of examples is stationary, the error rate of the learning model will decrease when the number of examples increases.

Detecting concept changes under non-stationary environments is, in most of the cases, inferred by monitoring the error rate of the learning model (Baena-García et al., 2006; Gama et al., 2004; Nishida and Yamauchi, 2007). In such problems, the key to figuring out if there is a change in the concept is to monitor the evolution of the error rate. A significant increase in the error rate suggests a change in the process generating data. For long periods of time, it is reasonable to assume that the process generating data will evolve. When there is a concept change, the current learning model no longer corresponds to the current state of the data. Indeed, whenever new concepts replace old ones, the old observations become irrelevant and thus the model will become inaccurate. Therefore the predictions outputted are no longer correct and the error rate will increase. In such cases, the learning model must be adapted in accordance with the current state of the phenomena under observation.

2.6 How to Cope with Time-Changing Data?

Often, changes make the model built on old data inconsistent with the new data, and regular updating of the model is necessary. Such adaptations intend to keep the decision model updated and can be roughly branched into data management methods and adaptation strategies.

2.6.1 Data Management

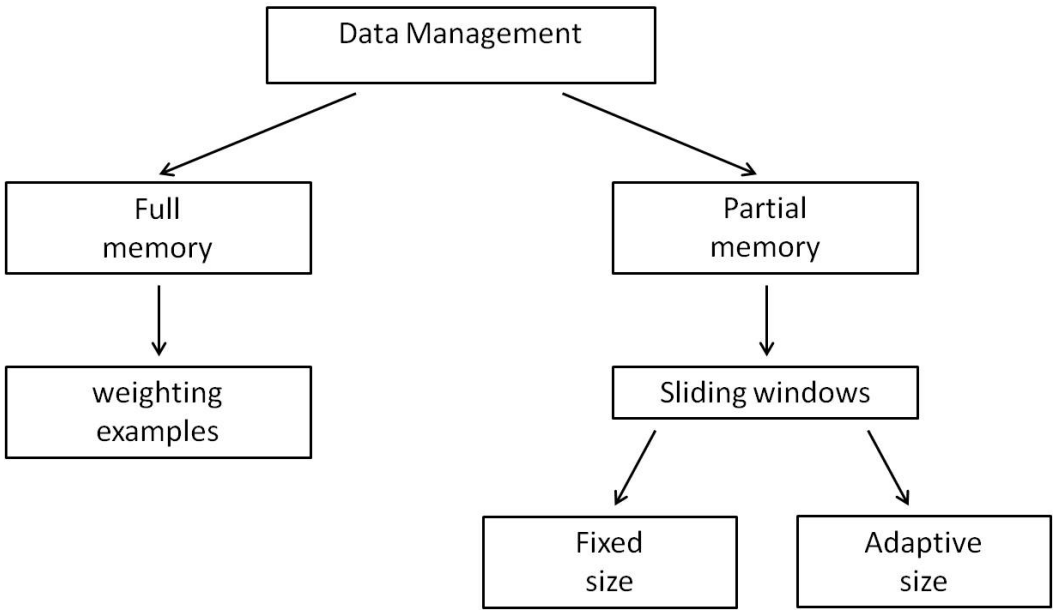

Figure 2.8: Categorization of data management.

In a dynamic environment, as new data is available, older observations are less useful. This underlines out the need to forget out-dated data, which does not describe the current state of the nature. Indeed, a stream learning model must be able to deal with evolving data within the learning process.

When and how to forget are the main questions in this context. Regarding how data is stored in memory to maintain a decision model consistent with the evolving data, data management can be categorized into full memory and partial memory (Gama et al., ress). Figure 2.8 presents the different data management methods to cope with time-changing data.

Full Memory methods keep in memory sufficient statistics over all the observations. These approaches are based on the assumption that relevance of information decays with time and address the evolution of data using decay (or weighting) functions. Such approaches allow old data to be forgotten gradually by assigning less weight to past observations and focusing the attention on the most recent data. A simple approach consists of multiplying the sufficient statistics by a fading factor α ($0 \ll \alpha < 1$), thereafter, older information has a lower contribution to the statistics than newer information.

Forgetting approaches with exponential decay can be found in Klinkenberg (2004) and

Pinto and Gama (2007). In the first approach the aging function is given by $w_\lambda(x) = \exp(-\lambda t_x)$, where observation x was found t_x time steps ago. In the second approach, each example is weighted according to: $w_i = k \times w_{i-1}$, with i being the counter of observations. Koychev (2000) defines a linear gradual decay function, where, at time i, the weight of the observation is given by: $w_i = -\frac{2k}{n-1}(i-1) + 1 + k$. In this decay function, the parameter k $(0 < k < 1)$ represents the decreasing percentage of the weight.

The weighting parameters (λ for the first and k for the last two approaches) controls the importance rate, establishing whether the forgetting process evolves abruptly or smoothly. For larger values of λ and smaller values of k, less weight is assigned to the examples and the lesser is the importance they have. If the value of k is null or the value of λ is equal to one, there is no forgetting in the process, since all observations will have the same weight.

Partial Memory methods keep in memory only the most recent observations, forgetting the oldest at a constant rate, by using a time-window that slides along the data stream. At each time step, as a new observation is added to the sliding window, the oldest one is discarded. Therefore, the learner provides a decision model based only on the learnt examples that are inside the window. The FLORA framework, initially proposed by Kubat (1989), handled the concept drift problem by forgetting data that lies outside a fixed window.

The main difficulty is how to select the appropriate length of the window, establishing a trade-off between good stability of the learning results in static phases and good adaptability to a new concept. If the length of the window is small, the model will be very responsive and will react quickly to changes, assuring a fast adaptability in phases with concept changes, but the accuracy of the classifier might be low due to the small number of observations within the window. Indeed, in more stable phases, it can affect the performance of the learner, since it is learning from a small number of observations and therefore do not provide a full image of the data. On the other hand, using a window with a large length, may result in a slow, but well trained decision model. In such cases, in stationary phases the decision model will output reliable and stable results, but in evolving environments would not be able to promptly react to changes.

To overcome this limitation, approaches with adaptive windows length had been proposed (Bifet and Gavaldà, 2007; Klinkenberg and Renz, 1998), adjusting the length to the extent of the current concept. Therefore partial memory methods can be divided into windows with fixed length and windows with adaptive length. In the adaptive length mechanism, the number of observations inside the window is variable. This approach tries to automatically adjust the length of the sliding window to discover the best trade-off between stability and

adaptability. The FLORA framework has been improved to FLORA 2 in order to provide this type of forgetting mechanism.

Adaptive windows are commonly used in combination with a change detection approach, which defines if the length of the window increases or decreases. Klinkenberg and Renz (1998) proposes adjusting the length of the sliding window progressively according to evaluation measures of the performance of the classifier (namely accuracy, recall and precision).

Discussion

Comparing the ability of data management methods to deal with concept changes, Klinkenberg (2004) found that a full memory approach without any kind of forgetting, while producing the most stable performance in static phases, handles concept drift worse than forgetting approaches, since the recovery phase after the occurrence of a change is too wide. With respect to the partial memory approaches, with the advantage of adjusting the length of the window, adaptive techniques outperform fixed ones, showing a good performance in stationary phases and fast adaptability to new concepts.

Within the partial memory methods, the data that lies outside the sliding window is thrown away which is known as a catastrophic forgetting of old data. With the full memory approach old data is forgotten gradually, considering that although recent data is the most important, old data is still slightly related with the current state of nature.

2.6.2 Adaptation Strategy

Despite aging, the problem of dealing with changes in a signal has caught the attention of the scientific community in recent years due to the emergence of real word applications. When dealing with time-changing data streams, strategies to accommodate the evolving data can be mainly divided into unsighted strategies and sighted strategies, according to Figure 2.9. In unsighted strategies the decision model is adapted to the non-static data without any explicit detection of changes, whilst in sighted strategies explicit change detection is performed.

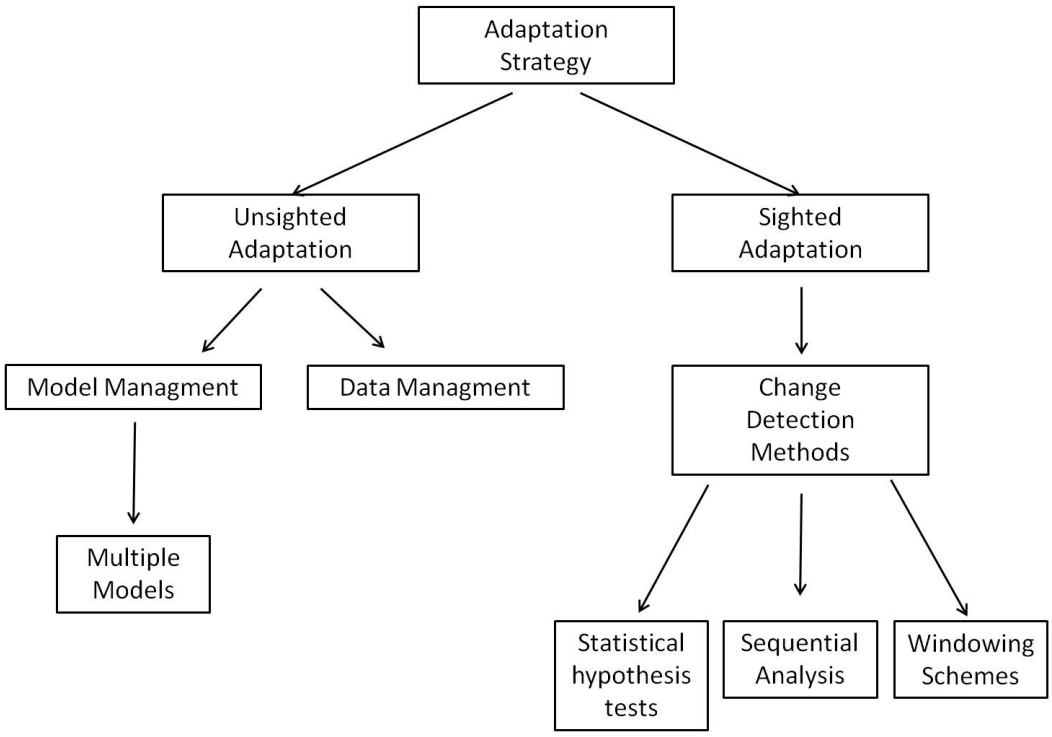

Figure 2.9: Characterization of adaptation strategies.

Unsighted adaptation strategies can be categorized according to data and model managements. The first category adapts the model at regular intervals without any explicit detection of changes. Weighted examples and time windows of fixed length are examples of this strategy.

The model management strategy relies on using multiple decision models, instead of one, known as ensembles. Examples of such approach can be found in Kolter and Maloof (2003); Kuncheva (2008); Zliobaite (2007). The basic idea is that aggregated decisions (by a voting rule) are more accurate than single ones. To meet such goals, the ensemble must be diversified. Obviously, this is a costly computational process as it requires several models to be kept in the memory. It requires, at least, k times more processing than a single model (for an ensemble of k models). Kuncheva (2004) presents an overview on classifier ensembles on evolving scenarios and proposes a new categorization of ensemble approaches.

One of the most important ensemble approaches is the Dynamic Weighted Majority (DWM) presented by Kolter and Maloof (2003), where the predictions are made using a weighted-majority voting rule. Within this ensemble method, learners are trained and weighted online according to their performance. Removals and insertions of learners are made depending

on the individual and overall performances, respectively.

Regarding the increasing concerns about concept detection, Minku and Yao (2012) presents classifiers equipped with a change detector. As it takes out benefits from the usage of a drift detection method, the Diversity for Dealing with Drifts (DDD) deals with time-changing data by explicitly detecting changes. Therefore, this approach is categorized as sighted and will be discussed in the next section.

Sighted strategies adapt the decision model depending on the information provided by a change detection method. Within this strategy, the decision model is modified only after explicit change detection. As a matter of course, sighted adaptation strategies are less computationally costly than unsighted adaptation strategies.

The methods to detect change found in literature can be mainly divided into 3 categories: statistical hypothesis tests, sequential analysis approaches and windowing schemes. These methods will be discussed in the next section. There are methods to detect changes that do not fit in these categories, such as Bayesian approaches (Çelik, 2011) and martingale frameworks (Ho, 2005; Ho and Wechsler, 2010).

2.7 Methods for Change Detection

The growth of high-speed rate and online data, provided by a wide field of applications, requires the online analysis of the gathered signals: especially in those cases where actions must be taken after the occurrence of a change. From this point of view it is essential to detect a change as soon as possible, ideally immediately after it occurs. This reduces the delay time between the occurrence of the change and its detection. Minimizing the detection delay time is of great importance in applications such as real-time monitoring in biomedicine and industrial processes, automatic control, fraud detection, safety of complex systems and many others.

In 1987, Basseville and Nikiforov (1993) addressed the theory of detection of abrupt changes and real word applications. This book got the data mining community interested in this subject. Resulting from applications in many fields, there is a broad number of methods to address the change detection problem. The recently book published, "Quickest Detection" (Poor and Hadjiliadis, 2009) discusses change detection and presents diverse methods to address this problem.

While providing deep insights into the topic under investigation, the wider published literature on change detection methods also highlights difficulties. It is impossible to

review all the contributions and it is ambitious to limit the review process to the major ones. On the other hand, it is of paramount importance to select them. The literature review was undertaken by searching for the main known researchers in the field worldwide and the most important cited work.

2.7.1 Statistical Hypothesis Tests

Statistical hypothesis tests have been developed to address time-series problems. In many time-series analysis problems, the main purpose is signal modulation. As a result, most of the available methods to detect distribution changes in time-series are suitable for use only when data collection precedes analysis. Statistical tests are divided into parametric and non-parametric tests: the great difference between both is the normality assumption of the data in the first case. In statistical hypothesis testing, the null hypothesis is that the previously seen values and the current observed values come from the same distribution. The alternative hypothesis is that they are generated from different distributions.

The parametric Student's t-test (Blair and Higgins, 1980) performs a t-test of the null hypothesis that two data sets are independent random samples from normal distributions with equal means and equal but unknown variances, against the alternative that the means are not equal. The non-parametric alternative, the Wilcoxon signed-rank test (Gibbons and Chakraborti, 2003), can often have better statistical power when the normality assumption does not hold. The Wilcoxon Signed-Rank test for paired samples implements a two-sided rank sum test of the null hypothesis that data in two data sets are independent samples from identical continuous distributions with equal medians, against the alternative that they do not have equal medians. This test cannot be generally used because it will not react to changes in standard deviation or in mean whenever they are dissociated from changes in the median. The non-parametric two-sample Kolmogorov-Smirnov (KS) test (Massey, 1951) determines if two data sets differ significantly. The KS-test has the advantage of making no assumption about the distribution of data (it is distribution free).

Although widely used to detect distribution changes in time-series, statistical hypothesis tests are not commonly found in published literature as suitable ways of detecting distribution changes in data streams. There are some issues that limit the application of statistical hypothesis tests to data streams. Parametric tests are not suitable for use in some data stream contexts as it is unlikely to make assumptions on *a priori* distribution. Along with this, for non-parametric tests, the critical values must be calculated for each distribution and these values cannot always be generated by computer software. Moreover, in the context of open-ended data, data manipulations tend to become more laborious.

Therefore, statistical hypothesis tests are not the best option when the dimension of data is considerably high, which is a common characteristic of streams of data. Ways to overcome some of the problems faced when applying statistical hypothesis tests to data streams are addressed in Sayad (2011). Regarding parametric tests, the author presents several procedures so that statistical tests can be updated online, processing data in the flow.

2.7.2 Sequential Analysis Methods

Derived from Statistical Quality Control applications, methods for monitoring and detecting changes in quality continuous processes have been developed to ascertain distribution changes.

Control charts, or process-behavior charts, were introduced by Shewhart (Shewhart, 1925, 1931) and provide visual information about the state of a monitored process. At regular time intervals, a statistic measuring a chosen quality characteristic is computed using a sample of some fixed size. This statistic is then drawn up along with its mean and standard deviation (computed using all the samples) and upper and lower control limits. Those control limits indicate a detection threshold and whenever the statistic falls outside these limits the process is considered out-of-control. Control charts have proved to be efficient in detecting significant changes in the mean or variance of a process, but fail to detect small changes.

Based on the idea of control charts, the Drift Detection Method (DDM) proposed in Gama et al. (2004) controls the error rate of a stream learning algorithm online. Assuming that classification errors follow a binomial distribution, the proposed approach computes a statistic dependent on the probability of misclassifying an example and its standard deviation, defining warning and drift levels. When the error exceeds the first (lower) threshold, the system enters into a warning mode and stores in a short-term memory (buffer) the examples within the warning level. If the error drops below the threshold again, the warning mode is canceled and the buffer is emptied. However, if in a sequence of examples, the error increases reaching the second (higher) threshold, a change in the concept is declared. The classifier is retrained using only the examples in the buffer and the variables are reinitialized. This method is more suitable for concept drift (sudden and abrupt changes) than for concept shift (smooth changes), since smooth changes can be observed without triggering the alarm level. The Early Drift Detection Method (EDDM) proposed by Baena-García et al. (2006) is a similar method, but monitors the distance error rate (the distance between two consecutive errors) instead of the error rate as the DDM. Although outperforming the DDM in some data sets, the EDDM is not better in detecting

concept shift. To overcome this drawback, the same authors later presented the ADWIN (Bifet and Gavaldà, 2007), which is based on a windowing scheme to detect changes.

Sequential tests, using all previous observations of the process to compute cumulative variables, are more sensitive to detecting small changes over the control charts. Wald (1947) proposed the sequential probability ratio test (SPRT) which monitors the evolution of a cumulative variable defined as the difference between the observed values and the cumulative sum of the log likelihood ratio. Whenever this cumulative variable exceeds a user defined threshold a change is assigned. Muthukrishnan et al. (2007) proposed a method based on SPRT to yield fast and space-efficient change detection on data streams. The main idea is to compare data distributions computing a test statistic based on the logarithm of the ratio between both distributions. Compared to three alternative approaches for change detection found in literature, this method proves to be more effective in detecting changes, namely for query quality and intrusion detection applications.

Later, the cumulative sum method (CUSUM) was proposed by Page (1954). The cumulative variable of CUSUM is computed using the current and previous observations, revealing deviations from a target value. A change is detected if the maximal value between the cumulative variable and zero exceeds a threshold. Although similar to SPRT in theory, the CUSUM variable is initialized with score zero and uses the maximal value between the cumulative variable and zero function as the lower barrier. This method has been revised and recently applied in different contexts (Hadjiliadis et al., 2009; Ross et al., 2009) and associated to Kalman filters (Bifet and Gavaldà, 2006; Severo and Gama, 2006).

As with the CUSUM, the Page-Hinkley Test (PHT) (Hinkley, 1971a,b; Hinkley and Hinkley, 1970; Page, 1954) is a sequential analysis technique typically used for monitoring change detection in the average of a Gaussian signal (Mouss et al., 2004). However, it is relatively robust in the face of non normal distributions. This test considers a cumulative variable defined as the accumulated difference between the observed values and their mean until the current moment. Whenever the difference between this cumulative variable and its maximal value is greater than a given threshold, a change is detected. Although presented as a test to search for increases in the signals behavior, computing the minimal value of the cumulative variable, this test can also be used to detect decreasing trends. Recent applications of this sequential technique (Gama et al., 2013; Hartland et al., 2006) sustain its feasibility.

With the same principles as PHT, the Exponentially-Weighted Moving Average (EWMA) chart, introduced by Roberts (1959), monitors the process mean. However, instead of directly considering the moving average of the signal, it weights examples in geometrically

decreasing order and computes the average using all the examples. Derived from Exponentially Weighted Moving Average, Ross et al. (2012) presents ECDD (EWMA for Concept Drift Detection). This method for detecting concept drift monitors the misclassification rate of a streaming classifier and employs the warning and drift level idea presented in DDM.

Within the context of online ensemble learning, the Diversity for Dealing with Drifts (DDD) approach, proposed by Minku and Yao (2012), presents a high diversity ensemble combined with a change detection method. The drift detection method is based on the idea that, in a static phase, the distance between two consecutive errors increases (as proposed by Baena-García et al. (2006)). Therefore, a concept drift is assigned if this distance decreases considerably with respect to a user defined threshold. The different severities of the learners that compose the ensemble ensure the robustness of the approach and the good stability in static phases, whilst the change detector improves the accuracy in the presence of drifts.

An overview of these methods can be found in Montgomery (2009), as well as deeper insights into control methods for change detection.

2.7.3 Windowing Schemes

Windowing schemes approaches for detecting changes in data, consist of monitoring distributions over two different time-windows, performing tests to compare distributions and decide if there is a change. The simple pseudocode of Algorithm 1 illustrates a change detection method based on a windowing-scheme that compares data distributions (algorithm based on (Kifer et al., 2004)).

In most change detection models, the data distribution on a reference window, which usually represents past information, is compared to the data distribution computed over a window from recent examples (Dasu et al., 2006; Kifer et al., 2004; Sebastião and Gama, 2007). Within a different conception, Bifet and Gavaldà (2007) proposes an adaptive windowing scheme to detect distribution and concept changes: the ADaptive WINDdowing (ADWIN) method. The ADWIN keeps a sliding window W with the most recently received examples and compares the distribution in two sub-windows (W_0 and W_1) of the former. Instead of being fixed *a priori*, the size of the sliding window W is determined online according to the rate of change observed in the window itself (growing when the data is stationary and shrinking otherwise). Based on the use of the Hoeffding bound, whenever two *large enough* sub-windows W_0 and W_1, exhibit *distinct enough* averages, the older sub-window is dropped and a change in the distribution of examples is assigned. When a change is

Algorithm 1 Detect Changes - Windowing Scheme.

Input: Labeled data set: x_1, x_2, ...
  Length of the first window: L_0
  Length of the second window: L_1
Output: Time of the detected changes
  $t \leftarrow 1$
  Step 1:
  for each *window* **do**
    $W_0 = \{x_i : i = t, \ldots, t + L_0 - 1\}$
    compute distribution in W_0: P_0
    $W_1 = \{x_i : i = t + L_0, \ldots, t + L_0 + L_1 - 1\}$
    compute distribution in W_1: P_1
  end for
  Step 2:
  while not at the end of the stream **do**
    for i = 1 ... N **do**
      if $D(P_0, P_1) > \lambda$ **then**
        $t \leftarrow i$
        report a change at time i: $t^* = i$
        Go to **Step 1**
      else
        slide W_1 by 1 observation
        compute distribution in W_1: P_1
      end if
    end for
  end while

detected, the examples inside W_0 are thrown away and the window W slides keeping the examples belonging to W_1. With the advantage of providing guarantees on the rates of false positives and false negatives, the ADWIN is computationally expensive, as it compares all possible sub-windows of the recent window. To cut off the number of possible sub-windows in the recent window, the authors have enhanced ADWIN. Using a data structure that is a variation of exponential histograms and a memory parameter, ADWIN2 reduces the number of possible sub-windows within the recent window.

The windows based approach proposed by Kifer et al. (2004) provides statistical guarantees on the reliability of detected changes and meaningful descriptions and quantification of these changes. The data distributions are computed over an ensemble of windows with different sizes and the discrepancy of distributions between two pairs of windows (with the same size) is evaluated performing statistical hypothesis tests, such as Kolmogorov-

Smirnov and Wilcoxon, among others. Avoiding statistical tests, the adjacent windows model proposed by Dasu et al. (2006) measures the difference between data distributions by the Kullback-Leibler distance and applies bootstrapping theory to determine whether such differences are statistically significant. This method was applied to multidimensional and categorical data, showing to be efficient and accurate in higher dimensions.

Addressing concept change detection, the method proposed by Nishida and Yamauchi (2007) detects concept changes in online learning problems, assuming that the concept is changing if the accuracy of the classifier in a recent window of examples decreases significantly compared to the accuracy computed over the stream hitherto. This method is based on the comparison of a computed statistic, equivalent to the Chi-Square test with Yates's continuity correction, and the percentile of the standard normal distribution. Using two levels of significance the method stores examples in short-term memory during a warning period. If the detection threshold is reached, the examples stored are used to rebuild the classifier and all variables are reset. Later, Bach and Maloof (2008) proposes paired learners to cope with concept drifts. The stable learner predicts based on all examples, while the active learner predicts based on a recent window of examples. Using differences in accuracy between the two learners over the recent window, drift detection is performed and whenever the target concept changes the stable learner is replaced by the reactive one.

The work presented in Kuncheva (2008) goes beyond the methods addressed in this section. Instead of using a change detector, it proposes an ensemble of windows-based change detectors. Addressing adaptive classification problems, the proposed approach is suitable for detecting concept changes either in labeled and unlabeled data. For the labeled data the classification error is recorded and a change is signalled comparing the error on a sliding window with the mean error hitherto. For labeled data, computing the classification error is straightforward, hence it is quite common to monitor the error or some error-based statistic to detect concept drift on the assumption that an increase in the error results from a change. However, when the labels of the data are not available, the error rate cannot be used as a performance measure of drifts. Therefore, changes in unlabeled data are handled by comparing cluster structures from windows with different length sizes. The advantage of an ensemble of change detectors is disclosed by their ability to effectively detect different kinds of changes.

2.8 Evaluation of Change Detection Methods

The evaluation of the performance of change detection method in time-changing environments is quantitatively assessed by measuring the following standard criteria:

- True detections: capacity to correct detects real changes.
- Missed detections: ability to not fail the detection of real changes.
- False alarms: resilience to false alarms when there is no change, which means that the change detection method is not detecting changes under static scenarios.
- Detection delay time: the number of examples required to detect a change after the occurrence of one.

When evaluating the ability of a change detector to effectively detect changes, detection rates are used to quantitatively assess its performance. Such quantities are dependent on the capacity of the change detector to detect real changes and to be resilient in a static scenario. A True Positive (TP) corresponds to a change that actually occurred, while a False Positive (FP) is a wrongly detected change (also known as a type I error and as a false alarm). A False Negative (FN) is a change that is not detected, when in fact one exists (also known as a type II error and as a missed detection). The True Negatives (TN) represents the data observations where the change detector do not detects a change and, indeed, there is no change.

These assessments allow the computation of quality metrics, such as Precision and Recall (also known as Sensitivity), which derive from the area of Information Retrieval. The precision measures the ratio between the correct detected changes (TP) and all the detected changes (TP+FP), while recall is defined as a ratio between the correct detected changes (TP) and all the occurred changes (TP+FN):

$$Precision = \frac{TP}{TP+FP} \qquad Recall = \frac{TP}{TP+FN}$$

For both quality metrics, the closer to 1 the more accurate is the change detection method. Both metrics are closely related to the concepts of type I and type II errors: an algorithm with high recall has a low type II error, which means that it misses a small number of changes detections. While an algorithm with high precision has a low type I error, which means that it is resilient to false alarms.

To jointly provide information on precision and recall of a change detector, the F_1 score, which is the harmonic mean of precision and recall, is commonly used:

$$F_1 = 2\frac{Precision*Recall}{Precision+Recall}$$

The F_1 scores the accuracy of the change detector, and 1 represents its best value and 0 the worst.

2.9 Conclusions

This chapter addresses the main concerns when dealing with data streams gathered in time-changing environments. In fact, data collected from real applications is becoming increasingly evolved. Therefore, the problems of detecting distribution changes and concept changes are of paramount importance when coping with this kind of data. Within these contexts, a literature review on methods for change detection was presented and metrics to evaluate the performance of such methods were described.

- 3 -

Histograms over Data Streams

"Tell me and I forget.
Teach me and I remember.
Involve me and I learn."
Benjamin Franklin (1705/1706 - 1790)

This chapter addresses the problem of constructing compact representations of data: when and why they are useful. These compact representations of data, also known as synopses structures or summaries, are designed to capture properties of the data that is being represented.

It is mandatory to create compact representations of data when dealing with massive data streams. Memory restrictions preclude keeping all received data in memory, making it obligatory that data must be discarded after being processed.

As a result of the summarization process, the size of a synopsis structure is small in relation to the length of the data stream represented. Reducing memory occupancy is of utmost importance when handling a huge amount of data. Along with this, data synopses allow fast and relative approximations to be obtained in a wide range of problems, without the need of accessing the entire stream.

The chapter is organized as follows. It starts by introducing the problem of constructing histograms and giving an overview of some of the existing techniques to address this problem. Section 3.2 proposes an approach to constructing online histograms, under error constraints, from open-ended data streams, which besides allowing properties of data to be remembered, it also provides visual information on data distribution. Section 3.3 presents two strategies, abruptly and smoothly, to forget outdated data. Finally, Section 3.4 concludes this chapter and provides the answers to research question 1.

3.1 Introduction

When very large volumes of data arrive at a high-speed rate, it is impractical to accumulate and archive in memory all observations for later use. Nowadays, the scenario of finite stored data sets is no longer appropriate because information is gathered assuming the form of transient and infinite data streams, and may not even be stored permanently. Therefore, it is unreasonable to assume that machine learning systems have sufficient memory capacity to store the complete history of the stream.

In the data stream context "you only get one look". Processing time, memory and sample size are the crucial constraints in knowledge discovery systems (Barbará, 2002) that handle this complex and interactive type of data. These restrictions impose that in the data stream systems, the data elements are quickly and continuously received, promptly processed and discarded immediately. Since data elements are not stored after being processed it is of utmost importance to create compact summaries of data, keeping only a small and finite representation of the received information.

Data synopses are helpful and critical in these circumstances: they provide a compact representation of data and their size is small compared to the original size of the data under analysis. Besides providing approximated representations of the data that is being processed, the data synopses will allow fast and relative approximations to be obtained in a wide range of problems, such as: range queries, selectivity estimation, similarity searching and database applications, classification tasks, change detection and concept drift.

As for the wide range of problems in which data synopses are useful, it is of paramount interest that these structures have broad applicability. This is a fundamental requirement for using the same data synopsis structure in different applications, reducing time and space efficiency in the construction process. The data stream context under which these synopses are used also imposes that their construction algorithms must be single pass, time efficient and have, at most, space complexity linear in relation to the size of the stream. Moreover, in most cases, data is not static and evolves over time. Synopses construction algorithms must allow online updates on the synopses structures to keep up with the current state of the stream.

Synopses structures for massive data sets are discussed in Gibbons and Matias (1999). Different kinds of summarization techniques are considered in order to provide approximated answers to different queries. The online update of such structures in a dynamic scenario is also discussed. Sampling (Vitter, 1985), hot lists (Cormode and Muthukrishnan, 2005b; Misra and Gries, 1982), wavelets (Chakrabarti et al., 2000; Gilbert et al., 2003; Karras and

Mamoulis, 2008), sketches (Cormode and Muthukrishnan, 2005a) and histograms (Guha et al., 2006; Ioannidis, 2003; Jagadish et al., 1998) are examples of synopses methods to obtain fast and approximated answers.

Within the context of this thesis, the histograms were selected among other synopses structures as a result of their simplicity, efficiency and effectiveness.

3.1.1 Histograms

A histogram is a synopsis structure that allows accurate approximations of the underlying data distribution and provides a graphical representation of data. Histograms are widely applied to compute aggregate statistics, to approximate query answering, query optimization and selectivity estimation (Ioannidis, 2003). Besides, histograms are useful in a large variety of data mining applications, such as change detection and classification tasks. Moreover, for multidimensional problems it is feasible to construct histograms over multiple attributes capturing the joint frequency distributions accurately.

Consisting of a set of k non-overlapping intervals (also known as buckets or bins), a histogram is visualized as a bar graph that shows frequency data. The height of each bar drawn on each interval is proportional to the number of observed values within that interval.

There is a broad number of works proposing histograms as a feasible data summarization technique. The most important developments in this synopsis structure are discussed in Ioannidis (2003). Also Poosala et al. (1996) presents a study on histograms, proposing a taxonomy to capture existing histograms types and deriving new ones.

Often, literature identifies several classes of histograms:

- In **equi-width** histograms, the range of the variable is divided into k intervals of equal length (the range of the variable is equalized).
- **Equi-depth** or **equi-height** histograms are constructed so that the range of the observed variable is divided into k buckets such that the frequency of the values in all buckets should be equal.
- **V-optimal** histograms are defined as those that minimize the variance of the difference between the observed values and the approximations given by the corresponding assigned bucket. A V-optimal histogram is the histogram with the least variance of all histograms with the same number of buckets.

- In **maxDiff** histograms, after sorting the data, a bucket boundary is placed between two adjacent values, if the difference between them is one of the $k-1$ largest.
- **Compressed** histograms begin by storing in l ($l < k$) buckets the most frequent l source values. The remaining are divided into $k-l$ buckets as proposed by the equi-depth method.

To construct histograms in the stream mining context, there are some requirements that need to be fulfilled: the algorithms must be one-pass, supporting incremental maintenance of the histograms, and must be efficient in time and space (Guha et al., 2006, 2004).

Therefore, when constructing online histograms from data streams there are two main characteristics to embrace:

- The updating facility.
- The error of the histogram.

It is well known that the V-optimal histograms, due to having the smallest variance possible among all the intervals, provide smaller errors (Poosala et al., 1996). Although presenting the most accurate estimation of data, V-optimal histograms are very difficult to update as new data arrives. In fact, keeping a V-optimal histogram along with the data could imply reconstructing the histogram entirely instead of updating the existing one. Regarding practical issues, the work in Ioannidis and Poosala (1995) devoted efforts to improve the online maintenance of V-optimal histograms.

Addressing the same problematic concerns, in Gama and Pinto (2006) the *Partition Incremental Discretization* algorithm provides a histogram representation of high-speed data streams. The advantage of such an algorithm, which uses an architecture composed by two layers, is that any base discretization method can be used: equal frequency, equal width, recursive entropy discretization, chi-merge, etc.

3.2 How to Remember?

In this thesis, the synopses structures used to remember data that is being discarded are provided by online equi-width histograms, in which the number of buckets is chosen under error constraints. Despite not presenting the smallest error, the equi-width histograms were chosen based on the following reasons:

- The construction is effortless: it simply divides the range of the random variable into k non-overlapping intervals with equal width.
- The updating process is easy: each time a new data observation arrives, it just identifies the interval where it belongs and increments the count of that interval.
- Information visualization is simple: the value axis is divided into buckets of equal width.

3.2.1 Online Histograms under Error Constraints

A histogram provides a data summarization showing a graphical representation of the distribution of a random variable: the values of the random variable are placed into non-overlapping intervals and the height of the bar drawn on each interval is proportional to the number of observed values within that interval.

Let i be the current number of observations of a given variable X from which a histogram is being constructed. A histogram H_k is defined by a set of k buckets $B_1, \ldots, B_k$ in the range of the random variable and a set of frequency counts $F_1(i), \ldots, F_k(i)$.

Definition 3.1. Let k be the number of non-overlapping intervals of a histogram. For each observation x_i of a given variable X, the histogram counters are defined as:

$$C_j(i) = \begin{cases} 1, & x_i \in B_j \\ 0, & x_i \notin B_j \end{cases}, \quad \forall j = 1, \ldots, k \tag{3.1}$$

Along with the definition of the histogram counters, come the definitions of the histogram counts and histogram frequencies.

Definition 3.2. Let k be the number of non-overlapping intervals of a histogram. For each observation x_i of a given variable X, the histogram counts are defined as:

$$Ct_j(i) = \sum_{l=1}^{i} C_j(l), \quad \forall j = 1, \ldots, k \tag{3.2}$$

Definition 3.3. Let k be the number of non-overlapping intervals of a histogram. For each observation x_i of a given variable X, the histogram frequencies are defined as:

$$F_j(i) = \frac{Ct_j(i)}{i}, \quad \forall j = 1, \ldots, k \tag{3.3}$$

The set of k buckets of a histogram are defined by a set of break points $bp_1, \ldots, bp_{k+1}$, which divides the range of the variable, and by a set of middle break points $m_1, \ldots, m_k$.

Definition 3.4. Let k be the number of non-overlapping intervals of a histogram. The histogram buckets are defined as:

$$B_j = \{bp_j, bp_{j+1}, m_j\}, \quad \forall j = 1, \ldots, k \tag{3.4}$$

One of the main problems in using histograms is the definition of the number of buckets. A histogram with too many buckets is over detailed. On the other hand, if the histogram is constructed with too few buckets, important information may not be represented. Either way, a wrong number of the histogram buckets do not allow the underlying distribution of the data to be perceived.

Several rules exist to decide the number of buckets in a histogram. A rule that has been widely used is the Sturges's rule (Sturges, 1926): $k = 1 + log_2 n$, where k is the number of intervals and n is the number of observed data points. This rule has been criticized because it implicitly uses a binomial distribution to approximate an underlying normal distribution. Sturges's rule has probably survived because, for moderate values of n (less than 200) produces reasonable histograms. Although, it does not work for a large number of observations. Alternative rules for constructing histograms include Scott's rule (Scott, 1979) for the class width: $h = 3.5sn^{-1/3}$ and Freedman and Diaconis's rule (Freedman and Diaconis, 1981) for the class width: $h = 2(IQ)n^{-1/3}$ where s is the sample standard deviation and IQ is the sample interquartile range. However, these rules are not suitable to use in the context of open-ended data streams. In this context, all values of the variable are never observed, and therefore the total number of observations is unknown. Overcoming this drawback, the number of buckets in the online equi-width histograms proposed is defined establishing a bound on the mean square error of the histogram.

While estimating the probability distribution of a random variable, a histogram presents an error due to the reason that all the values of the variable within an interval are represented by the corresponding middle point. In each bucket, all the contained observations x_i are estimated by the corresponding middle point, which means that this approximation error is bounded by half of the width (W) of the interval:

$$x_i - m_j \leq \tfrac{W}{2}, \quad bp_j \leq x_i < bp_{j+1} \text{ and } \forall j = 1, .., k.$$

Considering an equi-width histogram, the range of the random variable, R, is divided into k non-overlapping intervals with equal width ($W = \frac{R}{k}$). For each observation x_i, the bound

for the approximation error is:

$$x_i - m_j \leq \frac{R}{2k}, \quad bp_j \leq x_i < bp_{j+1}, \quad \forall j = 1, .., k. \tag{3.5}$$

Therefore, the error of representing a random variable by an equi-width histogram is defined as a function of this approximation error.

Definition 3.5. The square error[*1] of the bucket B_j, defined by the interval $\{bp_j, bp_{j+1}\}$ and the middle break point m_j is given by:

$$SE(B_j) = \sum_{bp_j < x_l \leq bp_{j+1}} (x_l - m_j)^2, \quad \forall j = 1, \ldots, k. \tag{3.6}$$

Moreover, since a histogram is defined by a set of non-overlapping buckets and each data observation belongs only to one bucket, the total error of a histogram can be expressed as a sum of the bucket errors.

Definition 3.6. The mean square error of a histogram H_k with buckets $B_1, \ldots, B_k$, and a total of n observations, is the mean of the sum of the square errors along all the buckets:

$$MSE(H_k) = \frac{\sum_{j=1}^{k} SE(B_j)}{n} \tag{3.7}$$

Using equation 3.5, the square error of each bucket B_j is at most $C_j \frac{R^2}{4k^2}$:

$$SE(B_j) = \sum_{bp_j < x_l \leq bp_{j+1}} (x_l - m_j)^2 \leq \sum_{bp_j < x_l \leq bp_{j+1}} \left(\frac{R}{2k}\right)^2 = C_j \frac{R^2}{4k^2}, \forall j = 1, \ldots, k \tag{3.8}$$

where C_j is the count of bucket B_j, $\forall j = 1, \ldots, k$.

After simple algebraic manipulation, the mean square error of an equi-width histogram H_k is bounded, in the worst case, by $\frac{R^2}{4k^2}$:

$$MSE(H_k) = \frac{\sum_{j=1}^{k} SE(B_j)}{n} \leq \frac{\sum_{j=1}^{k} C_j \frac{R^2}{4k^2}}{n} = \frac{R^2}{4k^2} \tag{3.9}$$

[*1] The square error is one of the most used error measures in histogram construction. It is also known as the V-Optimal measure and was introduced by Ioannidis and Poosala (1995).

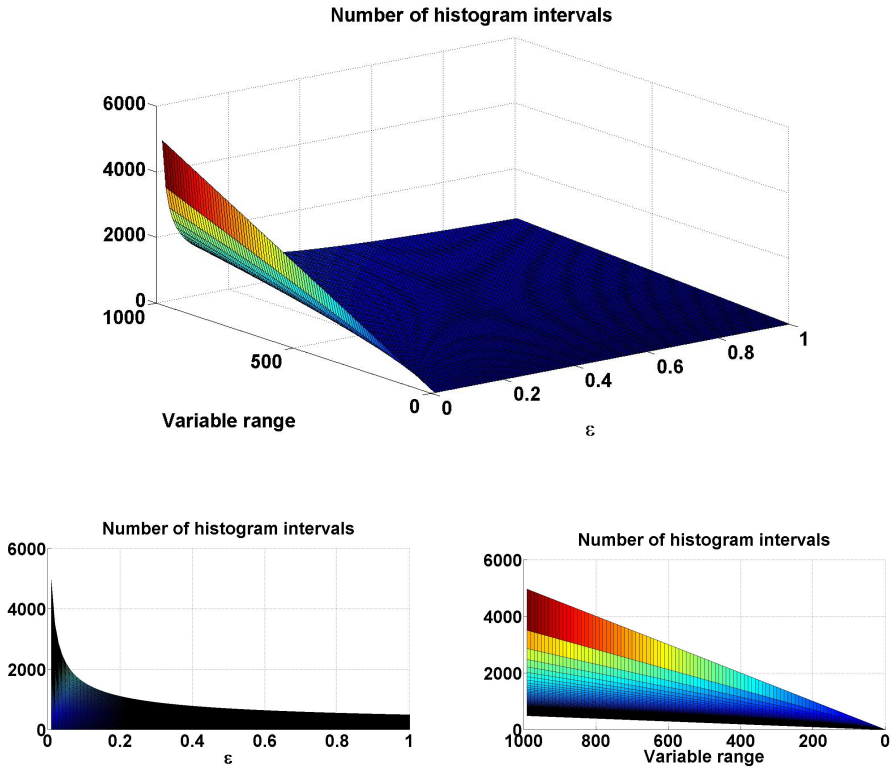

Figure 3.1: Representation of the number of non-overlapping intervals. The top figure shows the dependency from the admissible error ϵ and the variable range R. Bottom figures show it according to only one variable.

Let ϵ be the admissible error for the mean square error of a histogram H_k with k buckets. Then, from equation 3.9, it turns out that $\frac{R^2}{4k^2} \leq \epsilon$. And therefore, this guarantees that the mean square error of any equi-width histogram with at least $\frac{R}{2\sqrt{\epsilon}}$ buckets is, at most, ϵ.

It must be stressed out that the constraint $k \geq \frac{R}{2\sqrt{\epsilon}}$ does not guarantee that the equi-width histogram H_k is optimal, neither that, concerning the mean square error, k is the optimal number of buckets.

Figure 3.1 shows that the number of buckets linearly increases with the variable range (R) and when the admissible mean square error (ϵ) of the histogram decreases. Figure 3.1 (top) represents the number of buckets in function of ϵ and of R. The bottom figures give a projection of the number of buckets according to the variables ϵ and R (respectively).

Therefore, in this thesis, the input for the online equi-width histograms construction is the error parameter ϵ and the range of the variable. The range of the variable is only indicative and is used to define the non-overlapping intervals using an equi-width strategy

and to establish the number of buckets in the histogram. Each time a new value of the variable is observed, the histogram is updated. The updating process determines the bucket corresponding to the observed value and increments the counts. These updates are processed online, performing a single scan over the data stream. It can process infinite sequences of data, processing each example in constant time and space. Keeping the histogram up with the incoming data only requires the set of buckets, the set of counts and the current number of observations seen so far to be stored in memory.

3.3 How to Forget?

In a dynamic environment, as new data is available, older observations are less useful. This stresses the need to forget outdated data, which is not describing the current state of the nature.

In this context, the forgetting approach is deeply related with the evolution of the process generating data. The forgetting process can be done abruptly or smoothly. In the former, the data seen so far is thrown away, which is known by a catastrophic forgetting of old data. This approach can be applied either after a fixed period of time or after the occurrence of a change. The second case, considers that although recent data is the most important, old data is still slightly related with the current state of nature. Within this approach, old data is forgotten gradually, using fading factors.

In this thesis, evolving data streams are summarized using online histograms. With the construction of online histograms over a sliding window, a catastrophic forgetting of outdated data is achieved. On the other hand, a smooth forgetting mechanism is performed using fading histograms constructed over the entire stream.

3.3.1 Histograms over Sliding Windows

The most common approach to forgetting outdated data is sliding windows, where an algorithm is applied only to a small contiguous portion of the data set (window) which is moved (slide) along the data examples as they arrive. In a sliding windows approach, at each time step, the data distribution is approximated by a histogram constructed only from the observations that are included in the window. At every time i, a data record arrives and expires at time $i + w$, where w is the length of the window.

Histograms that are constructed over a sliding window, are, henceforth, referred to as

sliding histograms. Sliding histograms allow attention to be focused solely on the most recent data, which captures the current state of nature and therefore the current properties of the process generating data.

Definition 3.7. Consider a sliding window (SW) of length w at observation x_i:

$$SW = \{x_l : l = i - w + 1, \ldots, i\}.$$

The frequency counts of a sliding histogram (with k buckets) constructed at observation x_i over this window are defined by:

$$F_{w,j}(i) = \frac{\sum_{l=i-w+1}^{i} C_j(l)}{w}, \forall j = 1, \ldots, k, \tag{3.10}$$

To exemplify the forgetting ability of sliding histograms with respect to histograms constructed over the entire stream, artificial data was generated from two normal distributions. The initial 2500 observations follow a normal distribution with mean 5 and standard deviation 1 and the remaining 2500 observations follow a normal distribution with mean 10 and the same standard deviation.

For illustrative purposes, the number of buckets in each histogram was set to 20 (considering an admissible error $\epsilon = 0.1$ for the mean square error of the histograms and using equation 3.9). With respect to the sliding histograms, a window of length 1000 was used.

Figure 3.2 (top) shows the artificial data with a change at observation 2500. The remaining plots display sliding histograms (constructed over a window of length 1000) and over the entire stream, at different observations: 2000, 3000 and 4000.

From the first representations, while in the presence of a stationary distribution, it turns out that both histograms produce similar representations of the data distribution. The size of 1000 examples of the sliding window is enough to capture the behavior in this initial stage. The second and the third representations present a different scenario. At observations 3000, after the change occurred at observation 2500, the representations provided by both histograms strategies become quite different. It can be observed that in the sliding histogram representation, the buckets for the second distributions are reinforced, which does not occur on the histogram constructed over the entire stream. Furthermore, at observation 4000, in the histogram constructed over the entire stream, the buckets for the first distribution still filled, which is not in accordance with the current state of nature.

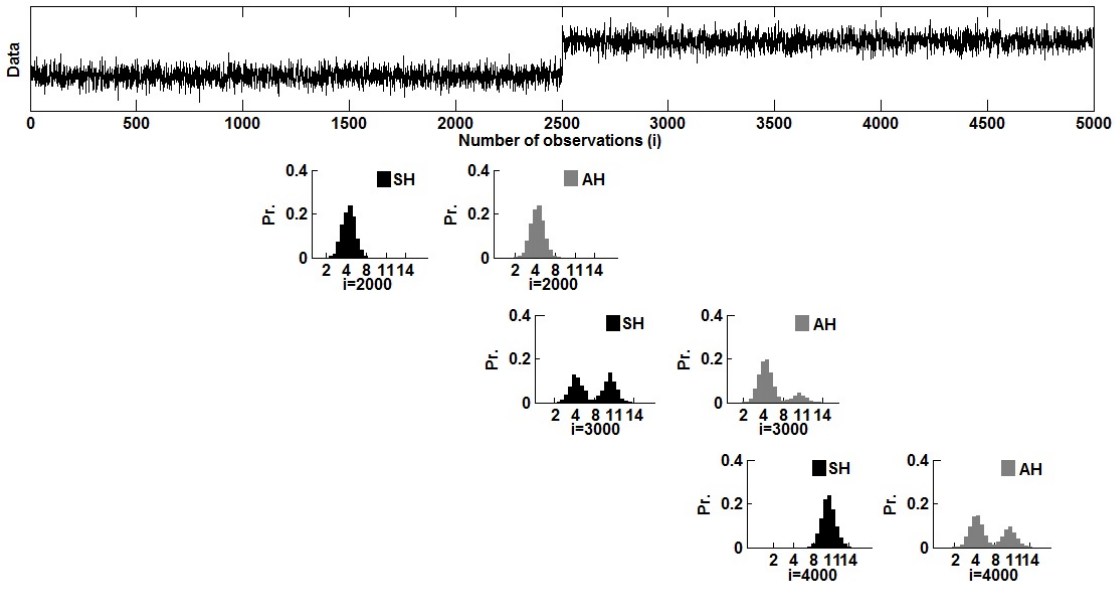

Figure 3.2: Comparison of histograms constructed over a sliding window of length 1000 (SH) and over the entire stream (AH).

While in the sliding histograms, those buckets are empty as a result of the fact that the sliding histogram is constructed only with the observations from a window on the current distribution.

However, to construct sliding histograms it is necessary to maintain in memory all the observations within the window, which could be costly depending on the applications and on the length of the window. Nevertheless, approaches based on sliding windows have been widely applied in summarization problems (Babcock et al., 2002; Lin et al., 2007). The next section advances a memoryless approach to forgetting outdated data, which avoids keeping recent observations in the memory.

3.3.2 Fading Histograms

As stated before, with the previous approach a catastrophic forgetting is obtained, which in some applications could turn out to be excessive. As an alternative, using fading factors, data is forgotten gradually. In dynamic data streams, recent data is usually more important than old data. Therefore, the most feasible way to attribute different weights to data observations is an exponential approach, where the weight of data observations decreases exponentially with time. Exponential fading factors have been applied successfully in data

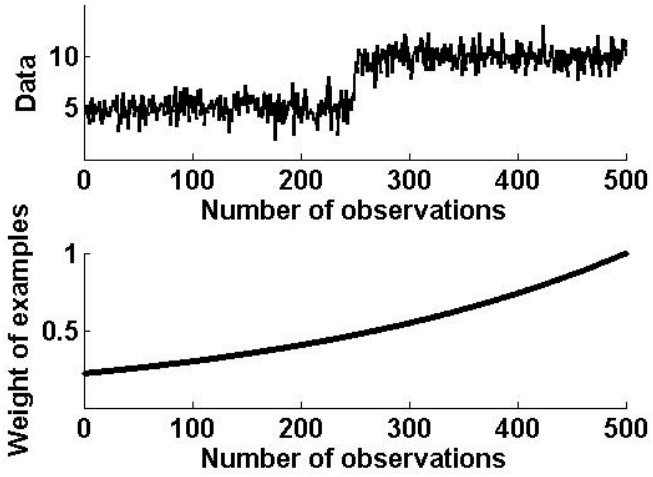

Figure 3.3: The weight of examples as a function of age, in an exponential approach.

stream evaluation (Gama et al., 2013).

Figure 3.3 illustrates the weight of examples according to their age, considering an exponential approach.

Following an exponential forgetting, histograms can be computed using fading factors, henceforth referred to as fading histograms. In this sense, data observations with high weight (the recent ones) contribute more to the fading histogram than observations with low weight (the old ones).

Definition 3.8. Let k be the number of buckets of a fading histogram. For each observation x_i of a given variable X, the α-frequencies are defined as:

$$F_{\alpha,j}(i) = \frac{\sum_{l=1}^{i} \alpha^{i-l} C_j(l)}{\sum_{j=1}^{k} \sum_{l=1}^{i} \alpha^{i-l} C_j(l)}, \forall j = 1, \ldots, k, \tag{3.11}$$

where α is an exponential fading factor, such that $0 \ll \alpha < 1$.

According to this definition, old data is forgotten gradually, since it contributes less than recent data. Moreover, the recursive form enables the construction of fading histograms in the flow.

The forgetting ability of fading histograms with respect to histograms constructed over the entire stream is illustrated using the same artificial data as in figure 3.2. The number of

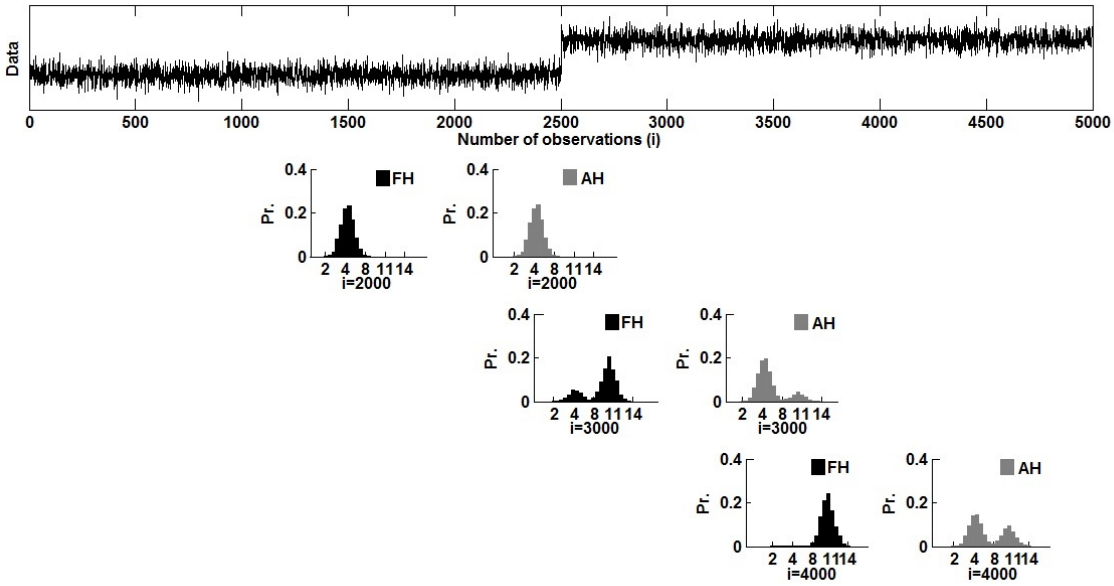

Figure 3.4: Comparison of histograms computed with a fading factor of $\alpha = 0.997$ (FH) and histograms constructed over the entire stream (AH).

buckets and the value of the fading factor for the fading histograms were set to 20 (the same settings as in Section 3.3.1) and $\alpha = 0.997$, respectively.

The top Figure 3.4 shows the artificial data with a change at observation 2500. The remaining plots display fading histograms (with a fading factor of $\alpha = 0.997$) and histograms constructed over the entire stream, at different observations: 2000, 3000 and 4000. The number of buckets in each histogram was set to 20 (the same settings as in Section 3.3.1).

These histograms are similar to those shown in Figure 3.2, and therefore the same conclusions can be taken for the fading histograms as those for the sliding histograms. Contrary to the histograms constructed over the entire stream, fading histograms capture the change better (as shown in the histograms produced at observation 3000), enhancing the fulfillment of the buckets for the second distribution. At observation 4000, it can be seen that the fading histogram produces a representation that keeps up with the current state of nature, forgetting outdated data (it must be pointed out that although they appear to be empty, the buckets for the first distribution present very low frequencies).

The fading factors are memoryless, an important property in streaming scenarios (Gama et al., 2013). Hence, it turns out that fading histograms present a clear advantage over sliding histograms, which require all the observations inside the window to be maintained in memory. Nevertheless, fading histograms are, in fact, approximations of histograms

constructed with data observations from a sliding window (for further details, see the Appendix A). Moreover, there is a relation between the length of the sliding window and the value of the fading factor that should be used to approximate that window (Rodrigues et al., 2010).

3.3.3 Representation Comparison

In order to compare sliding histograms with fading histograms, the same artificial data as in Figure 3.2 was used. With respect to the sliding histograms, a sliding window of length 1000 was used and to compute fading histograms, a fading factor of $\alpha = 0.997$ was used (from Rodrigues et al. (2010) it turns out that to approximate an estimate on a sliding window of length w with a fading estimate, the α must be set to $\alpha = \epsilon^{\frac{1}{w}}$, where ϵ is the admissible approximation error). The number of buckets in each histogram was set to 20 (the same settings as in Section 3.3.1).

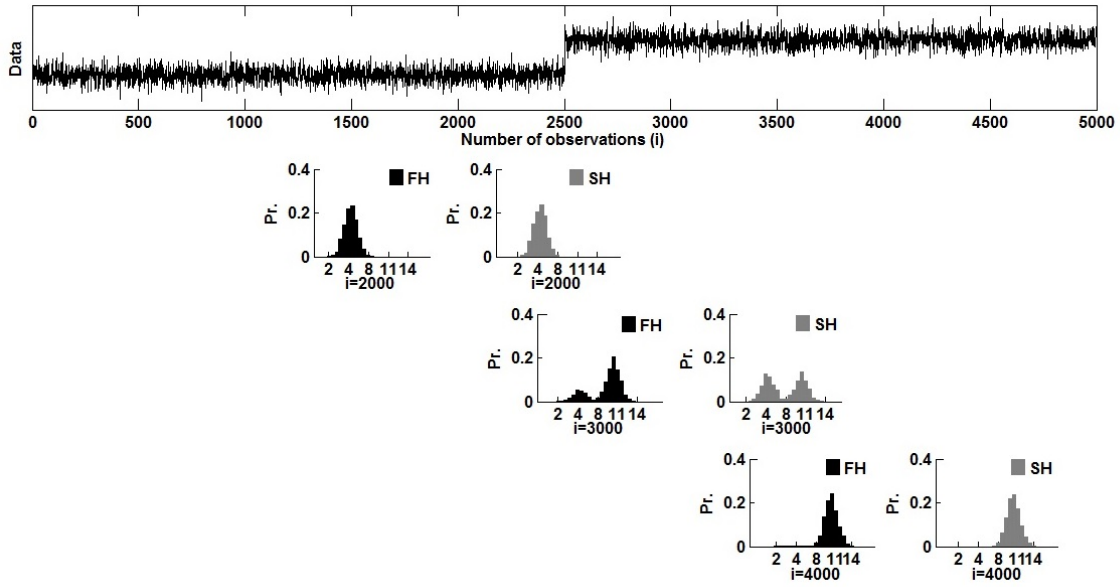

Figure 3.5: Comparison of histograms constructed over a sliding window (of size 1000) and with fading factors ($\alpha = 0.997$).

Figure 3.5 (top) shows artificial data with a change at observation 2500. The remaining plots display sliding histograms (constructed over a sliding window of length 1000) and fading histograms (with $\alpha = 0.997$) at different observations: 2000, 3000 and 4000.

From these representations, it is straightforward that both approaches to constructing

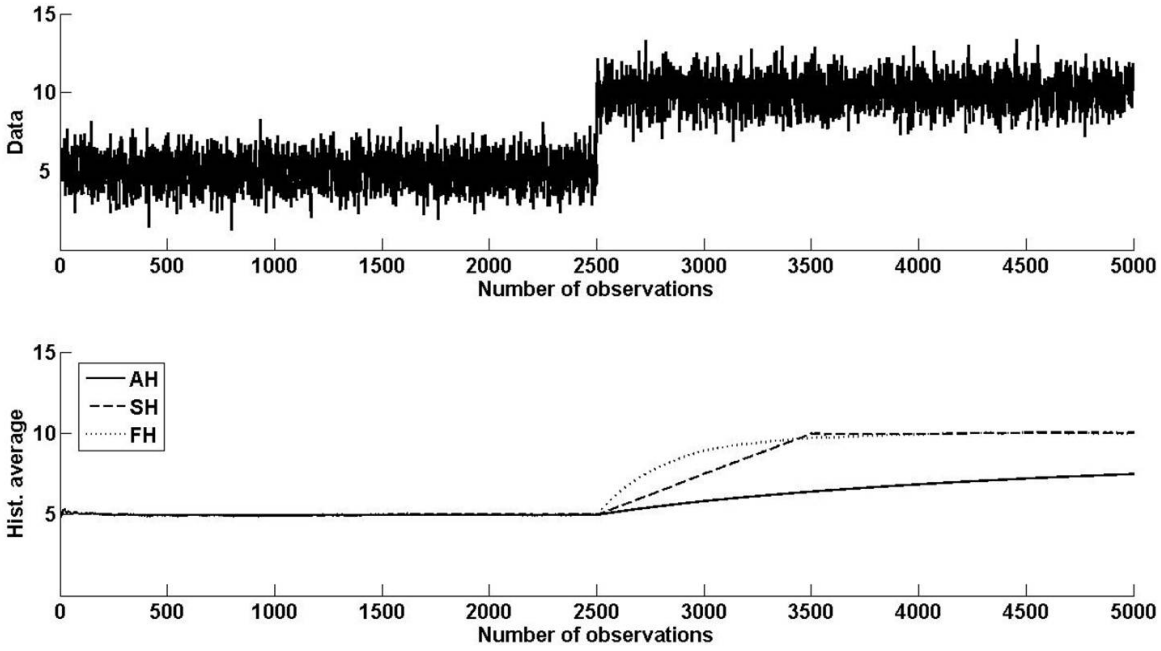

Figure 3.6: Comparison of the average counts of different kinds of histograms: a histogram constructed over the entire stream of data (solid thick line), a sliding histogram constructed over a sliding window of size 1000 (long dashed thick line) and a fading histogram computed with fading factor $\alpha = 0.997$ (dashed thick line).

histograms with a forgetting ability lead to similar results. Moreover, both histograms capture the change at observation 2500, as shown in the representations at observation 3000. However, it can be observed that the ability to forget outdated data is reinforced in the fading histogram, since the buckets from the initial distribution presented smaller values than the corresponding ones in the sliding window, while the buckets from the second distribution have higher values.

Figure 3.6 (top) shows the same artificial data, with a change at observation 2500, used in the previous figures. The bottom plot presents the average of histogram counts from a histogram constructed over the entire stream of data (solid thick line), from a sliding histogram (long dashed thick line) and from a fading histogram (dashed thick line). In the presence of a change in data distributions, the advantage of using sliding or fading histograms instead of a histogram constructed with all the examples is obvious: both adapt their data representation to the occurred change, keeping a representation which is more faithful to the new state of the data distribution. Indeed, the histogram constructed over the entire stream of data, is not able to overcome the contributions of the initial state of data distribution, and therefore the representation of new data provided is corrupted by the initial examples. In this sense, it is quite difficult to perceive from this histogram

representation the existence of a change in data distribution. In contrast, the representation provided by fading histograms make the change occurred in data distribution clear. From this example, it is easy to observe that the representation from fading histograms is able to reproduce the new state of data earlier than the representation constructed over sliding windows. Therefore, fading histograms provide representations of data more up-to-date than any of the others.

The advantages of fading histograms over sliding histograms and over histograms constructed with all the data examples are straightforward: they are able to adapt better to a new data distribution and they are memoryless with respect to sliding histograms, which is an important property in data stream context (Gama et al., 2013).

3.4 Conclusions & Research Question

In this chapter the problem of data summarization from open-ended data streams has been discussed and the approaches proposed to remember important properties of data streams and to forget outdated data were presented.

This chapter is devoted to research question 1, which consisted of two parts:

1. In the context of massive data streams, which strategy should be used to remember the discarded data?

2. In the context of time-changing data streams, how can a compact representation of data forget outdated data in order to be able to keep up with the current state of evolving nature?

In this thesis the properties of data are remembered using online histograms under mean square error constraints. The advantage of this summarization structure is bi-fold: it constructs a compact representation of data and provides a visual interpretation of the underlying distribution.

Nowadays, it is not feasible to consider that data is collected in a static scenario. To deal with non-stationary data streams, besides remembering data that is being discarded after it is processed, the online histograms must also forget data that is not describing the current state of nature. To accomplish this target, online histograms are constructed over sliding windows or using fading factors. The former strategy enforces an abrupt forgetting while

the second approach, which has the advantage of being memoryless, lets outdated data slip out gradually.

The following chapter advances the proposed approach of monitoring and comparing data distributions in dynamic environments. Data distribution is approximated using the histograms mentioned in Section 3.3.

- 4 -

Monitoring Data over Sliding Windows

"Todo o mundo é composto de mudança,
Tomando sempre novas qualidades."
Luís Vaz de Camões (1524 - 1580)

Consider a data stream mining algorithm, whose main purpose is to create a model describing certain properties of a data stream. While the underlying distribution that generates data remains stable the model will be accurate. However, at the occurrence of a change, old observations become irrelevant to the current state of nature, and their contribution to the model becomes inappropriate. Therefore, the description provided by the model will no longer be accurate. Old information must be forgotten and a new model representing the current state of nature is needed, which stresses out the demand of detecting such changes.

To cope with this problem, a change detection model based on monitoring data distributions is proposed. Research question 2 addresses the characteristics of the change detection model and therefore is dealt with in this chapter.

The chapter is organized as follows. It starts with the objectives and methodology of research question 2. Then proposes an approach to detect changes through the monitoring of data distributions over two time windows: the Cumulative Windows Model (CWM) (Sebastião and Gama, 2007). Section 4.3 designs experiments to evaluate the performance of CWM on detecting both distribution and concept changes, using artificial data, real data and a public data set. This chapter ends with conclusions on the change detection model and answers to the respective research question.

4.1 Research Question

One of the main challenges in data mining occurs when the underlying distribution that generates the data streams changes over time. In dynamic environments it is of utmost importance to perform change detection tests to instigate if the underlying distribution generating data is stationary. This chapter presents a model to detect changes by monitoring data in two time windows. Considering the scope of this approach, research question 2 is raised:

1. In the development of a model to detect changes through the comparison of distributions over two time windows, which is the appropriate step to perform comparisons?
2. When evaluating the distance between distributions, how do the forgetting rates of fading histograms affect the detection delay time?
3. What is the robustness against noise of the proposed change detection model?
4. What is the effect of the extension of a stationary phase in the performance of the proposed change detection model?

Within this research question, the objectives are to:

1. Propose a windows-based model for change detection, where the step to perform data distributions comparison is automatically defined according to the data stationarity.
2. Evaluate the overall performance of the CWM in detecting distribution changes with different magnitudes and rates, namely to:
    - Evaluate the advantage of using an adaptive evaluation step instead of a fixed one.
    - Evaluate the benefit, in detection delay time, of using fading histograms when comparing data distributions to detect changes.
    - Evaluate robustness to detect changes against different amounts of noise.
    - Evaluate the stability in static phases with different lengths and how it affects the ability to detect changes.

The above objectives are accomplished with the following methodology:

1. The step to perform comparisons is defined according to the distance between data distributions: the step is increased if the distance is small (which suggests that data is generated according to the same distribution, hence it is a stationary phase) and is decreased if the distance is high (which means that data from both windows is further apart).

2. Design experimental evaluation in order to assess the overall performance of the change detection approach under different evolving scenarios.

4.2 Cumulative Windows Model for Change Detection

The Cumulative Windows Model (CWM) is proposed for detecting changes when monitoring data over sliding windows, addressing both distribution and concept changes. Within this approach, the reference window (RW) has a fixed length and reflects the data distribution observed in the past. The current window (CW) is cumulative and it is updated sliding forward and receiving the most recent data. In the CWM, the data distribution is computed by the histograms presented in Section 3.3 of the previous chapter.

In change detection problems, it is mandatory to detect changes as soon as possible, minimizing the delay time in detection. Along with this, the false and the missed detections must be minimal. Therefore, the main challenge when proposing an approach for change detection is reaching a trade-off between the robustness to false detections (and noise) and sensitivity to true changes. The CWM should detect true changes with high probability (with few spurious detections - false positives) and as soon as they occur. Along with this, CWM should present a minimal number of missed detections (false negatives). Moreover, an effective change detection method must be able to forget outdated data, be a single pass and allow constant updates in time and memory. The proposed CWM was designed to fulfill these requirements.

It must be pointed out that the proposed CWM is a non-parametric approach, which means that it makes no assumptions on the form of the distribution. This is a property of major interest, since real data streams rarely follow known and well-behavior distributions.

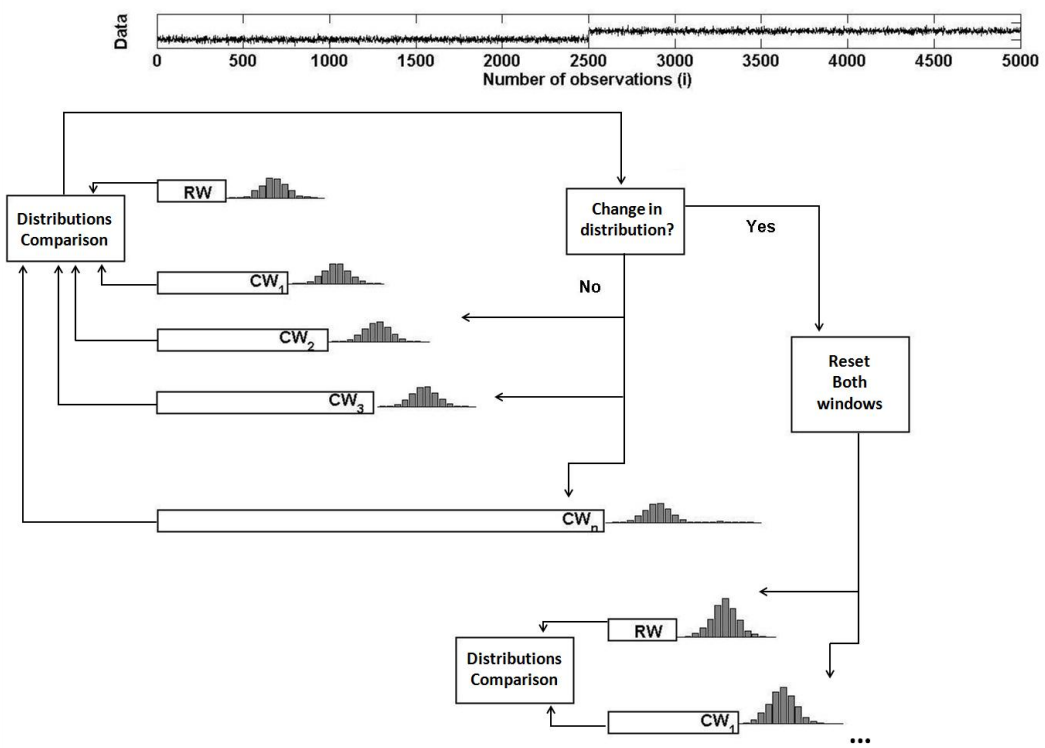

Figure 4.1: Workflow of the Cumulative Windows Model (CWM) for change detection.

Figure 4.1 shows the workflow of the CWM for change detection. The histograms representations were constructed from the observed data, with different number of observations. At every evaluation step, the data distribution in the Current Window (CW) is compared with the distribution of the data in the Reference Window (RW). If a change in the distribution of the data in the CW with respect to the distribution of the data in the RW is not detected, the CW is updated with more data observations. Otherwise, if a change is detected, the data in both windows is cleaned, new data is used to fulfill both windows and a new comparison starts.

4.2.1 Distance Between Distributions

From information theory (Berthold and Hand, 1999), the Relative Entropy is one of the most general ways of representing the distance between two distributions (Dasu et al., 2006). Contrary to the Mutual Information this measure assesses the dissimilarity between two variables. Also known as the Kullback-Leibler divergence, it measures the distance between two probability distributions and therefore is suitable for use in comparison purposes.

Assuming that the data in the reference window has distribution P_{RW} and that data in the

current window has distribution P_{CW}, the Kullback-Leibler Divergence (KLD) is used as a measure to detect whenever a change in the distribution has occurred.

Considering two discrete distributions with empirical probabilities $P_{RW}(i)$ and $P_{CW}(i)$, the relative entropy of P_{RW} with respect to P_{CW} is defined by:

$$KLD(P_{RW}||P_{CW}) = \sum_i P_{RW}(i) log \frac{P_{RW}(i)}{P_{CW}(i)}.$$

The Kullback-Leibler divergence is not a real metric since is asymmetric:

$$KLD(P_{RW}||P_{CW}) \neq KLD(P_{CW}||P_{RW}).$$

Nevertheless, it satisfies many important mathematical properties: is a nonnegative measure, it is a convex function of $P_{RW}(i)$ and equals zero only if $P_{RW}(i) = P_{CW}(i)$.

Consider a reference window with empirical probabilities $P_{RW}(i)$, and a current sliding window with probabilities $P_{CW}(i)$. Taking into account the asymmetric property of the Kullback-Leibler divergence and that the minimal value of the difference between $KLD(P_{RW}||P_{CW})$ and $KLD(P_{CW}||P_{RW})$, which is zero, is achieved when P=Q: smaller values of this difference correspond to smaller dispersion between both data distributions, meaning that the data is similar; and higher values of this difference suggest that distributions are further apart.

4.2.2 Decision Rule

Considering a change detection approach based on monitoring data distribution over two windows, the problem of detecting changes in a data stream is accomplished evaluating whether the distance between data distribution on both windows is different enough.

In the proposed Cumulative Windows Model for change detection, the decision rule used to assess changes in data distribution is based on the asymmetry of the Kullback-Leibler divergence. It is defined that a change has occurred in the data distribution of the current window relatively to the data distribution of the reference window, if the absolute difference of the KLD between the distributions of the reference and the current windows and the KLD between the distributions of the current and the reference windows is greater than a given threshold δ:

$$|KLD(P_{RW}||P_{CW}) - KLD(P_{CW}||P_{RW})| > \delta.$$

The threshold δ represents a trade-off between sensitivity to true changes and robustness to false detections of the proposed model for change detection. A high value of δ entails fewer false alarms, but might miss or delay the detection of true changes. A small value of δ increases the rate of detecting small changes and ensures lower detection delay times, but also increases the rate of false alarms.

If no change occurs, the reference distribution is maintained and more data points are considered in the current window, and a new comparison is made. If any anomalies and/or deviations from what is expected are detected, an alert alarm can be triggered and a change is assigned. In that case, both windows are cleaned, initializing a new process of change detection.

4.2.3 Evaluation Step for Data Distributions Comparison

In the proposed model for change detection, the evaluation step is the increment of the cumulative current window. In a stationary phase, does it make any sense to compare data distributions at each time a new data point is available? Regarding efficiency, computational costs and resources availability, the answer is no. On the other hand, in a non-stationary phase, a large evaluation step could compromise the change detection, by delaying it.

This raises the first part of research question 2: *In the development of a model to detect changes through the comparison of distributions over two time windows, which is the appropriate step to perform comparisons?*

When comparing data distributions over sliding windows, at each evaluation step the change detection method is induced by the examples that are included in the sliding window. Here, the key difficulty is how to select the appropriate evaluation step. A small evaluation step may ensure fast adaptability in phases where the data distribution changes. However, a small evaluation step implies that more data comparisons are made. Therefore, it tends to be computationally costly, which can affect the overall performance of the change detection method. On the other hand, with a large evaluation step, the number of data distribution comparisons decreases, increasing the performance of the change detection method in stable phases but not allowing quick reactions when a change in the distribution occurs.

Therefore, the *a priori* definition of the evaluation step to perform data distribution comparisons is a compromise between computational costs and detection delay time. In the proposed approach, the evaluation step, instead of being fixed and selected by the user, is automatically defined according to the data stationarity and to the distance

between data distributions. Starting with an evaluation step of $IniEvalStep$, the step length is increased if the distance between distributions is small (which suggests that data is generated according to the same distribution, hence it is a stationary phase) and is decreased if the distance between distributions is high (which means that data from both windows is further apart), according to the following relation:

$$EvalStep = max(1, round(IniEvalStep * (1 - \tfrac{1}{\delta}) * |KLD(P_{RW}||P_{CW}) - KLD(P_{CW}||P_{RW})|)),$$

where KLD denotes the Kullback-Leibler divergence between data distributions and δ is the change detection threshold.

Figure 4.2 illustrates the dependency of the evaluation step on the distance between data distributions of an artificial data set, for a change detection threshold $\delta = 0.1$.

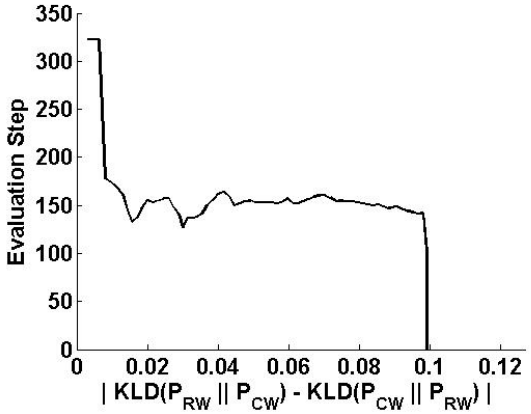

Figure 4.2: Representation of the evaluation step for data distributions comparison with respect to the absolute difference between $KLD(P_{RW}||P_{CW})$ and $KLD(P_{CW}||P_{RW})$ (for a change detection threshold of $\delta = 0.1$).

4.2.4 Pseudocode

The presented Cumulative Windows Model (CWM) was designed to detect changes online in the distribution of streams of data. The data distribution is computed by the histograms presented in Section 3.3 of the previous chapter. In order to detect changes, the data distributions in two time windows are compared using the Kullback-Leibler divergence. Algorithm 2 presents the pseudocode for this CWM.

Algorithm 2 CWM

Input: Data set: $x_1, x_2, \ldots$
Number of buckets in the histogram $nBuckets$
Length of the Reference window: L_{RW}
Initial evaluation step $IniEvalStep$
Change detection threshold δ
Output: Time of the detected changes: t^*
$t_i \leftarrow 0$
Step 1:
Initialize the histogram in the reference window (P_{RW}) as empty
Initialize the histogram in the current window (P_{CW})as empty
Define the first evaluation point: $EvalPoint = L_{RW} + IniEvalStep$
while not at the end of the stream **do**
if $t \leq t_i + L_{RW}$ **then**
Compute the histogram in the RW: P_{RW}
Compute the histogram counts for the CW
else if $t = EvalPoint$ **then**
Compute the histogram in the CW: P_{CW}
Compute the next evaluation step:
$EvalStep = max(1, round(IniEvalStep(1 - \frac{1}{\delta}) * |KLD(P_{RW}||P_{CW}) - KLD(P_{CW}||P_{RW})|))$
Compute the next evaluation point: $EvalPoint = EvalPoint + EvalStep$
if $|KLD(P_{RW}||P_{CW}) - KLD(P_{CW}||P_{RW})| > \delta$ **then**
$t_i \leftarrow i$
report a change at time t: $t^* = t$
Go to **Step 1**
end if
else
Compute the histogram counts for the CW
end if
end while

4.3 Experimental Evaluation

In this section, the proposed CWM is evaluated under different evolving scenarios, using artificial and real data, and assessing the ability to detect distribution changes and concept changes. Moreover, regarding the detection of concept changes, a comparison of the CWM with 3 known algorithms taken from literature is presented. The ability of CWM to detect distribution and concept changes was evaluated through the criteria and metrics presented in Section 2.8. The artificial data as obtained in MATLAB (MATLAB®& Simulink®, 2007). All the experiments were implemented in MATLAB as well as the graphics produced.

4.3.1 Distribution Changes

This section presents the performance of the CWM in detecting distribution changes. To detect distribution changes, the model is evaluated using artificial data, presenting distribution changes with different magnitudes and rates, and using real world data from an industrial process and a biomedical problem.

4.3.1.1 Controlled Experiments with Artificial Data

The objectives of these controlled experiments with artificial data is to provide answers to the research question 2. The data sets and the experimental designs were outlined in order to evaluate the overall performance of the CWM in detecting distribution changes in different evolving scenarios, namely to:

1. Evaluate the advantage of using an adaptive evaluation step instead of a fixed one.
2. Evaluate the benefit, in detection delay time, of using fading histograms when comparing data distributions to detect changes.
3. Evaluate robustness to detect changes against different amounts of noise.
4. Evaluate the stability in static phases with different lengths and how it affects the ability to detect changes.

The data sets were generated according to a normal distribution with certain parameters. Both the mean and the standard deviation parameters were varied, generating 2 different

problems according to the source of the change. Each data stream consists of 2 parts, where the size of the second is N.

Two data sets were generated. In the first data set, the length of the first part of data streams was set to N. In the second data set, the length of the first part was set to $1N$, $2N$, $3N$ $4N$ and $5N$, in order to simulate different lengths of static phases.

The first data set was used to carry out the first, the second and the third experimental designs and the second data set was used to perform the fourth experimental design, evaluating the effect of different extensions of the stationary phase on the performance of the CWM in detecting changes.

Within each part of the data streams the parameters stay the same, which means that only 1 change happens between both parts and different changes were simulated by varying among 3 levels of magnitude (or severity) and 3 rates (or speed) of change, obtaining a total of 9 types of changes for each changing source (therefore, a total of 18 data streams with different kind of changes). Although there is no golden rule to classify the magnitude levels, they were defined in relation to one another, as abrupt, medium and smooth according to the variation of the distribution parameter, as shown in Table 4.1. For each type of changes, 30 different data streams were generated with different seeds.

Table 4.1: Magnitude levels of the designed data sets.

Parameter changed	Value of the fixed parameter	Parameter variation (before $\rightarrow$ after change)	Magnitude of change
μ	$\sigma = 1$	$\mu = 0 \rightarrow \mu = 5$	Abrupt
		$\mu = 0 \rightarrow \mu = 3$	Medium
		$\mu = 0 \rightarrow \mu = 2$	Smooth
σ	$\mu = 0$	$\sigma = 1 \rightarrow \sigma = 5$	Abrupt
		$\sigma = 1 \rightarrow \sigma = 3$	Medium
		$\sigma = 1 \rightarrow \sigma = 2$	Smooth

The rates of change were defined assuming that the examples from the first part of data streams are from the old distribution and the $N - ChangeLength$ last examples are from the new distribution, where $ChangeLength$ is the number of examples required before the change is complete. During the length of the change, the examples from the new distribution are generated with probability $p_{new}(t) = \frac{t-N}{ChangeLength}$ and the examples from the old distribution are generated with probability $p_{old}(t) = 1 - p_{new}(t)$, where $N < t < N + ChangeLength$. As for the magnitude levels, the rates of change were

defined in relation to one another, as sudden, medium and low, for a $ChangeLength$ of 1, $0.25 * N$ and $0.5 * N$, respectively. The value of N was set to 1000.

Therefore, the first data set is composed by a total of 540 data streams with 2000 examples each, and the second data set consists of a total of 2700 data streams with five different lengths, 540 data streams of each length.

Setting the parameters of the CWM and of the online histograms

The CWM and the online histograms require the setting of the following parameters:

- L_{RW} - length of the reference window (CWM);
- $IniEvalStep$ - initial evaluation step (CWM);
- δ - change detection threshold (CWM);
- ϵ - admissible mean square error of the histogram;

It was established that 5% was an admissible mean square error of the histogram. To instigate the values for the remaining parameters, an experiment was performed on a training data set with the same characteristics as the first data set, varying the L_{RW} within $1k, 2k, \ldots 10k$ (where k is the number of buckets in the online histograms) and δ within $0.01, 0.05, 0.1, 0.2$. However, in this training data set, only 10 data streams were generated with different seeds for each type of drift, obtaining a total of 118 data streams with length $2N$ ($N = 1000$). In this experiment, the CWM was performed with a unitary evaluation step and the summary results were analyzed. Table 4.2 presents the precision, recall and F_1 score for a reference window of length $50k$ and $10k$ and for a change detection threshold of 0.05 and 0.1, for a total of 180 true changes. Although compromising the delay time in change detection, the best F_1 score is obtained for a reference window of $10k$ examples and a change detection threshold of 0.05.

Table 4.2: Precision, Recall and F_1 score, obtained when performing the CWM, with a reference window of length $5k$ and $10k$ and a change detection threshold of 0.05 and 0.1.

		L_{RW}	
		$5k$	$10k$
δ	0.05	$Precision = 0.87$ $Recall = 1$ $F_1 = 0.93$	$Precision = 0.98$ $Recall = 0.97$ $F_1 = 0.98$
	0.1	$Precision = 0.97$ $Recall = 0.96$ $F_1 = 0.97$	$Precision = 1$ $Recall = 0.88$ $F_1 = 0.93$

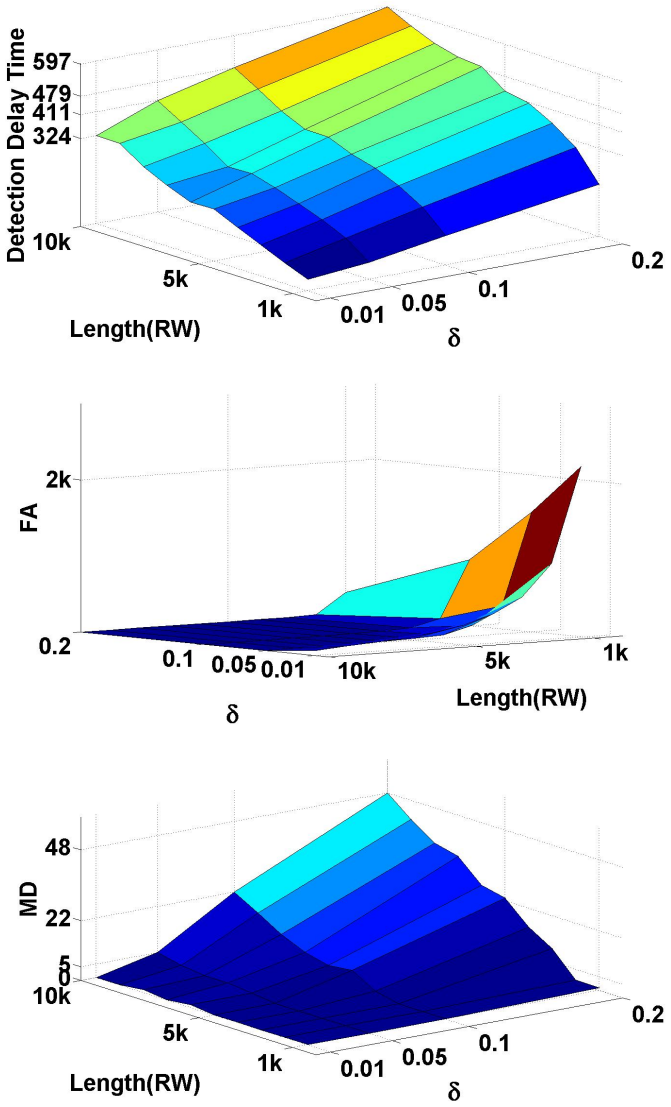

Figure 4.3: Detection delay time, total number of false alarms (FA) and missed detections (MD), depending on the L_{RW} and δ.

Figure 4.3 shows the detection delay time (average of 10 runs) and the total number of false alarms (FA) and missed detections (MD), for a total of 180 true changes, depending on the length of reference window of length (L_{RW}) and on the change detection threshold (δ). It can be observed, that an increase in the detection delay time, controlled by the value of δ, is followed by a decrease in the number of false alarms (and an increase in the number of missed detections). However, for $L_{RW} = 10k$ and $\delta = 0.05$, the false alarm rate ($3/180$) and the miss detection rate ($5/180$) are admissible.

Thereafter, the settings for the parameters of the CWM were the following:

- The length of the reference window was set to $10k$ ($L_{RW} = 10k$);
- The initial evaluation interval was set to k ($IniEvalStep = k$);
- The threshold for detecting changes was set to 5% ($\delta = 0.05$);

The parameter k is the number of buckets in the online histograms, and was computed establishing the admissible mean square error of the histogram as 5% ($\epsilon = 0.05$).

Evaluate the advantage of using an adaptive evaluation step instead of a fixed one

This experiment was designed to study the advantage of performing the CWM with an adaptive evaluation step (ACWM) against a fixed evaluation step (FCWM). The first data set was used in this experimental design. Figure 4.4 shows the advantage, in detection delay time, of an adaptive evaluation step over the fixed one (average results for 30 runs on data generated with different seeds). Except for the distribution change in the mean parameter, with smooth magnitude and sudden rate, the detection delay time is shorter when performing the ACWM. This decrease in the detection delay time, when performing ACWM, is obtained without compromising the false alarm and miss detection rates (except for one case: change with smooth magnitude and abrupt rate, in the mean parameter, see Table 4.3).

Using the results of the 30 replicas of the data, a paired, two-sided Wilcoxon signed rank test was performed to assess the statistical significance of the comparison results. It was tested the null hypothesis that the difference between the detection delay times of the ACWM and the FCWM comes from a continuous, symmetric distribution with zero median, against the alternative that the distribution does not have zero median. For all types of changes in both mean and standard deviation parameters, the null hypothesis of zero median in the differences was rejected, at a significance level of 1%. Therefore, considering the very low p-values obtained, there is strong statistical evidence that the detection delay time of ACWM is smaller than of FCWM.

In Table 4.3, besides the detection delay time using the CWM with an adaptive and a fixed evaluation steps, the total number of missed detections and the total number of false alarms are also presented. The results report the average and standard deviation of 30 runs.

As expected, greater distribution changes (abrupt magnitudes and sudden rates) are easier to detect by the CWM, either using an adaptive or a fixed evaluation step. On the other

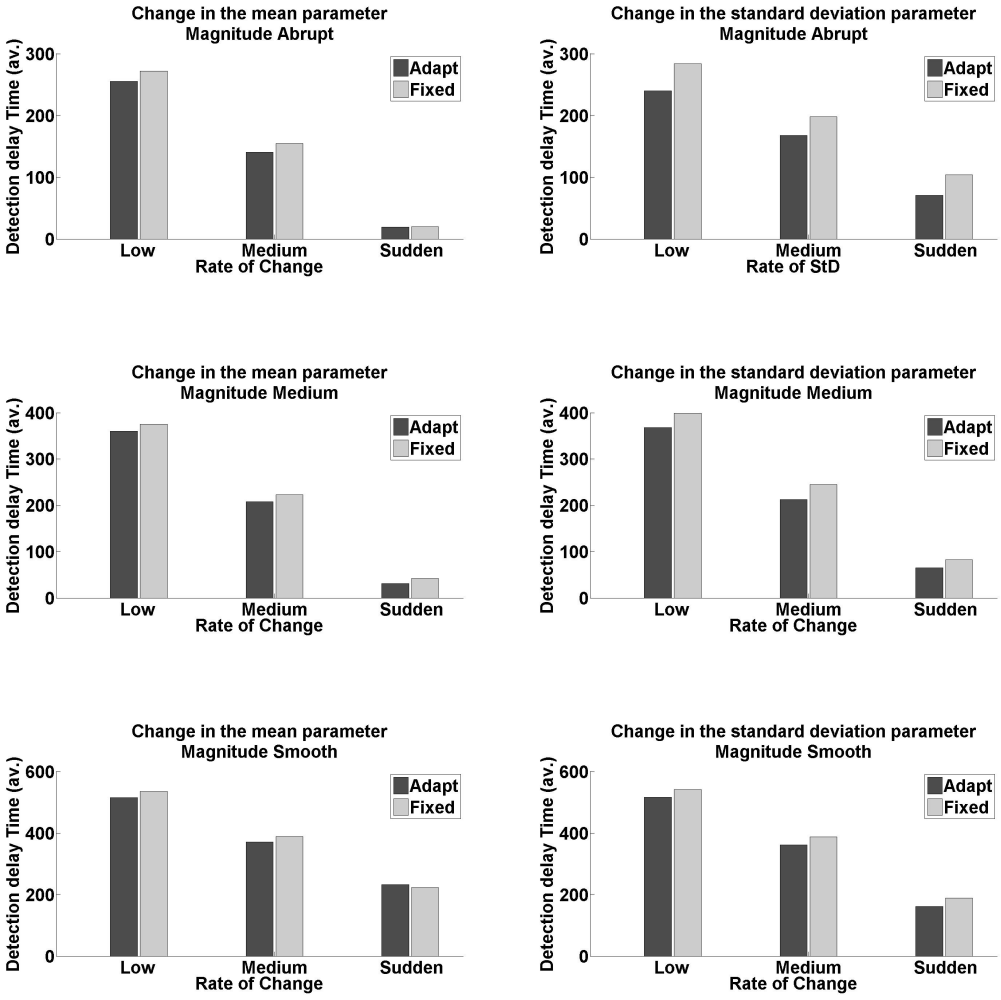

Figure 4.4: Detection delay time (average of 30 runs) using the CWM with an adaptive (ACWM) and a fixed (FCWM) evaluation steps.

hand, for smaller distribution changes (smooth magnitudes and low rates) the detection delay time increases. The decrease in the detection delay time in this experiment sustains the use of an adaptive evaluation step, when performing the CWM. Although the decrease in detection delay time is small, these results must be taken into account that the length of the data was also small. With data with higher length, the decrease of detection delay time will be reinforced. Moreover, for both strategies, the execution time of performing the CWM is comparable.

Evaluate the benefit, in detection delay time, of using fading histograms when comparing data distributions to detect changes.

Table 4.3: Detection delay time (DDT) using the ACWM and the FCWM. The results report the average and standard deviation of 30 runs. In parenthesis is the number of runs, if any, where the CWM misses detection or signals a false alarm: they are in the form (Miss; False Alarm).

Parameter changed	Mag.	Rate	Adaptive Step DDT ($\mu \pm \sigma$)	Fixed Step DDT ($\mu \pm \sigma$)
Mean	Abrupt	Low	260 ± 57	275 ± 53 (0;0)
		Medium	153 ± 24 (1;1)	178 ± 59 (0;1)
		Sudden	19 ± 4 (0;1)	24 ± 0 (0;1)
	Medium	Low	410 ± 131 (0;1)	(0;1)424 ± 138
		Medium	242 ± 125 (0;1)	259 ± 122 (0;1)
		Sudden	36 ± 22 (0;1)	52 ± 26 (0;1)
	Smooth	Low	516 ± 171 (7;2)	535 ± 177 (7;2)
		Medium	371 ± 233 (5;0)	389 ± 232 (5;0)
		Sudden	233 ± 229 (1;0)	223 ± 193 (3;0)
STD	Abrupt	Low	240 ± 34	284 ± 42
		Medium	168 ± 16	198 ± 30
		Sudden	71 ± 10	104 ± 0
	Medium	Low	368 ± 87	399 ± 93
		Medium	213 ± 28	245 ± 32
		Sudden	65 ± 15	83 ± 27
	Smooth	Low	517 ± 158	542 ± 154
		Medium	362 ± 127	387 ± 129
		Sudden	162 ± 60 (1;0)	189 ± 63 (1;0)

As stated before, fading histograms attribute more weight to recent data. In an evolving scenario, this could be a huge advantage since it enhances small changes. Therefore, when comparing data distributions to detect changes, the detection of such changes will be easier. This experimental design intends to evaluate the benefit of using fading histograms as a synopsis structure to represent the data distributions that will be compared, for detecting changes, within the ACWM (which will be referred to as ACWM-fh). Thus, data distributions, within the reference and the current windows, were computed using fading histograms with different values of fading factors: 1 (no forgetting at all), 0.9994, 0.9993, 0.999, 0.9985 and 0.997.

As with evaluating the statistical significance of the results of ACWM and FCWM, a paired, two-sided Wilcoxon signed rank test was performed to assess the statistical significance of differences between ACWM and ACWM-fh, using the results of the 30 replicas of the data. With the exception of the change in the mean parameter with abrupt magnitude and sudden rate (for the fading factors tested except 0.997), for all the other types of changes in both mean and standard deviation parameters, the null hypothesis of zero median in the differences between detection delay times was rejected, at a significance level of 1%. Therefore, considering the very low p-values obtained, there is strong statistical evidence

that the detection delay time of ACWM-fh is smaller than of ACWM.

Table 4.4 presents a summary of the detection delay time (average and standard deviation from 30 runs on data generated with different seeds) using the ACWM-fh for comparing the data distributions in the first data set. The total number of missed detections and the total number of false alarms are also presented. This experiment out underlines the advantage of using fading histograms to compute the data distributions: the detection delay time decreases by decreasing the fading factor and without compromising the number of missed detections and false alarms (except when using a fading factor of 0.997). The increase of false alarms when using a fading factor of 0.997 suggests that fading histograms computed with this value are over reactive, therefore fading factors of values equal or smaller than 0.997 are not suitable for use in this data set.

Table 4.4: Detection delay time (average and standard deviation), using the ACWM and computing data distributions with fading histograms (with different fading factors). The results report the average and standard deviation of 30 runs. In parenthesis is the number of runs, if any, where the ACWM-fh misses detection or signals a false alarm: they are in the form (Miss; False Alarm).

Parameter changed	Mag.	Rate	Fading Factor 1	0.9994	0.9993	0.999	0.9985	0.997
Mean	Abrupt	Low	260 ± 57	246 ± 64	246 ± 64	241 ± 68	233 ± 77	226 ± 70 (0;5)
		Medium	153 ± 24 (1;1)	145 ± 27 (0;1)	150 ± 33 (0;2)	140 ± 31 (0;2)	140 ± 32 (0;2)	125 ± 36 (0;2)
		Sudden	19 ± 4 (0;1)	19 ± 5 (0;1)	19 ± 5 (0;1)	18 ± 6 (0;1)	16 ± 6 (0;1)	13 ± 6 (0;1)
	Medium	Low	410 ± 131 (0;1)	387 ± 142 (0;1)	385 ± 144 (0;1)	381 ± 151 (0;1)	365 ± 148 (0;2)	311 ± 96 (0;5)
		Medium	242 ± 125 (0;1)	215 ± 69 (0;2)	213 ± 69 (0;2)	211 ± 58 (0;3)	205 ± 58 (0;3)	186 ± 54 (0;5)
		Sudden	36 ± 22 (0;1)	27 ± 16 (0;1)	25 ± 15 (0;1)	23 ± 11 (0;1)	22 ± 9 (0;1)	36 ± 110 (0;2)
	Smooth	Low	516 ± 171 (7;2)	510 ± 202 (4;2)	496 ± 204 (4;2)	496 ± 197 (3;2)	448 ± 173 (2;2)	369 ± 150 (0;9)
		Medium	371 ± 233 (5;0)	338 ± 208 (3;0)	327 ± 193 (3;0)	324 ± 209 (1;0)	289 ± 168	250 ± 151 (0;4)
		Sudden	233 ± 229 (1;0)	165 ± 170 (1;0)	159 ± 168 (1;0)	139 ± 159 (1;0)	138 ± 180	66 ± 76 (0;1)
STD	Abrupt	Low	240 ± 34	219 ± 36	216 ± 38	208 ± 41	204 ± 45	186 ± 52
		Medium	168 ± 16	157 ± 18	155 ± 19	151 ± 22	143 ± 25	138 ± 40
		Sudden	71 ± 10	60 ± 13	58 ± 14	52 ± 16	42 ± 19	23 ± 18
	Medium	Low	368 ± 87	327 ± 76	322 ± 76	310 ± 76	294 ± 79	249 ± 92
		Medium	213 ± 28	185 ± 30	180 ± 30	170 ± 32	159 ± 35	140 ± 40
		Sudden	65 ± 15	53 ± 13	52 ± 14	49 ± 13	43 ± 13	39 ± 14
	Smooth	Low	517 ± 158	445 ± 140	435 ± 137	411 ± 136	380 ± 145	316 ± 113 (0;2)
		Medium	362 ± 127	305 ± 101	295 ± 92	278 ± 88	260 ± 85	204 ± 52
		Sudden	162 ± 60 (1;0)	116 ± 41 (1;0)	122 ± 74	109 ± 74	87 ± 46	62 ± 40

The detection delay time (average of 30 runs on data generated with different seeds) of this experimental design is shown in Figure 4.5. It can be observed that the advantage of using fading histograms is strengthened when detecting small changes, which is explained by the greater importance attributed to recent examples that enhances a change and eases its detection by the ACWM.

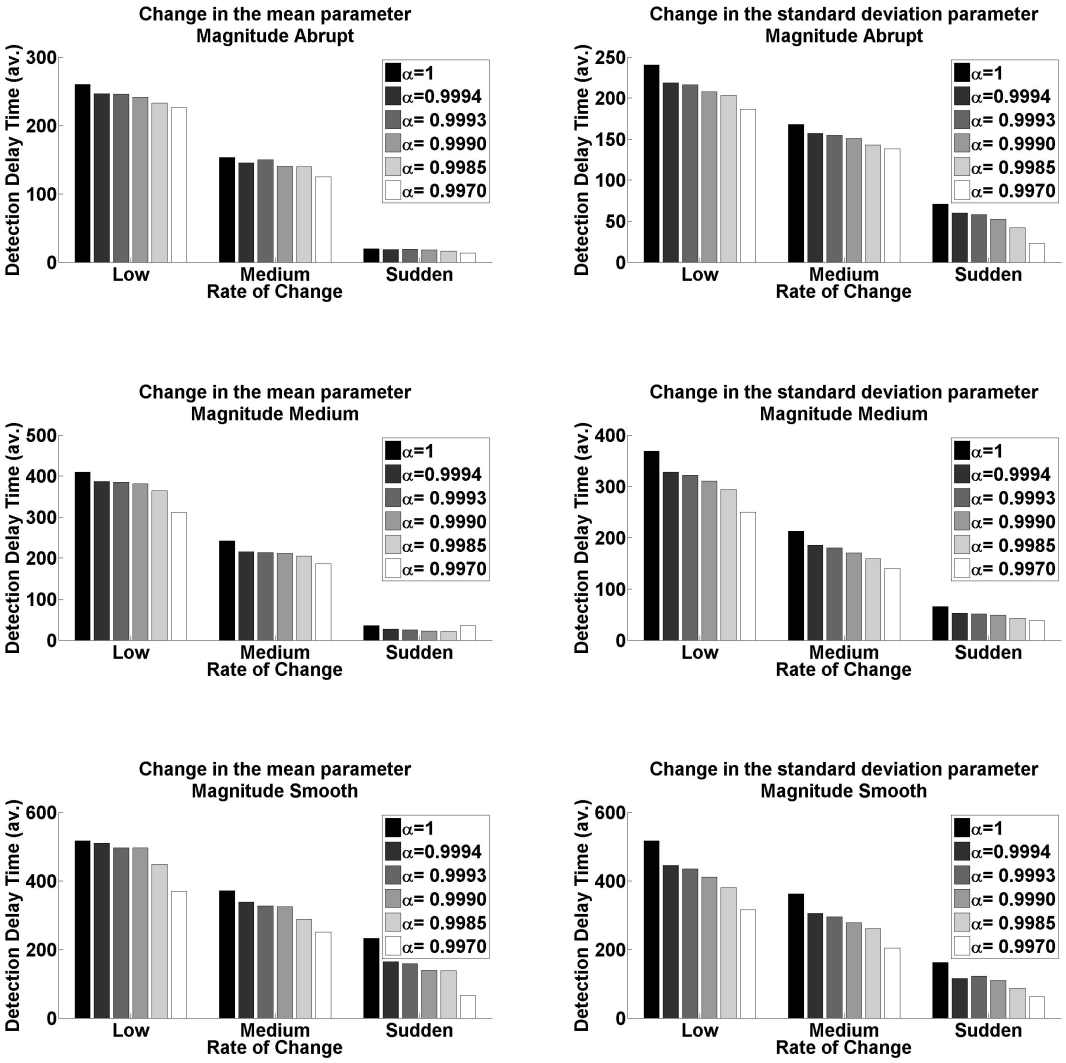

Figure 4.5: Detection delay time (average of 30 runs) of the ACWM-fh.

Evaluate the robustness to detect changes against different amounts of noise.

Within this experimental design, the robustness of the ACWM against noise was evaluated. Noisy data was generated by adding different percentages of Gaussian noise with zero mean and unit variance to the first data set. Figure 4.6 shows the obtained results by varying the amount of noise from 10% to 50%.

The detection delay time (average of 30 runs on data generated with different seeds) of this experimental design is shown in Figure 4.6. The ACWM presents a similar performance along the different amounts of noise, with the exception of a change in the standard deviation parameter with abrupt magnitude and medium and sudden rates (for a level of

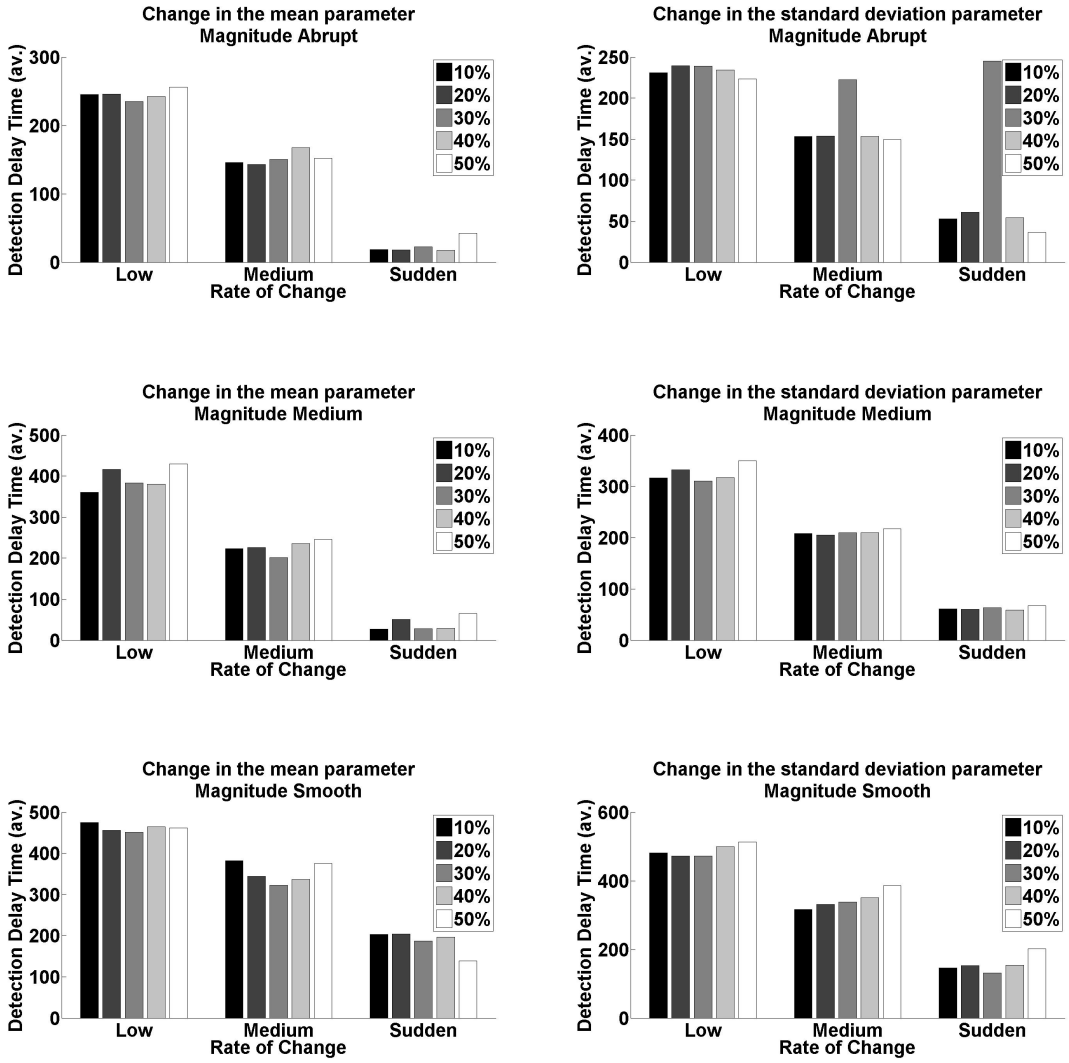

Figure 4.6: Detection delay time (average of 30 runs) of the ACWM with different amounts of noise.

noise of 30%). In these cases, the average detection delay time increases when compared with other amounts of noise. This experiment sustains the argument that the ACWM is robustness against noise while effectively detects distribution changes in the data.

Table 4.5 presents a summary of the detection delay time (average and standard deviation from 30 runs on data generated with different seeds) using the ACWM for comparing the data distributions in the first data set. The total number of missed detections and the total number of false alarms are also presented. Regarding the total number of missed detections and false alarms, with an amount of 50% of noise a slight increase is noticeable for both, mainly for changes in the mean parameter.

Table 4.5: Detection delay time (average and standard deviation), using the ACWM with different amounts of noise. The results report the average and standard deviation of 30 runs. In parenthesis is the number of runs, if any, where the ACWM misses detection or signals a false alarm: they are in the form (Miss; False Alarm).

Parameter changed	Mag.	Rate	Noise Scale 10%	20%	30%	40%	50%
Mean	Abrupt	Low	245 ± 62	246 ± 67	235 ± 44	242 ± 82	256 ± 75 (0;3)
		Medium	146 ± 25 (0;2)	143 ± 29 (0;1)	150 ± 30 (0;1)	167 ± 148 (0;2)	152 ± 36 (0;2)
		Sudden	19 ± 5	18 ± 4	22 ± 6	17 ± 4	42 ± 150 (0;1)
	Medium	Low	360 ± 152 (1;0)	416 ± 123 (1;2)	383 ± 90 (1;1)	380 ± 127 (1;1)	429 ± 144 (1;4)
		Medium	223 ± 77 (0;3)	225 ± 70 (0;1)	201 ± 50	235 ± 72 (0;2)	246 ± 112 (1;5)
		Sudden	27 ± 12 (0;1)	51 ± 138 (0;2)	28 ± 9 (0;1)	29 ± 13 (0;1)	65 ± 120 (0;1)
	Smooth	Low	475 ± 201 (4;1)	456 ± 165 (6;1)	451 ± 137 (5;0)	464 ± 138 (7;2)	461 ± 144 (6;5)
		Medium	382 ± 237 (6;1)	344 ± 206 (5;1)	322 ± 166 (5;1)	336 ± 237 (4;1)	375 ± 205 (5;1)
		Sudden	203 ± 226 (0;1)	204 ± 234 (3;1)	187 ± 235 (1;0)	197 ± 221 (2;1)	139 ± 203 (3;9)
STD	Abrupt	Low	231 ± 35	240 ± 38	239 ± 36	234 ± 38	223 ± 53
		Medium	153 ± 17	154 ± 15	222 ± 89 (1;0)	153 ± 17	150 ± 27
		Sudden	53 ± 14	61 ± 14	245 ± 163 (2;0)	54 ± 13	36 ± 7
	Medium	Low	316 ± 47	332 ± 49	310 ± 63	317 ± 64	349 ± 79
		Medium	208 ± 24	205 ± 31	210 ± 29	209 ± 33	217 ± 56
		Sudden	61 ± 16	60 ± 18	63 ± 14	59 ± 14	67 ± 17
	Smooth	Low	481 ± 117	472 ± 129	472 ± 141 (1;0)	499 ± 128 (1;0)	513 ± 181 (2;1)
		Medium	316 ± 142 (1;0)	331 ± 107	338 ± 104	350 ± 134	386 ± 210 (0;1)
		Sudden	146 ± 80	153 ± 69	132 ± 63	154 ± 77	202 ± 152 (1;0)

Evaluate the stability in static phases with different lengths and how it affects the ability to detect changes.

This experiment was carried out with the second data set. The performance of the ACWM was evaluated varying the length of stationary phases from $1N$ to $5N$ ($N = 1000$).

The detection delay time (average of 30 runs on data generated with different seeds) of this experimental design is shown in Figure 4.7.

Overall, it can be observed that the detection delay time for the ACWM increases within the increase of the stationary phase. This is even more evident in distribution changes with sudden rates. Indeed, the stability of the ACWM in stationary phases, compromises the ability to effectively detect changes. However, this can be overthrown by using fading histograms to compute the data distributions, as shown in Figure 4.8.

Actually, in stationary phases, the ability of the fading histograms to forget outdated data works in favor of the change detection model, by decreasing the detection delay time. However, a decrease in the value of the fading factor results in the increase of the number of false alarms. Table 4.6 presents the detection delay time (average and standard deviation of 30 runs on data generated with different seeds for the 9 types of changes for each source parameter) using the ACWM-fh in different stationary phases. The total number of missed detections and the total number of false alarms are also presented. From the results presented, it can be noted that a decrease in the detection delay time is achieved, establishing a commitment with respect to the number of false alarms and

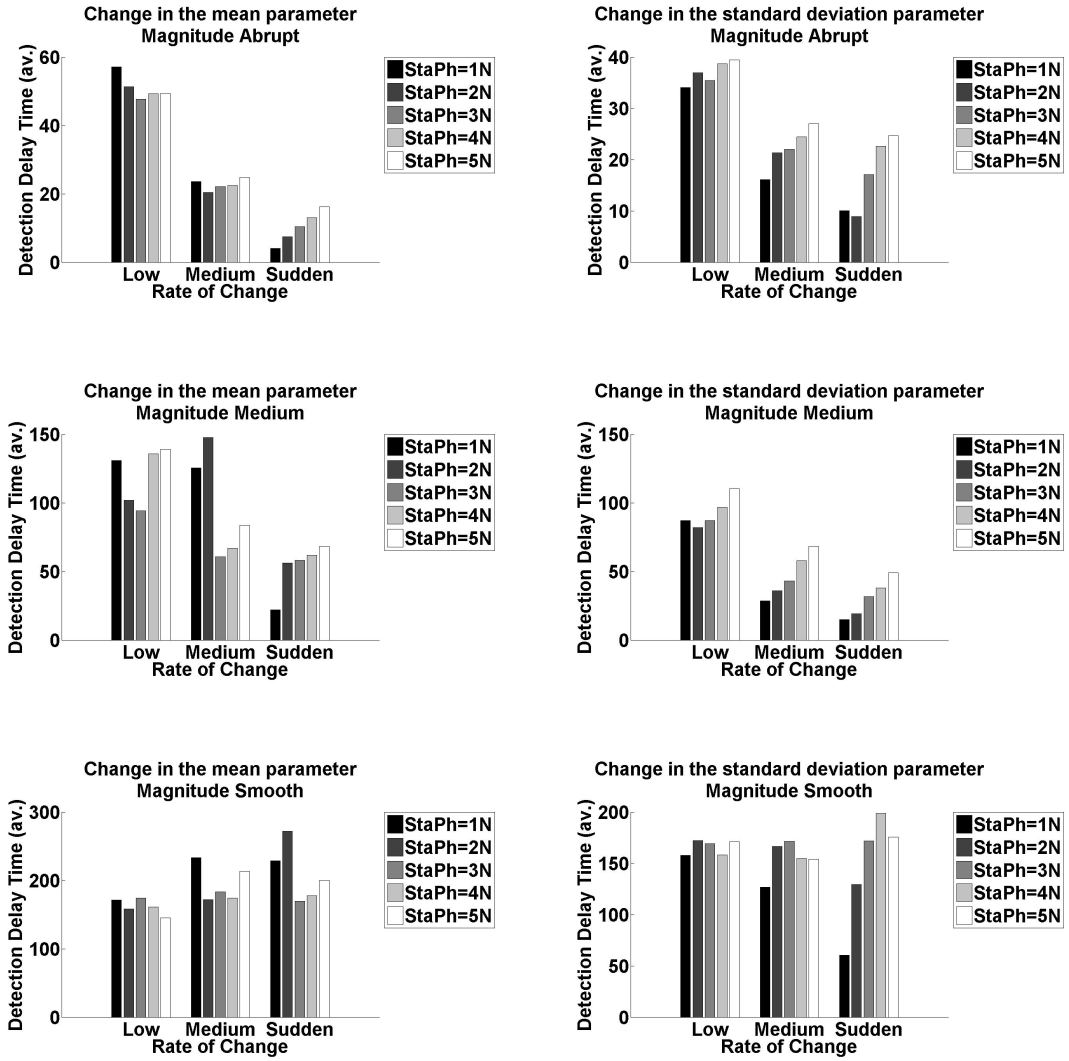

Figure 4.7: Detection delay time (average of 30 runs) of the ACWM with different lengths of stationary phases.

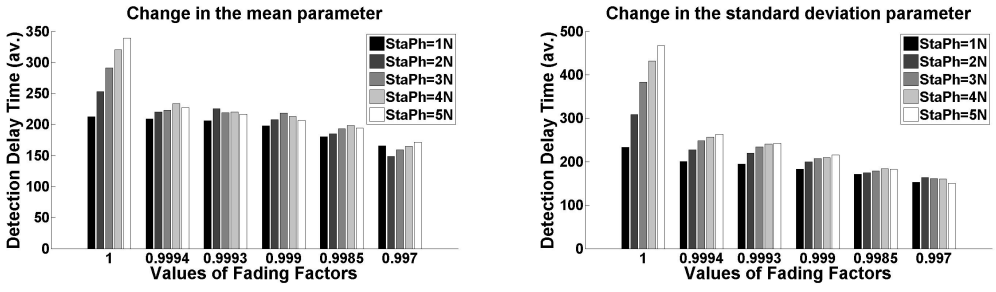

Figure 4.8: Detection delay time (average of 30 runs for the 9 types of changes) of the ACWM-fh with different lengths of stationary phases.

missed detections.

Table 4.6: Detection delay time (average and standard deviation) using the ACWM-fh in different stationary phases. The results report the average and standard deviation of 30 runs for the 9 types of changes for each source parameter. In parenthesis is the number of runs, if any, where the ACWM-fh misses detection or signals a false alarm: they are in the form (Miss; False Alarm).

Parameter changed	Fading Factor	Length of Stationary Phase				
		1k	2k	3k	4k	5k
Mean	1	249 ± 166 (14;7)	282 ± 172 (25;11)	300 ± 171 (31;7)	334 ± 183 (31;7)	365 ± 189 (35;7)
	0.9994	228 ± 163 (8;8)	248 ± 163 (12;14)	260 ± 159 (14;15)	258 ± 164 (12;17)	254 ± 159 (15;21)
	0.9993	224 ± 159 (8;9)	246 ± 170 (9;16)	247 ± 151 (14;18)	252 ± 162 (10;21)	250 ± 162 (11;24)
	0.999	219 ± 160 (5;10)	228 ± 161 (9;17)	229 ± 150 (10;23)	227 ± 153 (10;30)	228 ± 155 (11;34)
	0.9985	206 ± 146 (2;11)	221 ± 157 (4;20)	219 ± 162 (3;33)	228 ± 164 (1;47)	223 ± 163 (4;60)
	0.997	176 ± 125 (0;34)	188 ± 121 (0;79)	182 ± 127 (3;121)	177 ± 131 (0;146)	187 ± 122 (0;182)
Standard deviation	1	241 ± 150 (1;0)	317 ± 185 (4;0)	385 ± 203 (11;0)	441 ± 208 (23;0)	495 ± 220 (34;0)
	0.9994	207 ± 131 (1;0)	235 ± 141 (1;0)	251 ± 148 (1;0)	257 ± 151 (1;0)	263 ± 152 (1;0)
	0.9993	204 ± 127	227 ± 137 (1;0)	238 ± 142 (1;0)	243 ± 143 (1;0)	246 ± 144 (1;0)
	0.999	193 ± 122	208 ± 128 (1;0)	212 ± 130 (1;0)	213 ± 129 (1;1)	214 ± 131 (1;1)
	0.9985	179 ± 116	188 ± 118	186 ± 117 (1;1)	189 ± 124 (2;6)	187 ± 118 (0;9)
	0.997	151 ± 99 (0;2)	155 ± 96 (1;10)	160 ± 97 (1;21)	162 ± 105 (4;42)	162 ± 103 (3;60)

4.3.1.2 Industrial Data Set

This industrial data set was obtained within the scope of the work presented in Correa et al. (2009), with the objective of designing different machine learning classification methods for predicting surface roughness in high-speed machining. Data was obtained by performing tests in a Kondia HS1000 machining center equipped with a Siemens 840D open-architecture CNC. The blank material used for the tests was 170 $\times$ 100 $\times$ 25 aluminum samples with hardness ranging from 65 to 152 Brinell, which is a material commonly used in automotive and aeronautical applications. These tests were done with different cutting parameters, using sensors for registry vibration and cutting forces. A multi-component dynamometer with an upper plate was used to measure the in-process cutting forces and piezoelectric accelerometers in the X and Y axis for vibrations measures. Each record includes information on several variables used in a cutting process and the measurements for each test were saved individually.

For change detection purposes, the measurements of the cutting speed on X axes from 7 tests were joined sequentially in order to have only one data set with 6 changes with different magnitudes and sudden and low rates.

Figure 4.9 shows this data set and the change detection time of ACWM. It can be noted that ACWM takes too many observations to detect the fourth change. This can be explained by the fact that this change has smooth magnitude with respect to the remaining changes. The last change, which occurs at observation 420.000, is a change with a low rate: from

the beginning of the change until approximately observation 430.000 (which is the length of this change), the observations come from both the new and old tests. The ACWM is able to detect the beginning of the change as well as the end.

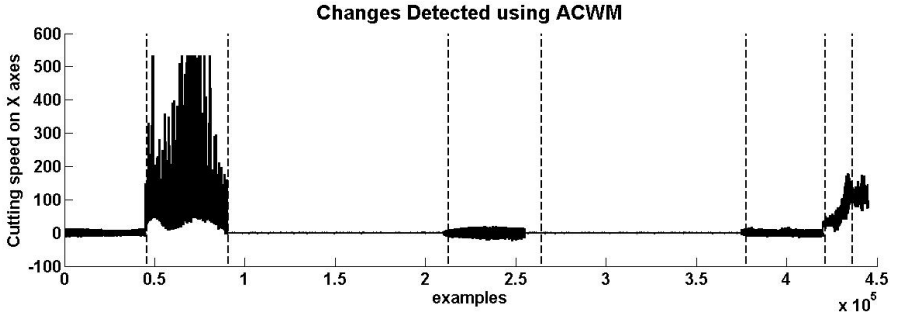

Figure 4.9: Detection delay time of the ACWM on the industrial data set.

The goal of this experiment is to evaluate the feasibility of the proposed ACWM with an industrial problem and comparing the advantage of using fading histograms. To this end, data distributions, within the reference and the current windows, were computed using fading histograms with different values of fading factors: 1 (no forgetting at all), 0.999998 and 0.999994.

The ACWM-fh is able to detect the 6 changes in the data with smaller detection delay time than when using online histograms. Moreover, with both approaches for data representations, ACWM did not miss any changes and although data has different kinds of changes, ACWM-fh presented a performance which was highly resilient to false alarms. The same is true of the last change when performing the ACWM-fh. Table 4.7 presents the detection delay time of the ACWM-fh when applied to this industrial data set. It can be observed that, as well as for the ACWM, ACWM-fh presents a high delay time when detecting the fourth change (with smooth magnitude)

4.3.1.3 Medical Data Set - CTGs

The CWM was evaluated on five Fetal Cardiotocographic (CTG) problems, collected at the Hospital de São João, in Porto. Fetal Cardiotocography is one of the most important methods of assessment of fetal well-being. CTG signals contain information about the fetal heart rate (FHR) and uterine contractions (UC).

Five antepartum FHR with a median duration of 70 min (min-max: 59 - 103) were obtained and analyzed by the SisPorto® system. These cases corresponded to a median gestational age of 39 weeks and 5 days (min-max: 35 weeks and 4 days - 42 weeks and 1 day).

Table 4.7: Detection delay time of the ACWM-fh on the industrial data set.

	Fading Factor (α)		
True Change	**1**	**0.999998**	**0.999994**
45000	906	901	910
90000	988	1105	1331
210000	2865	2571	2100
255000	9142	8763	8452
375000	2496	2235	1806
420000	1340	1114	1018
Average	**2956**	**2782**	**2603**

The SisPorto® system, developed at INEB (Instituto Nacional de Engenharia Biomédica), starts the computer processing of CTG features automatically after 11 min of tracing acquisition and updates it every minute (Ayres-de Campos et al., 2008), providing the FHR baseline estimation, identifying accelerations and decelerations and quantifying short-term and long-term variability according to algorithms described in Ayres-de Campos et al. (2000). Along with these features, the system also triggers alerts, such as "Normality criteria met alert", "Non-reassuring alerts" and "Very non-reassuring alerts" (further details can be founded in Ayres-de Campos et al. (2000)). However, the system usually takes about 10 min to detect these different behaviors. In the "Normal" stage of FHR tracing four different patterns may be considered (Gonçalves et al., 2007):

- *A* - corresponding to calm or non-eye movement (REM) sleep;
- *B* - active or rapid eye movement (REM) sleep;
- *C* - calm wakefulness;
- *D* - active wakefulness;

Figure 4.10 shows an example of the analysis of a CTG exam exactly as it is produced by the SisPorto® system. The FHR and UC tracings are represented at the top and at the bottom, respectively. The FHR baseline estimation, accelerations and decelerations and different alerts stages also can be observed in this figure. The "Normal" stage is represented with a green bar in between the FHR and UC tracings. The "Suspicious" stage is represented with yellow and orange bars and the "Problematic" stage with a red bar.

The aim is to apply the CWM for this clinical data (with an adaptive and a fixed evaluation step) and assess whether the changes detected are in accordance with the changes identified

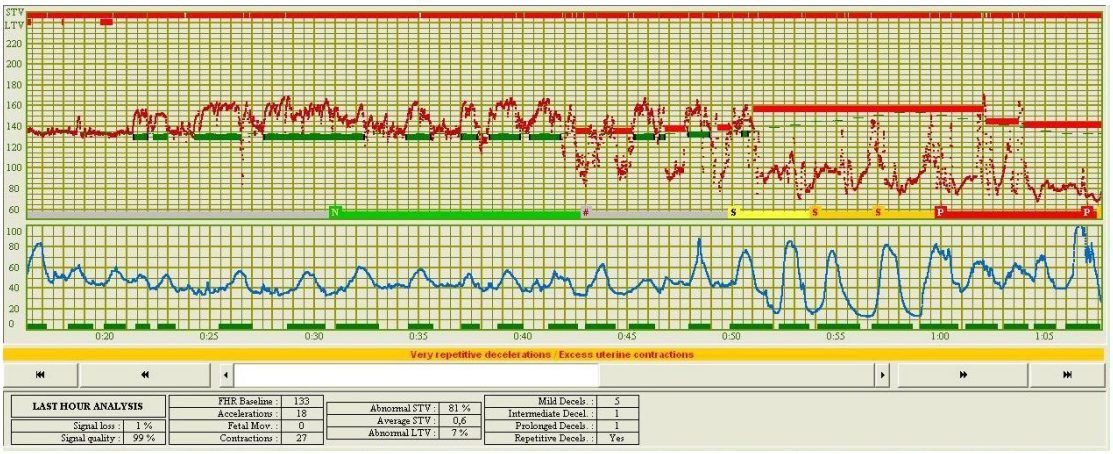

Figure 4.10: FHR (top) and UC (bottom) tracings. This figure also includes the FHR baseline estimation, accelerations and decelerations and patterns classification.

by the SisPorto® system. Ideally these changes should be detected earlier with CWM. The CWM was applied to the FHR tracings.

The data was represented by online histograms, with an admissible mean square error of 5% of the range of the data. The input parameters for the CWM were the following:

- The length of the reference window was set to $10k$, where k is the number of buckets in the online histograms;
- The initial evaluation interval was set to $k/2$, where k is the number of buckets in the online histograms;
- The threshold for detecting changes was set to 5%;

The achieved results are consistent with the system analysis and the CWM detects the changes between the different stages earlier than the SisPorto® system. Further than the analysis of this program, the method is able to detect some changes between different patterns of the "Normal" stage. Due to difficulties in ascertaining the exact change points between these behaviors it is not possible to perform a detection delay evaluation. However the preference of an adaptive evaluation step is again supported by the detections results in this data set.

4.3.2 Concept Changes

Concerning the detection of concept drifts, a comparison of the ACWM with three well known methods taken from literature was undertaken, namely:

- Drift Detection Method (DDM), presented by Gama et al. (2004);
- ADaptive WINDdowing (ADWIN) method, introduced by Bifet and Gavaldà (2007);
- Page-Hinkley Test (PHT), described in Page (1954);

Drift detection method (DDM) This online drift detection method monitors the trace of the error rate of an online classifier, for streaming observations, and considers that the error rate follows the binomial distribution. At each time t, the error rate of the online classifier is the probability of misclassifying, p_t, with a standard deviation $s_t = \sqrt{p_t(1-p_t)/t}$. According to the *Probability Approximately Correct* (PAC) learning model, this method assumes that in a stationary concept, the error rate decreases with the number of observations. Therefore an increase in the error rate indicates a change in the concept. While monitoring the error rate, the DDM stores p_{min} and s_{min}, which correspond to the minimum probability and minimum standard deviation (respectively), and are obtained when $p_t + s_t$ reaches its minimum value. Based on these minimum values, the DDM establishes two levels as follows:

- The warning level: when $p_t + s_t \geq p_{min} + 2s_{min}$;
- The drift level: $p_t + s_t \geq p_{min} + 3s_{min}$;

When the error rate exceeds the lower threshold, the system enters in a warning mode and stores the observations within the warning level in a short-term memory. If the error drops below the threshold again, the warning mode is cancelled. However, if the error increases reaching the second (higher) threshold, a change in the concept is assigned. The online classifier is retrained using only the observations in the buffer and reinitializes the variables. The pseudocode for this algorithm is presented in Appendix B.1.

ADaptive WINdowing (ADWIN) The ADaptive WINDdowing method keeps a sliding window W (with length n) with the most recently received examples and compares the distribution on two sub-windows of W. Whenever two *large enough* sub-windows, W_0 and W_1, exhibit *distinct enough* averages, the older sub-window is dropped and a change in the distribution of examples is assigned. The window cut threshold is computed as follows:

$\epsilon_{cut} = \sqrt{\frac{1}{2m} ln \frac{4}{D}}$, with $m = \frac{1}{1/n_0 + 1/n_1}$, where n_0 and n_1 denote the lengths of W_0 and W_1.

A confidence value D is used within the algorithm, which establishes a bound on the false positive rate. However, as this first version was computationally expensive, the authors

propose to use a data structure (a variation of exponential histograms), in which the information on the number of 1's is kept as a series of buckets (in the Boolean case). It keeps at most M buckets of each size 2^i, where M is a user defined parameter. For each bucket, two (integer) elements are recorded: *capacity* and *content* (*size* or the number of 1s it contains). The pseudocode for ADWIN algorithm is presented in Appendix B.2.

Page Hinkley Test (PHT) The Page-Hinkley Test (PHT) (Page, 1954) is a sequential analysis technique typically used for monitoring change detection in the average of a Gaussian signal (Mouss et al., 2004). This test considers a cumulative variable U_T defined as the accumulated difference between the observed values and their mean until the current moment:

$$U_T = \sum_{t=1}^{T} (x_t - \bar{x}_T - \delta)$$

where $\bar{x_T} = 1/T \sum_{t=1}^{T} x_t$ and δ corresponds to the magnitude of changes that are allowed. To detect increases, it computes the minimum value of U_t: $m_T = min(U_t, t = 1 \ldots T)$ and monitors the difference between U_T and m_T: $PH_T = U_T - m_T$. When the difference PH_T is greater than a given threshold (λ) a change in the distribution is assigned. The threshold λ depends on the admissible false alarm rate. Increasing λ will entail fewer false alarms, but might miss or delay some changes. Controlling this detection threshold parameter makes it possible to establish a trade-off between the false alarms and the missed detections. The pseudocode for the PHT is presented in Appendix B.3.

4.3.2.1 Controlled Experiments with Artificial Data

To assess the performance of these methods in detecting concept changes in different scenarios, different experiments were carried out. The number of false alarms, the missed detections and detection delay time were evaluated using data underlying a Bernoulli distribution and a public data set.

This set of experiments uses data streams of lengths L = 2.000, 5.000 and 10.000, underlying a stationary Bernoulli distribution of parameter $\mu = 0.2$ during the first $L - 1.000$ examples. During the last 1.000 examples the parameter is linearly increased to simulate concept drifts with different magnitudes. The following slopes were used: 0 (no change), 10^{-4}, 2.10^{-4}, 3.10^{-4} and 4.10^{-4}. For each type of simulated drift, 100 data streams were generated with different seeds. These experiments also allow to analyze the influence, in the delay time until detections, of the length of the stationary part (the first $L - 1.000$ samples).

Table 4.8: Average detection delay time (DDT), number of false alarms ($\#FA$) and the number of missed detections ($\#MD$), for the four methods, using the data streams with lengths 2.000, 5.000 and 10.000 and with different slopes in the Bernoulli parameter distribution. For $slope = 0$ (no change) the measurements DDT and $\#MD$ are not applicable.

Length	Slope	ADWIN			DDM			PHT			ACWM-fh ($\alpha = 0.9994$)		
		DDT	#FA	#MD	DDT	#FA	#MD	DDT	#FA	#MD	DDT	#FA	#MD
2.000	0	(n.a.)	5	(n.a.)	(n.a.)	0	(n.a.)	(n.a.)	4	(n.a.)	(n.a.)	0	(n.a.)
	1.10^{-4}	582	0	3	627	0	2	573	0	3	687	0	2
	2.10^{-4}	578	0	0	687	0	16	523	0	0	752	0	4
	3.10^{-4}	428	0	0	537	0	0	397	0	0	531	0	0
	4.10^{-4}	359	0	0	534	0	0	331	0	0	530	0	0
5.000	0	(n.a.)	17	(n.a.)	(n.a.)	17	(n.a.)	(n.a.)	41	(n.a.)	na	0	(n.a.)
	1.10^{-4}	722	16	30	866	21	77	650	23	13	849	0	27
	2.10^{-4}	512	13	13	732	19	37	463	25	0	632	0	0
	3.10^{-4}	383	14	14	668	20	17	337	32	0	539	0	0
	4.10^{-4}	320	10	10	587	9	12	279	29	0	273	0	0
10.000	0	(n.a.)	15	(n.a.)	(n.a.)	44	(n.a.)	(n.a.)	68	(n.a.)	(n.a.)	20	(n.a.)
	1.10^{-4}	722	19	35	829	39	94	650	60	10	828	14	54
	2.10^{-4}	505	19	19	843	56	57	466	71	0	678	15	5
	3.10^{-4}	401	17	17	720	29	53	344	68	0	576	16	1
	4.10^{-4}	327	23	23	642	52	41	280	66	0	507	22	6

Table 4.8 shows a summary of the performance of the four methods compared: ADWIN, DDM, PHT and ACWM-fh (with fading factor $\alpha = 0.9994$). The rows are indexed by the value of L and corresponding slope, presenting the delay time (DDT) until the detection of the change that occurs at time stamp $L - 1.000$ (averaged over the 100 runs), the total number of missed detections ($\#MD$) and the total number of false alarms ($\#FA$).

For different stream lengths, the first row (slope 0) gives the number of false alarms. The PHT tends to present more false alarms than any of the other methods. The ACWM-fh only presents false alarms for streams with a length of 10.000. For these cases, the number of false alarms of ACWM-fh and ADWIN is similar.

In general, the increase of the data streams length leads to an increase in the number of false alarms and missed detections. As is reasonable for all the methods, the increase in the slope of Bernoulli's parameter contributes to a decrease in the time until the change is detected. Fewer false alarms and missed detections resulted also from slope increases. For all the streams and looking at the detection delay time, the ADWIN wins over DDM, presenting a similar number of false alarms and missed detections. For detection delay time, in all the cases, the PHT outperforms the ADWIN. Additionally, in most cases, PHT does not miss changes. However, PHT results are compromised with the highest number of false alarms among the four methods. In a concept drift problem, when a change detector is embedded in a learning algorithm, this is a huge drawback. The occurrence of a concept drift implies the relearning of a new model in order to keep up with the current state of nature. In the presence of a false detection, the model, which is up-to-date, will

be unnecessarily replaced by a new one. On the other hand, in learning scenarios, missed detections are also harmful. They entail outdated models that are not describing the new evolving data.

Regarding this trade-off between false alarms and missed detections, the ACWM-fh presents the best results, with detection delay times almost as low as the ADWIN.

4.3.2.2 Experiments on a Public Data Set

In the previous experiment, the data set did not allow the performance of the different change detection methods to be evaluated in large problems, which is important since concept drift mostly occurs in huge amounts of data arriving in the form of streams. To overcome this drawback, an evaluation of the change detection algorithms was performed using the SEA concepts data set (Street and Kim, 2001), a benchmark problem for concept drift. Figure 4.11 shows the error rate (computed using a naive-Bayes classifier), which presents three drifts. The drifts and the corresponding detections, signed by the analyzed methods, are represented by dashed and solid lines, respectively.

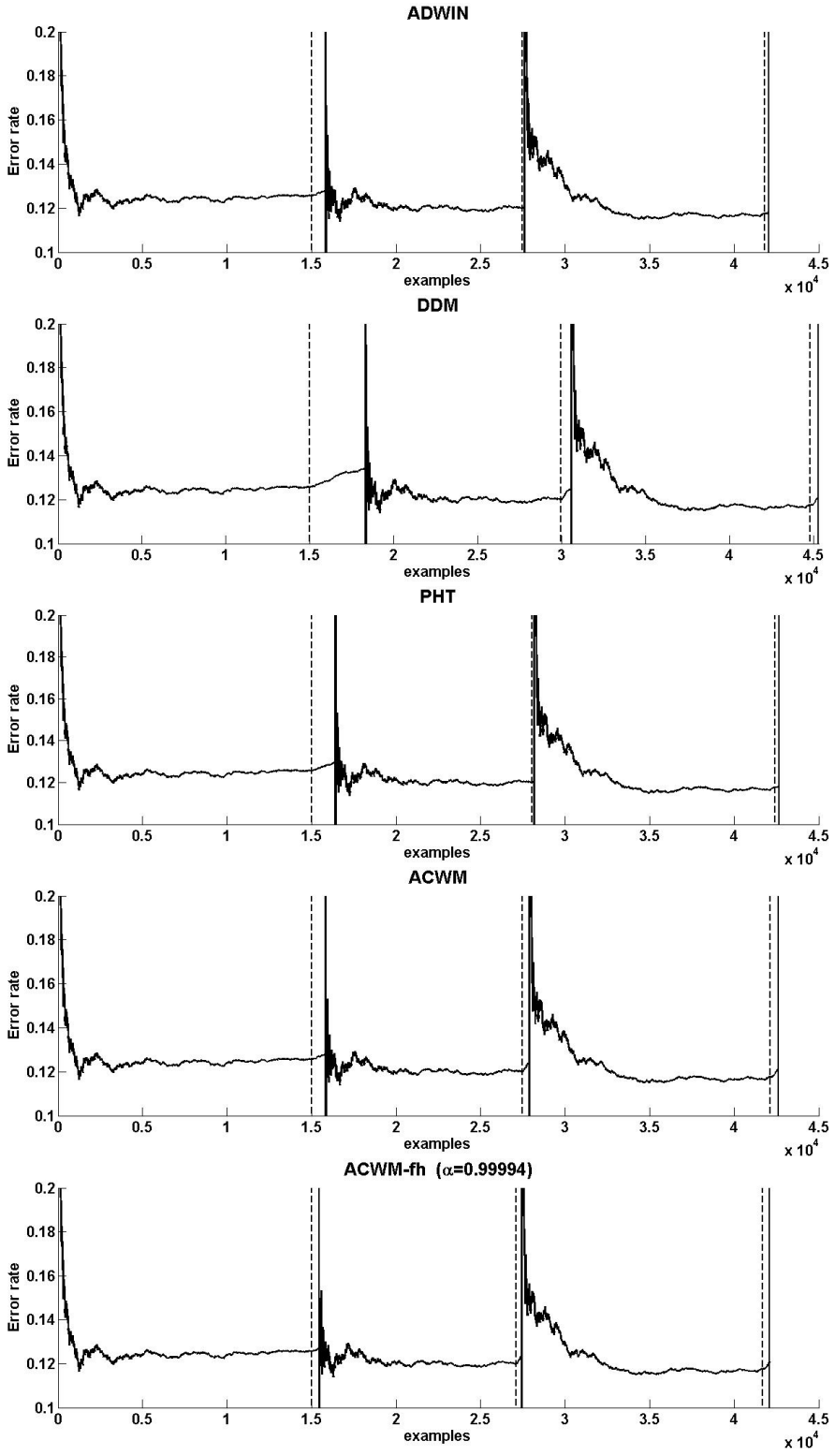

Figure 4.11: Evolution of the error rate and the delay times in drift detection using the four presented methods (ACWM was performed with two variants). Vertical dashed lines indicate drift in data, and vertical solid lines indicate when drift was detected.

Table 4.9 presents the delay time in detecting concept drifts in this data set. In can be seen that all the algorithms require too many examples to detect the first drift. The exception is ACWM-fh, which takes only almost half the time of the second "best" methods (ACWM and ADWIN) to detect the first drift. For all the methods, the resilience to false alarms and the ability to reveal changes without missing detections must be stressed.

Table 4.9: Detection delay time of the compared methods, when performed in the SEA data set.

	Detection Delay Time				
# **Drift**	ADWIN	DDM	PHT	ACWM	ACWM-fh
1	826	3314	1404	818	442
2	115	607	118	620	343
3	242	489	249	489	434

Comparing the performance of both ACWM, the approach using fading histograms, has a clear advantage. The detection delay time is significantly reduced for the first and second drifts. For the second and third drifts, the performance of the ACWM is comparable to the performance of the DDM; while for the first drift the detection delay time is similar to the one presented by ADWIN. The ACWM-fh clearly performs better than DDM and PHT. When compared to ADWIN, although it wins in the detection of the first drift, it is beaten on the second and on the third.

In evolving learning scenarios, the time required to process examples plays an important role. When comparing the MATLAB execution time of these methods, the ACWM performs best, as shown it Table 4.10. It must be pointed out that ADWIN was performed in MATLAB but the code was implemented in JAVA, which may increase the execution time.

Table 4.10: MATLAB execution times when performing the methods analyzed.

	Execution Time				
# **Drift**	ADWIN	DDM	PHT	ACWM	ACWM-fh
1	0.942387	0.938678	2.478019	0.034037	0.040603
2	0.614463	0.477378	1.260515	0.000695	0.004226
3	0.736048	0.621921	1.882912	0.044618	0.046027

4.4 Conclusions & Research Question

This chapter presented the CWM, which is a windowing scheme to detect changes through the monitoring of data distributions over two time windows. The performance of the

CWM was evaluated in different evolving scenarios, using artificial and real data, revealing its suitability to detect distribution changes as well as concept changes. Research question 2 enquires about distribution changes and was therefore addressed in this chapter:

1. *In the development of a model to detect changes through the comparison of distributions over two time windows, which is the appropriate step to perform comparisons?*
   The reasoning presented in Section 4.2.3 leads to an adaptive evaluation step, which is automatically defined according to the data stationarity and according to the distance between data distributions. The experimental designs with artificial data sustain this decision based on the decrease of the detection delay time and on comparable execution times of performing the CWM.

2. *When evaluating the distance between distributions, how do the forgetting rates of fading histograms affect the detection delay time?*
   The ability of the fading histograms to forget outdated data is preferred when performing the ACWM, by decreasing the detection delay time without compromising the number of missed detections and false alarms. Moreover, the advantage of using fading histograms is more evident when detecting small changes.

3. *What is the robustness against noise of the proposed change detection model?*
   The experiments designed sustain that the ACWM is robustness against noise while effectively detects distribution changes in the data.

4. *What is the effect of the extension of a stationary phase in the performance of the proposed change detection model?*
   Overall, a long stationary phase will entail a greater detection delay time, compromising the ability to effectively detect changes. However, since the ACWM compares data distributions to assess changes, this drawback can be overcome by forgetting outdated data using fading histograms, but this compromises the number of false alarms and missed detections. Hence, good stability in stationary phases and a good detection of distributions changes can be achieved establishing a trade-off between detection delay time and false alarms/missed detections. However, these conceptions are contradictory and this trade-off may not be easy to define.

- 5 -

New Criteria for Learning from Data Streams

"For the things we have to learn before we can do them,
we learn by doing them."
Aristotle (384 BCE - 322 BCE)

The ability to gather data is changing drastically: nowadays, huge amounts of unbounded data streams are produced at high-speed rate. Therefore, the learning algorithms need to be modified to accommodate this new data. Moreover, this unbounded data is generated in non-stationary environments, which is even more challenging. Hence, the golden standards to evaluate learning algorithms must be revised. In this chapter, the general framework for evaluating learning algorithms found in Gama et al. (2013) is presented, proposing the use of forgetting prequential error estimates in different assignments. The forgetting strategy in the computation of the error estimates is achieved either by computing the error over a sliding window or through the use of fading factors in the error estimation. Furthermore, for consistent learners, convergence proof of different error estimates is introduced. The performance of stream learning algorithms is compared by applying the McNemar test over the proposed forgetting error estimates. The detection of concept drift is performed throughout the monitoring of forgetting prequential error estimates and on the ratio of these. The experimental data was obtained in MOA (Bifet et al., 2010), the change detection tests were implemented in MATLAB (MATLAB®& Simulink®, 2007) and the graphics were produced in R (R Development Core Team, 2008).

This chapter is structured as follows: the objectives and methodology of research question 3 begin this chapter. Next, Section 5.2 is devoted to new strategies for evaluating stream learning algorithms and presents the proofs of convergence of the proposed error estimates to the Bayes error. Section 5.3 presents a comparison of the performance of learning

algorithms. Before presenting the overall conclusions in Section 5.5, Section 5.4 addresses concept drift detection, proposing two approaches based on forgetting mechanisms.

5.1 Research Question

The main problem in evaluation methods when learning from time-changing data streams is the correct monitoring of the evolution of the learning process. Therefore, raises the research question 3:

1. How should the performance of stream learning algorithms be evaluated?
2. Can forgetting mechanisms provide reliable error estimates?
3. How can the performance of learning algorithms in non-static environments be compared?
4. How can forgetting mechanisms be extended to cope with concept drift problems?

Within this research question, the objective is to propose a new methodology to evaluate stream learning algorithms in the presence on non-stationary data. In particular, design forgetting mechanisms strategies to accomplish the following assignments:

1. Computation of error estimates in the flow.
2. Comparison of the performance of stream learning algorithms.
3. Concept drift detection.

The above objectives are accomplished with the following methodology:

1. Design of experimental work to evaluate and compare decision models that evolve over time.
2. Design a set of experiments to assess the ability of the Page-Hinkley test to detect drift using forgetting mechanisms, namely with two approaches:
   (a) Monitoring drift using error estimates computed with forgetting mechanisms.
   (b) Monitoring drift with the ratio of error estimates computed with forgetting mechanisms.

5.2 Evaluation of Stream Learning Algorithms

Just as the main properties of data streams render batch learning systems useless, the characteristics of the newly developed stream learning algorithms render traditional techniques for evaluating their performance inadequate. In the batch learning scenario, cross-validation and variants (leave-one-out, bootstrap) are the standard methods of evaluating the learning algorithms. Those evaluation methods are only appropriate for restricted size data sets and are not suitable to be used in open-ended data streams contexts.

Therefore, this section presents new strategies to evaluate algorithms while learning from data streams. At the same time, the first and the second parts of research question 3 are addressed.

Suppose there is a sequence of examples in the form of pairs (x_i, y_i). For each example, the current decision model predicts $\hat{y}_i$, that can be either True ($\hat{y}_i = y_i$) or False ($\hat{y}_i \neq y_i$). For each time i in the sequence, the expected value of the error rate is p_i. For a set of examples, the error (e_i) is a random variable from Bernoulli trials. The Binomial distribution gives the general form of the probability for the random variable that represents the number of errors in a sample of examples: $e_i \sim \text{Bernoulli}(p_i) \Leftrightarrow Prob(e_i = False) = p_i$.

Definition 5.1. Consistent learner
In the *Probability Approximately Correct* (PAC) learning model (Kearns and Vazirani, 1994), a learner is called consistent if, for a sufficiently large number of independent examples generated by a stationary distribution, it outputs a hypothesis with error arbitrarily close to the Bayes error ($B + \epsilon$), with at least probability $1 - \delta$:

$$Prob(e_i - B < \epsilon) \geq 1 - \delta$$

In fact, if the distribution of the examples is stationary and the examples are independent, the error rate of a consistent learning algorithm (p_i) will decrease when the number of training examples, i, increases. With probability greater than or equal to $1 - \delta$, for an infinite number of training examples, the error rate will tend to approximate the Bayes error (B). Which means that for every real number $\epsilon > 0$, there exists a natural number N_1, so that for every $i > N_1$ it results, with a probability of greater than or equal to $1 - \delta$, that $|p_i - B| < \epsilon$:

$$\forall \epsilon > 0, \quad \exists N_1 \in \mathbb{N} : \quad \forall i > N_1 \quad |p_i - B| < \epsilon$$

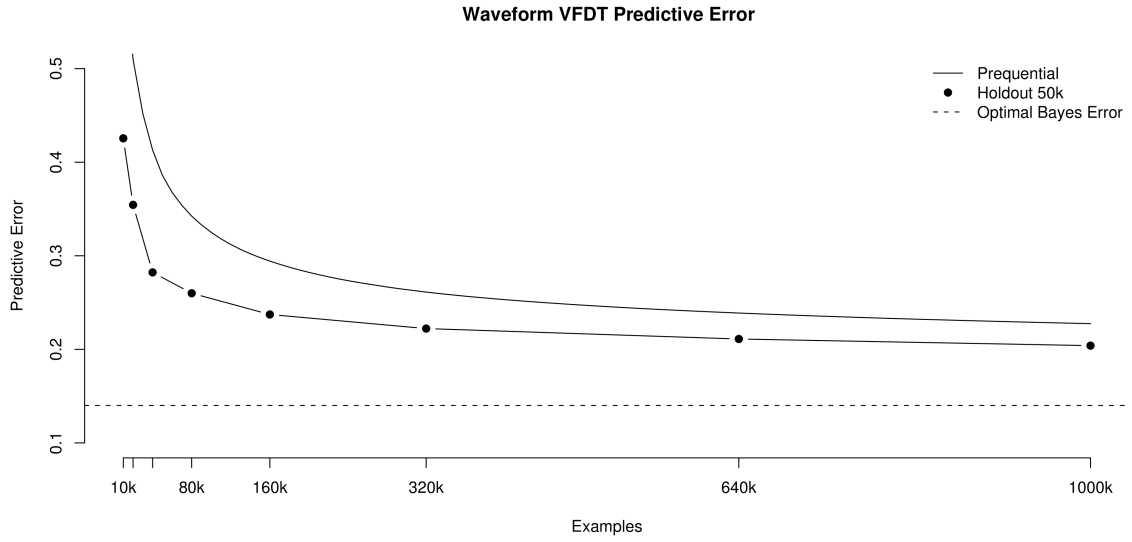

Figure 5.1: Comparison of error evolution as estimated by holdout and prequential strategies, in a stationary stream (waveform data set). The learning algorithm is VFDT.

For example, Duda and Hart (1973) prove that the k-nearest neighbor algorithm is guaranteed to yield an error rate no worse than the Bayes error rate as data approaches infinity. Bishop (1995) presents similar proofs for feed-forward neural networks, and Mitchell (1997) for decision trees.

Henceforward, the mathematical proofs presented are applied to consistent learners.

Within the context of stream learning, two viable methodologies to evaluate stream learning algorithms are the *prequential* method and the *holdout* sampling strategy.

In the *predictive sequential* (or *prequential*) (Dawid, 1984) method, the stream decision model is evaluated through the evolution of the error rate.

In the holdout strategy, the decision model is applied to a test set at regular time intervals (or set of examples). For a large enough test set, the loss estimated in the holdout is an unbiased estimator.

Figure 5.1 shows a comparison of error evolution as estimated by these two strategies on the waveform data set problem, for the Very Fast Decision Tree (VFDT) algorithm.

Definition 5.2. Holdout error estimate

In a holdout test set with M examples, the error estimate is computed as:

$$H_e(i) = \frac{1}{M} \sum_{k=1}^{M} L(y_k, \hat{y}_k) = \frac{1}{M} \sum_{k=1}^{M} e_k.$$

Theorem 5.3. ***Limit of the holdout error estimate***
For consistent learning algorithms and for large enough [*1] *holdout sets, the limit of the error estimated in the holdout is the Bayes error:* $\lim_{i\to\infty} H_e(i) = B$.

Proof. Assuming that at time i the probability of observing a false is p_i, the errors in the holdout test are independent and identically distributed (i.i.d.) random variables, all Bernoulli distributed with success probability p_i: $e_k \sim \text{Bernoulli}(p_i), \forall k = 1, \ldots, M$, where the expected value of e_k is p_i: $E(e_k) = p_i$. Then, from the *Law of Large Numbers*, the average obtained from a large number of trials ($H_e(i)$) converges to the expected value:

$$\forall \epsilon > 0, \quad \exists N_1 \in \mathbb{N} : \quad \forall i > N_1 \quad \left| \frac{1}{M} \sum_{k=1}^{M} e_k - E(e_k) \right| < \epsilon$$

$$\Leftrightarrow |H_e(i) - E(e_k)| < \epsilon \Leftrightarrow |H_e(i) - p_i| < \epsilon.$$

Since for an infinite number of examples, the error rate of the learning algorithm (p_i) will tend to approximate the Bayes error (B), the result is:

$$\forall \epsilon > 0, \quad \exists N_1 \in \mathbb{N} : \quad \forall i > N_1 \quad |H_e(i) - B| < \epsilon \Leftrightarrow \lim_{i\to\infty} H_e(i) = B.$$

□

For each example in the stream, the current model makes a prediction based only on the example attribute-values. The prequential method computes the error of the model from that sequence of examples.

Definition 5.4. Prequential error estimate
The prequential error estimate, computed at time i, is based on an accumulated sum of a loss function between the prediction and observed values:

$$P_e(i) = \frac{1}{i} \sum_{k=1}^{i} L(y_k, \hat{y}_k) = \frac{1}{i} \sum_{k=1}^{i} e_k.$$

Theorem 5.5. Limit of the prequential error estimate
For consistent learning algorithms, the limit of the prequential error estimate is the Bayes error: $\lim_{i\to\infty} P_e(i) = B$.

[*1] A window large enough to achieve an unbiased estimator of the average must be used.

Proof. Using simple algebraic manipulation:

$$|P_e(i) - B| = \left|\frac{1}{i}\sum_{k=1}^{i} e_k - B\right| = \left|\frac{1}{i}\sum_{k=1}^{i}(e_k - B)\right|$$

$$= \left|\frac{1}{i}\sum_{k=1}^{N_1}(e_k - B) + \frac{1}{i}\sum_{k=N_1+1}^{i}(e_k - B)\right| \leq \left|\frac{1}{i}\sum_{k=1}^{N_1}(e_k - B)\right| + \left|\frac{1}{i}\sum_{k=N_1+1}^{i}(e_k - B)\right|$$

Let $\epsilon > 0$. Then, there exists $N_1 \in \mathbb{N}$, so that for any $i > N_1$, the result is:

$$Prob(e_i \sim \mathrm{Ber}(B)) \geq 1 - \epsilon.$$

Therefore, from the *Law of Large Numbers*, the average obtained from a large number of trials converges to the expected value (B):

$$\left|\frac{\sum_{k=N_1+1}^{i} e_k}{i - N_1} - B\right| < \frac{\epsilon}{2}$$

Hence, for $i > N_1$, the result is:

$$\left|\frac{1}{i}\sum_{k=1}^{N_1}(e_k - B)\right| + \left|\frac{1}{i}\sum_{k=N_1+1}^{i}(e_k - B)\right| <$$

$$< \left|\frac{1}{i}\sum_{k=1}^{N_1}(e_k - B)\right| + \frac{(i - N_1)\epsilon}{2i} < \left|\frac{1}{i}\sum_{k=1}^{N_1}(e_k - B)\right| + \frac{\epsilon}{2}$$

Considering $N_2 > N_1 \in \mathbb{N}$, such that $\forall i > N_2 : \quad \frac{1}{i}\sum_{k=1}^{N_1}|e_k - B| < \frac{\epsilon}{2}, \quad \forall \epsilon > 0$, the result is:

$$\exists\{N_1, N_2\} \in \mathbb{N} : \quad \forall i > N_2 : \left|\frac{1}{i}\sum_{k=1}^{i} e_k - B\right| < \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon, \quad \forall \epsilon > 0$$

$$\Leftrightarrow \lim_{i\to\infty} P_e(i) = B$$

□

Prequential evaluation provides a learning curve that monitors the evolution of learning as a process. Figure 5.1 shows that using holdout evaluation, it is possible to obtain a similar

curve by applying the current model to the holdout set, at regular time intervals. Both error estimates can be affected by the order of the examples.

5.2.1 Error Estimates using a Forgetting Mechanism

The prequential error estimate is pessimistic because it is computed over the stream hitherto and therefore is strongly influenced by the first part of the error sequence, when few examples have been used to train the learning algorithm.

Stream learning models evolve over time, improving their performance with more labeled data. Therefore, it is straightforward that while estimating the prequential error older errors should contribute less than recent ones. This means that a prequential error estimate computed with a forgetting mechanism would be more accurate than the one computed over the entire stream of errors. This can be accomplished either by using a sliding window of the most recently observed errors or by applying fading factors. With a sliding window of size infinite or a fading factor equal to 1, these forgetting error estimators equal the prequential estimator.

5.2.1.1 Error Estimators using Sliding Windows

Sliding windows are one of the most commonly used forgetting strategies. They are used to compute statistics from the most recent past.

Definition 5.6. Sliding prequential error estimate
The prequential error is computed, at time i, over a sliding window of size w ($\{e_j : j \in]i-w, i]\}$) as:

$$P_w(i) = \frac{1}{w} \sum_{k=i-w+1}^{i} L(y_k, \hat{y}_k) = \frac{1}{w} \sum_{k=i-w+1}^{i} e_k.$$

Theorem 5.7. Limit of the prequential error estimate computed over a sliding window
*For consistent learning algorithms, the limit of the prediction error computed over a sliding window of (large enough *2) size w is the Bayes error:* $\lim_{i\to\infty} P_w(i) = B$.

Proof. Using simple algebraic manipulation:

*2 A window large enough to achieve an unbiased estimator of the average must be used.

$$|P_w(i) - B| = \left| \frac{1}{w} \sum_{k=i-w+1}^{i} e_k - B \right| =$$

$$= \left| \frac{1}{w} \sum_{k=i-w+1}^{N_1} (e_k - B) + \frac{1}{w} \sum_{k=N_1+1}^{i} (e_k - B) \right| \leq \left| \frac{1}{w} \sum_{k=i-w+1}^{N_1} (e_k - B) \right| + \left| \frac{1}{w} \sum_{k=N_1+1}^{i} (e_k - B) \right|$$

From the *Law of Large Numbers*, it follows that:

$$\left| \frac{\sum_{k=N_1+1}^{i} e_k}{i - N_1} - B \right| < \frac{\epsilon}{2}.$$

Hence, for $i > N_1$:

$$\leq \left| \frac{1}{w} \sum_{k=i-w+1}^{N_1} (e_k - B) \right| + \left| \frac{1}{w} \sum_{k=N_1+1}^{i} (e_k - B) \right| <$$

$$< \left| \frac{1}{w} \sum_{k=i-w+1}^{N_1} (e_k - B) \right| + \frac{(i - N_1)\epsilon}{2w} < \left| \frac{1}{w} \sum_{k=i-w+1}^{N_1} (e_k - B) \right| + \frac{\epsilon}{2}$$

Considering $N_2 \in \mathbb{N}$ so that $\forall w > N_2$:

$$\frac{1}{w} \sum_{k=i-w+1}^{N_1} |e_k - B| < \frac{\epsilon}{2}, \quad \forall \epsilon > 0,$$

it results that:

$$\exists \{N_1, N_2\} \in \mathbb{N} : \forall i > N_1, \forall w > N_2 : \left| \frac{1}{w} \sum_{k=i-w+1}^{i} e_k - B \right| < \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon, \forall \epsilon > 0$$

$$\Leftrightarrow \lim_{i \to \infty} P_w(i) = B.$$

□

Lemma *The prequential error estimator, $P_e(i)$, is greater than or equal to the prequential error computed over a sliding window, $P_w(i)$, assuming a large enough sliding window $w \ll i$: $P_e(i) \geq P_w(i)$.*

Proof.

$$P_e(i) = \frac{1}{i}\sum_{k=1}^{i} e_k \Leftrightarrow P_e(i) = \frac{\sum_{k=1}^{i-w} e_k + \sum_{k=i-w+1}^{i} e_k}{i} \Leftrightarrow$$

$$\Leftrightarrow P_e(i) = \frac{(i-w)\frac{\sum_{k=1}^{i-w} e_k}{i-w} + w\frac{\sum_{k=i-w+1}^{i} e_k}{w}}{i} \Leftrightarrow$$

$$\Leftrightarrow P_e(i) = \frac{(i-w)\bar{e}_{i-w} + wP_w(i)}{i} \Rightarrow {}^{\star 3}P_e(i) \geq \frac{(i-w)\bar{e}_i + wP_w(i)}{i} \Leftrightarrow$$

$$\Leftrightarrow P_e(i) \geq \frac{(i-w)P_e(i) + wP_w(i)}{i} \Leftrightarrow i * P_e(i) - (i-w)P_e(i) \geq wP_w(i) \Leftrightarrow$$

$$\Leftrightarrow (i-(i-w)) * P_e(i) \geq wP_w(i) \Leftrightarrow P_e(i) \geq P_w(i)$$

□

5.2.1.2 Error Estimators using Fading Factors

Another approach to discount older information across time consists of using fading factors. The *fading sum* $S_\alpha(i)$ of observations from a stream x is computed at time i as:

$$S_\alpha(i) = x_i + \alpha \times S_\alpha(i-1)$$

where $S_\alpha(1) = x_1$ and α $(0 \ll \alpha \leq 1)$. This way, the *fading average* at observation i is then computed as:

$$M_\alpha(i) = \frac{S_\alpha(i)}{N_\alpha(i)} \tag{5.1}$$

where $N_\alpha(i) = 1 + \alpha \times N_\alpha(i-1)$ is the corresponding *fading increment*, with $N_\alpha(1) = 1$. An important feature of the fading increment is that:

$$\lim_{i\to\infty} N_\alpha(i) = \tfrac{1}{1-\alpha}.$$

This way, at each observation i, $N_\alpha(i)$ gives an approximated value for the weight given to recent observations used in the fading sum.

[*3] From the PAC learning theory the error rate of the learning algorithm will decrease, so the average of errors can be seen as a decreasing function and so if $N_1 < N_2 \Rightarrow \bar{e}_{N_1} > \bar{e}_{N_2}$.

Definition 5.8. Fading prequential error estimate

The prequential error computed at time i, with fading factor α, can be written as:

$$P_\alpha(i) = \frac{\sum_{k=1}^{i} \alpha^{i-k} L(y_k, \hat{y}_k)}{\sum_{k=1}^{i} \alpha^{i-k}} = \frac{\sum_{k=1}^{i} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}}, \text{ with } 0 \ll \alpha \leq 1.$$

Theorem 5.9. Limit of the prequential error estimate computed with fading factors

For consistent learning algorithms, the limit of the prequential error estimate computed with fading factors is approximately equal to the Bayes error: $\lim_{i\to\infty} P_\alpha(i) \approx B$*, with probability greater than or equal to* $1 - \delta$*.*

Proof. Using simple algebraic manipulation:

$$|P_\alpha(i) - B| = \left| \frac{\sum_{k=1}^{i} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}} - B \right| = \left| \frac{\sum_{k=1}^{N_1} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}} + \frac{\sum_{k=N_1+1}^{i} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}} - B \right|$$

From the proof of the limit of the prequential error estimate, it results:

$$\frac{\sum_{k=N_1+1}^{i} e_k}{i - N_1} = B.$$

Hence, $\exists N_1 \in \mathbb{N} : \quad \forall i > N_1$, it results:

$$\left| \frac{\sum_{k=1}^{N_1} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}} + \frac{\sum_{k=N_1+1}^{i} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}} - B \right| \approx \left| \frac{\sum_{k=1}^{N_1} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}} + \frac{\sum_{k=N_1+1}^{i} \alpha^{i-k} \bar{e}_k}{\sum_{k=1}^{i} \alpha^{i-k}} - B \right| =$$

$$= \left| \frac{\sum_{k=1}^{N_1} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}} + \frac{\sum_{k=N_1+1}^{i} \alpha^{i-k} B}{\sum_{k=1}^{i} \alpha^{i-k}} - B \right| = \left| \frac{\sum_{k=1}^{N_1} \alpha^{i-k} e_k}{\sum_{k=1}^{i} \alpha^{i-k}} + B \left(\frac{\sum_{k=N_1+1}^{i} \alpha^{i-k}}{\sum_{k=1}^{i} \alpha^{i-k}} - 1 \right) \right| \leq \epsilon, \quad \forall \epsilon > 0 \Leftrightarrow$$

$$\Leftrightarrow \lim_{i\to\infty} P_\alpha(i) \approx B$$

□

Furthermore, the prequential estimator computed using fading factors, $P_\alpha(i)$, will be lower than the prequential error estimator, $P_e(i)$.

The proof of the previous theorems assumes the stationarity of the data and the independence of the training examples. The main lesson is that any of these estimators converge, for an infinite number of examples, to the Bayes error. All these estimators can be used in any experimental study. However, these results are even more relevant when dealing with data with concept drift. Indeed, the above theorems and the memoryless advantage support the use of prequential error estimated using fading factors to assess performance of stream learning algorithms in presence of non-stationary data. This topic is addressed in Section 5.4.

5.2.1.3 Illustrative Example

The objective of this experiment is to illustrate the demonstrated convergence properties of the error estimates using the strategies described above. The learning algorithm is VFDT as implemented in MOA (Bifet et al., 2010). The experimental work was done using the *Waveform* and the *LED* (Bache and Lichman, 2013) data sets, because the Bayes-error is known: 14% and 26%, respectively. The *RandomRBF* (RBF) and *Random Tree* (RT) data sets available in MOA were also used. The Waveform stream is a three class problem defined by 21 numerical attributes, the LED stream is a ten class problem defined by 7 Boolean attributes, the RBF and the RT streams are two-class problems defined by 10 attributes.

Figure 5.2 plots the holdout error estimate, the prequential error estimate, the prequential error estimated using sliding windows of different sizes and the prequential error estimated using different fading factors. All the plots are averages from 10 runs of VFDT on data sets generated with different seeds. The most relevant fact is that the window size and the fading factor does not matter too much: the prequential error estimated using forgetting mechanisms always converges fast to the holdout estimate.

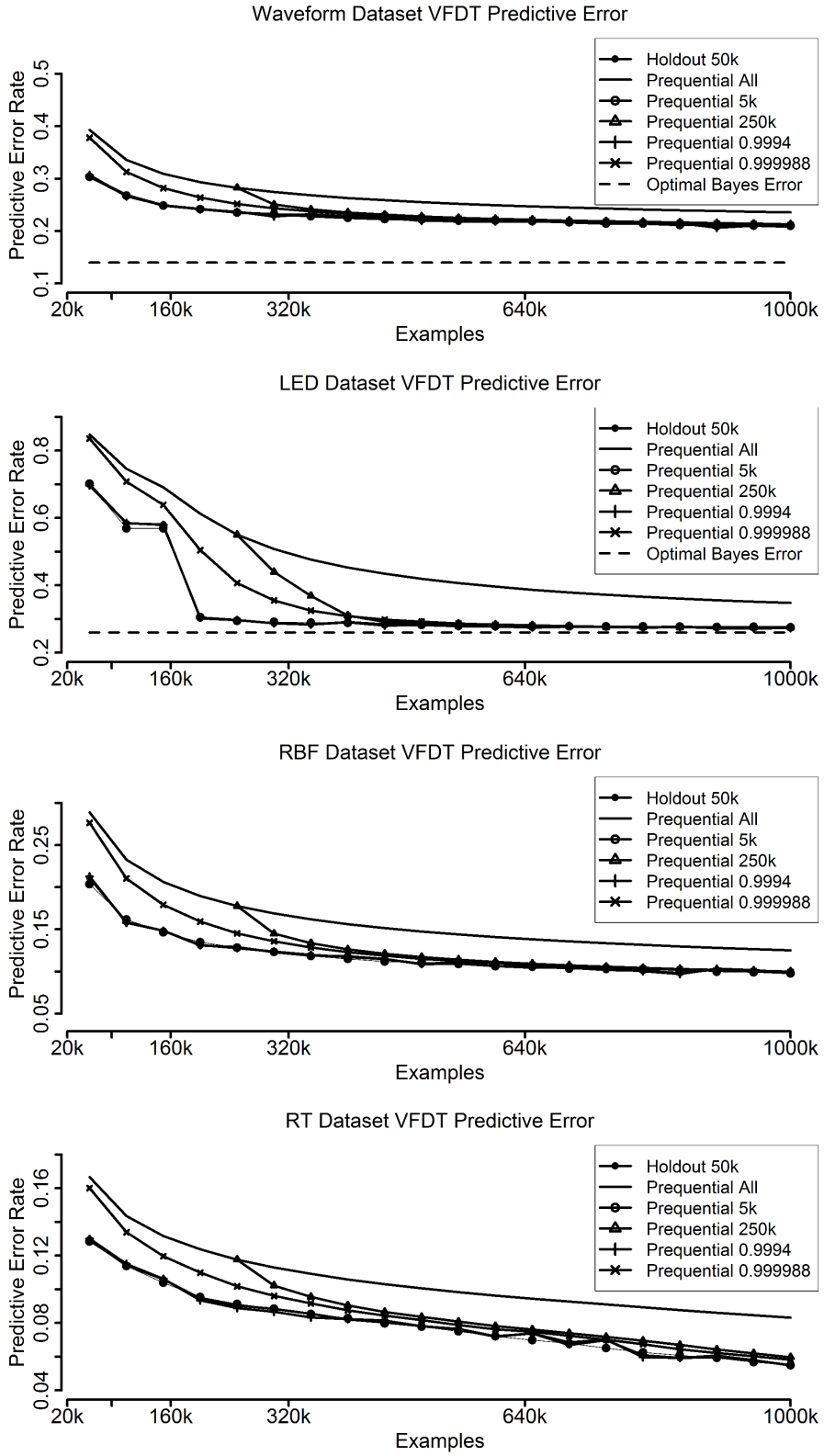

Figure 5.2: Comparison of error estimates evolution between holdout, prequential, prequential over sliding windows of different sizes and prequential using fading factors.

5.2.2 Learning in Evolving Environments

As the evolving scenario that generates data imposes that stream learning algorithms need to adapt the model along with the evolving data, standard techniques to evaluate learning algorithms have been rendered useless. The prequential method and the holdout strategy are not suitable for use in evolving scenarios for the following reasons:

- The prequential error reflects the overall accuracy but as it is long term influenced, is not able to capture the recovering phase of the algorithm fast.
- In the holdout strategy, the test set can be extracted from a different concept other than the one that is being learnt by the algorithm.

Two feasible alternatives that represent the recovery phase of a model fast are sliding windows and fading factors. Both methods have been used for blind model adaptation without explicit change detection, in drift scenarios (Klinkenberg, 2004; Koychev, 2000). Therefore, the presented forgetting error estimators are adequate strategies to evaluate stream learning algorithms in evolving environments.

5.3 Evaluation Comparison

After deriving metrics to evaluate the performance of stream learning algorithms, it is natural that the concerns about whether one algorithm is learning better than another prompt the following question: Is algorithm A more accurate than algorithm B? The comparison of the performance of algorithms can be assessed by comparing the proposed error estimates.

In this section, the third part of research question 3 is addressed through the design of experimental work to evaluate and compare decision models that evolve over time.

To compare the performance of two algorithms, in the batch scenario, one of the most commonly used tests is the McNemar test. McNemar's test is a non-parametric method used on nominal data. It assesses the significance of the difference between two correlated proportions, where the two proportions are based on the same finite sample. This test is applied to a contingency table (Dietterich, 1998), which tabulates the outcomes of two algorithms, as shown in Table 5.1:

Table 5.1: Contingency table when McNemar's test is applied.

# examples missclassified by both A & B ($n_{0,0}$)	# examples missclassified by A and not by B ($n_{0,1}$)
# examples misclassified by B and not by A ($n_{1,0}$)	# examples misclassified by neither A nor B ($n_{1,1}$)

Under the null hypothesis, both algorithms have the same error rate, hence $n_{0,1} = n_{1,0}$.

Although well established to compare the performance of correlated proportions in a finite set of examples, this test can be extended to evaluate the comparison of two stream learning algorithms: for that, error estimates must be computed in the flow. This will allow the contingency table to be updated incrementally, which is a desirable property in mining high-speed data streams. The McNemar test statistic is computed as follows:

$$M = sign(n_{0,1} - n_{1,0}) \times \frac{(n_{0,1}-n_{1,0})^2}{n_{0,1}+n_{1,0}}$$

and has a χ^2 distribution with 1 degree of freedom, under the null hypothesis. For a confidence level of 0.99, the null hypothesis is rejected if the statistic is greater than 6.635.

5.3.1 Illustrative Example

A benchmark problem for concept drift was used to compare the performance of two learning algorithms: the SEA concepts data set (Street and Kim, 2001). Figure 5.3 (top panel) shows the evolution of the prequential error of two naive-Bayes variants: a standard one and a variant that detects and relearns a new decision model whenever a drift is detected. The vertical dashed lines indicate drift in the data, and the vertical solid lines indicate when the drift was detected.

The McNemar test was performed to compare both algorithms. The bottom panel shows the evolution of the signed McNemar statistic computed for these two algorithms over the entire stream. As can be observed, once this statistic rises above the threshold value (6.635), it never falls below it, which is not informative about the dynamics of the process under study.

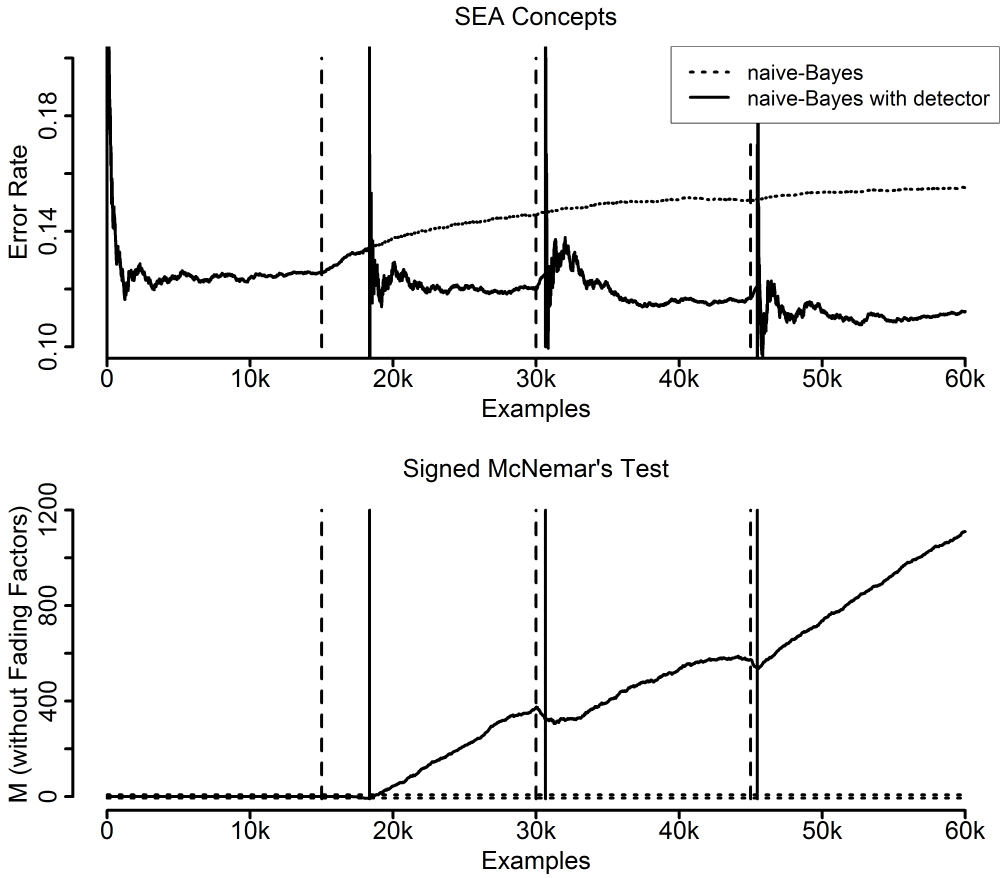

Figure 5.3: The evolution of signed McNemar statistic between two algorithms. Vertical dashed lines indicate drift in data, and vertical solid lines indicate when drift was detected. The top panel shows the evolution of the error rate of two naive-Bayes variants: a standard one and a variant that detects and relearns a new model whenever a drift is detected. The bottom panel shows the evolution of the signed McNemar statistic computed for these two algorithms.

It is well known, that the power of statistical tests, the probability of signaling differences where they do not exist, is highly affected by data length. Data streams are potentially unbounded, which might increase the number of Type II errors.

To overcome this drawback, and since the fading factors are memoryless and prove to exhibit similar behaviors to sliding windows, this statistical test was computed using different window sizes and fading factors.

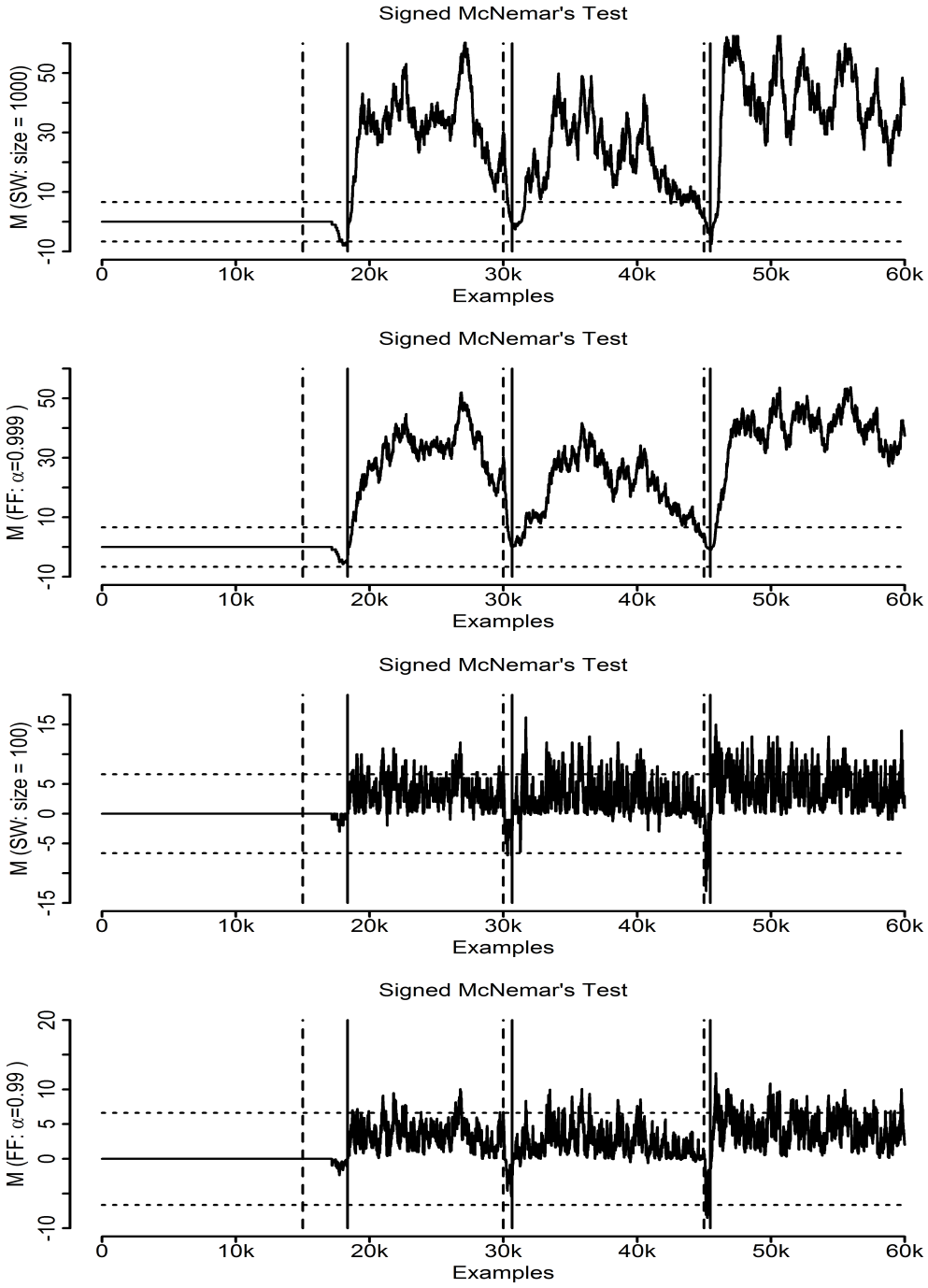

Figure 5.4: The evolution of signed McNemar statistic between two naive-Bayes variants. The top panels show the evolution of the signed McNemar statistic computed over a sliding window of 1000 examples and computed using a fading factor of $\alpha = 0.999$, and the bottom panels show the evolution of the signed McNemar statistic computed over a sliding window of 100 examples and computed using a fading factor of $\alpha = 0.99$. The dotted line is the threshold for a significance level of 99%. For different fading factors, different results for the significance of the differences are obtained.

Figure 5.4 presents a comparison of the evolution of a signed McNemar statistic between the two naive-Bayes variants, computed over a sliding window of 1000 and 100 examples and computed using a fading factor of $\alpha = 0.999$ and $\alpha = 0.99$. From the dotted line representing the threshold for a significance level of 99%, it can be observed that, for both strategies and almost at the same point, there is statistical evidence to reject the null hypothesis.

This statistical test shows to be feasible to compare stream learning algorithms in evolving scenarios by applying sliding windows or fading factors techniques. Nevertheless, these experiments point out that for different fading factors, as well as for different lengths of the windows, different results for the significance of the differences are obtained.

5.4 Evaluation under Concept Drift

The main problem in evaluation methods when learning from time-changing data streams is the monitoring of the evolution of the learning performance, which is a common strategy for concept drift detection (Klinkenberg, 2004; Street and Kim, 2001; Widmer and Kubat, 1996). Therefore, using forgetting error estimates, this section presents two strategies to detect drift, focusing on the fourth part of research question 3.

5.4.1 Monitoring Drift using Prequential Error Estimates

The forgetting error estimates proposed to evaluate the performance of stream learning algorithms have been shown to be more advantageous for the continuous assessment of the quality of stream learning algorithms. Therefore, the forgetting prequential errors can be used to assess drifts in evolving scenarios by monitoring their evolution. For this purpose, the Page Hinkley test was applied.

The experiments were done with data from Waveform, LED, RT and RBF generators, as implemented in MOA. The data was generated by emulating an abrupt concept drift event as a combination of two distributions. For the LED, Waveform and RBF data sets the first distribution was generated with the LEDGenerator, the WaveformGenerator and the RandomRBFGenerator (respectively) and the second distribution was generated with the LEDGeneratorDrift, the WaveformGeneratorDrift and the RandomRBFGeneratorDrift (respectively). For the second stream of LED and Waveform data sets, the number of attributes with drift were set to 7 and 21 (respectively). The second RBF stream was

$\mu \pm \sigma$	LED		RBF		RT		Waveform	
	MC	NBa	MC	NBa	MC	NBa	MC	NBa
Fading factors								
α=0.9970	2155±370	486±10	3632±4413	1416±342	911±132	800±84	1456±326	836±490
α=0.9985	3391±797	693±15	4288±4413	1992±430	1317±177	1163±121	2072±437	1129±565
α=0.9990	4279±947	872±19	6474±4995 (1;0)	2454±486	1676±212	1481±153	2678±541	1420±740
α=0.9993	5433±1130	1076±24	9241±7769 (1;0)	3023±571	2122±255	1861±190	3452±656	1766±952
α=0.9994	6103±1331	1183±26	9818±7587 (1;0)	3369±644	2365±279	2068±208	3874±718	1931±1002
Sliding windows								
w=1000	2542±512	725±14	3765±4444	1617±321	1157±120	1054±79	1637±290	1069±444
w=2000	3484±619	1146±30	4631±4430	2397±338	1866±156	1718±121	2491±353	1622±523
w=3000	4454±619	1503±36	7085±5205(1;0)	3151±344	2549±206	2331±164	3366±400	2158±713
w=4000	5436±804	1836±49	8315±5320 (1;0)	3987±384	3209±249	2925±203	4246±433	2654±824
w=5000	6419±860	2162±55	9198±5683 (2;0)	4899±466	3859±295	3507±237	5152±463	3133±918

Table 5.2: Detection delay time using PH test over different error estimates. The learning algorithms are VFDT-MC (MC) and VFDT-NBAdaptive (NBa). The results report the average and standard deviation of 10 runs. In parenthesis is the number of runs, if any, where PH test misses the detection or signals a false alarm: they are in the form (Miss; False Alarm).

generated setting the seed for the random generation of the model to 10 and adding speed drift to the centroids of the model (0.01). For the RT data set, both distributions were generated with the RandomTreeGenerator, varying the seed of the second concept. For the LED data stream the change occurs at example 128k and for the other streams the change occurs at example 32k.

The learning algorithms were VFDT-MC (VFDT using Majority Class for leaf prediction) and VFDT-NBAdaptive (VFDT using Naive Bayes Adaptive for leaf prediction), as implemented in MOA. The PH test parameters were $\delta = 0.01$ and $\lambda = 100$. Sliding windows of sizes 1k, 2k, 3k, 4k and 5k and fading factors of 0.9970, 0.9985, 0.9990, 0.9993 and 0.9994 were used to compute the forgetting prequential error estimates.

Table 5.2 presents a summary of the delay times in the drift detection on the aforementioned data sets, varying the parameters of the different prequential error estimates. The results refer to the average and standard deviation of 10 runs on streams generated with different seeds. These experiments highlight the advantage of using forgetting error estimators. The PH test using the prequential error computed over the entire stream only detects the change in the Waveform stream and VFDT-NBAdaptive learner (in all the runs), and misses the detection in all the other cases.

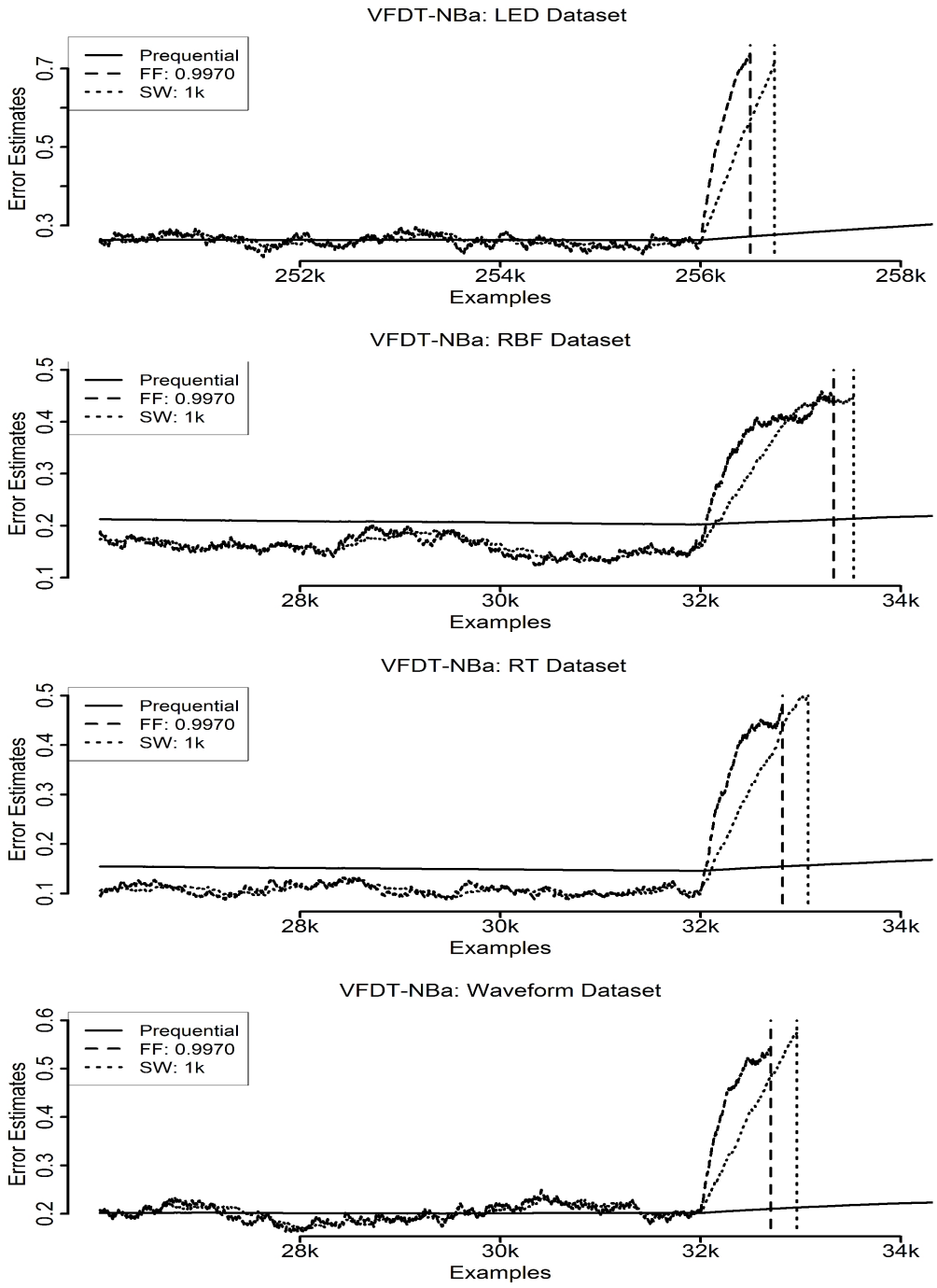

Figure 5.5: Change detection using PH test on prequential error estimates. The learning algorithm is VFDT-NBAdaptive. Each plot corresponds to a data stream and presents the evolution of prequential error estimates. The vertical lines indicate the point where change was detected. In these experiments, the fastest drift detection method is the prequential estimate based on fading factors.

The PH test computed over prequential error estimates using sliding windows or fading factors always detects the drift without any false alarm. The exception is on the RBF stream and VFDT-MC learner where, for a small number of runs (reported in parenthesis), the PH test was not able to detect the change. These results point out that the delay time in detection is increased by increasing the window size or the fading factor. The PH test detects much faster change points with VFDT-NBAdaptive than when using VFDT-MC. Moreover, there is some evidence that error estimates based on fading factors allow faster detection rates.

Figure 5.5 presents, for each stream, the 3 error estimates: the prequential error using the full history, the prequential error over a sliding window of size 1k and the prequential error using a fading factor of 0.997. Each plot shows the evolution of the different prequential estimates and the point where a change is detected. The learning algorithm was the VFDT-NBAdaptive.

5.4.2 Monitoring Drift with the Ratio of Prequential Error Estimates

A common approach to detect changes consists of using two sliding windows: a short window containing the most recent information and a large window, used as reference, containing a larger set of data including the data in the short window (Bach and Maloof, 2008; Nishida and Yamauchi, 2007). The rationale behind this approach is that the short window is more reactive while the large window is more conservative. When a change occurs, statistics computed in the short window will capture the event faster than using the statistics in the larger window. Similarly, using fading factors, a smooth forgetting mechanism, a smaller fading factor will detect drifts earlier than larger ones. Based on this assumption, a new online approach to detect concept drift is proposed. It is proposed to perform the PH test with the ratio between two error estimates: a long term error estimate (using a large window or a fading factor close to one) and a short term error estimate (using a short window or a fading factor smaller than the first one). If the short term error estimator is significantly greater than the long term error estimator, a drift alarm is signaled.

For fading factors, α_1 and α_2 were considered (with $0 \ll \alpha_2 < \alpha_1 < 1$) and the fading prequential error estimate was computed for both: $P_{\alpha_1}(i)$ and $P_{\alpha_2}(i)$ at observation i. The ratio between them is described as: $R_\alpha(i) = P_{\alpha_2}(i)/P_{\alpha_1}(i)$. The PH test monitors the evolution of $R_\alpha(i)$ and signals a drift when a significant increase of this variable is

observed. The pseudocode is presented in Appendix B.4.

For the approach with sliding windows, the procedure is similar. Using two sliding windows of different sizes w_1 and w_2 (with $w_2 < w_1$), the sliding prequential error estimate is computed for both sliding windows: $P_{w_1}(i)$ and $P_{w_2}(i)$, at observation i. The ratio between the two sliding estimates is computed as: $R_w(i) = P_{w_2}(i)/P_{w_1}(i)$ and the PH test monitors the evolution of this ratio.

In these experiments the same streams, learning algorithms and parameters of the PH test were used as in the previous section. In this study, the values of the first fading factor and the length of the larger window were varied to compare results. Table 5.3 presents the detection delay time for the experiments in the context described above. The rate of forgetting can be controlled using different fading factors or different windows lengths. With respect to the ratio between different fading factors, the value of the second fading factor was set to 0.9970 and the value of the first one varied from 0.9994 to 0.9990. For the ratio between different sliding windows, the length of the second window was set to 1k and the length of the first one varied from 5k to 3k. Greater differences between the fading factors values (or the length of sliding windows) will reinforce the weight of most recent data, enhancing the capacity to forget old data and leading to earlier detections.

Figure 5.6 illustrates the delay time in detecting drifts of both methods. Fading factors α_1 = 0.9994 and α_2= 0.9970 were used and sliding windows of size 1k and 5k were used. As stated in the previous sections, the fading factors, besides consuming less memory, have advantage over sliding windows allowing fast concept drift detections. It is also possible to note that an increase in the length of the larger window and an increase in the first fading factor produce similar results in the detection delay time: the ratio of error rates computed with a fading factor close to one presents smaller delay times in drift detection, similar to the ratio of error rates computed over a window with a larger length (larger windows present behavior comparable to higher fading factors).

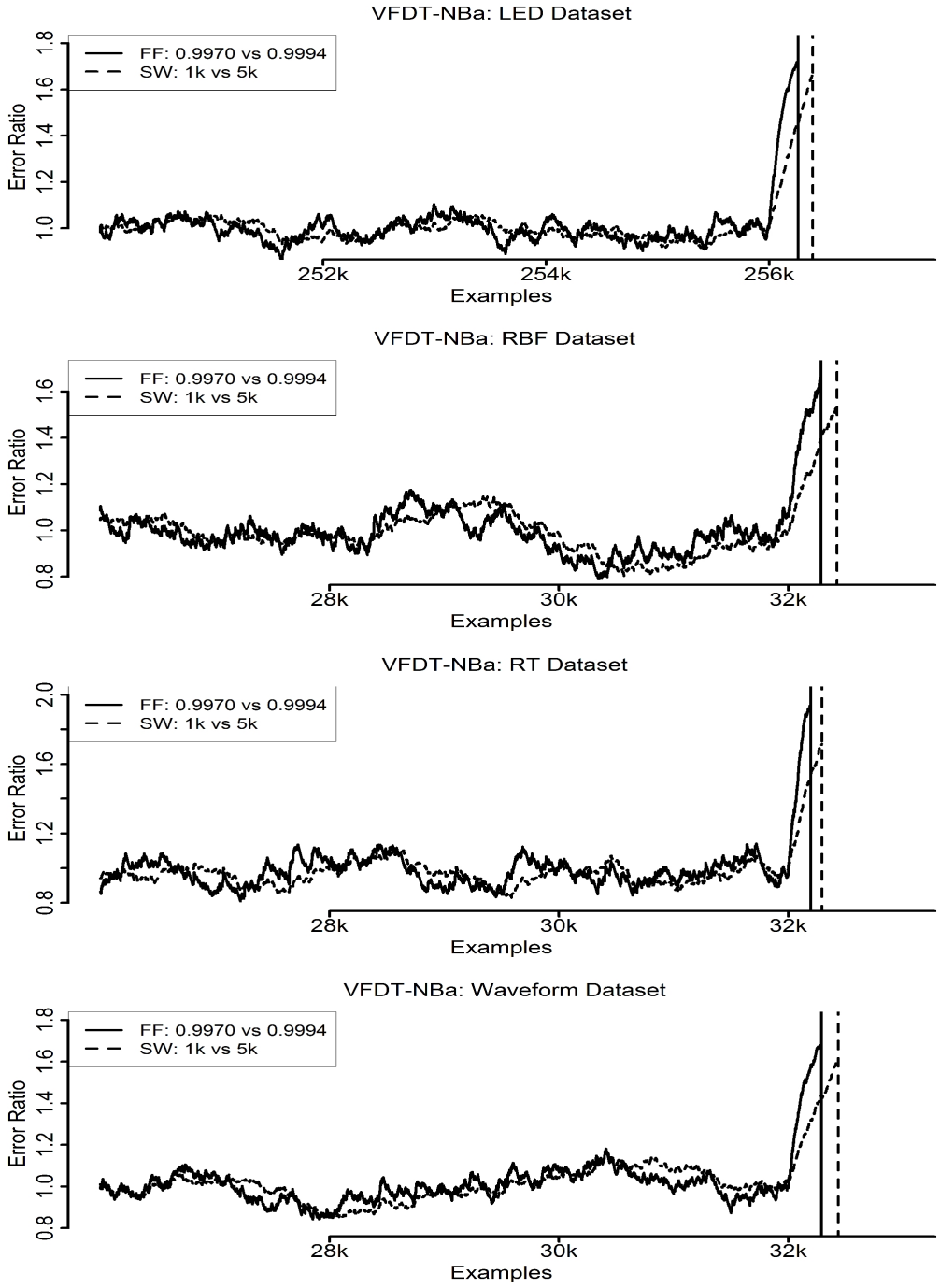

Figure 5.6: The evolution of the ratio of error rate estimates and the delay times in drift detection. The learning algorithm is VFDT-NBAdaptive. Each figure corresponds to a data stream and plots the ratio of error estimates using two different fading factors and two different sliding windows. The vertical lines indicate the point where change was detected.

The order of magnitude in detection delay time of the results presented in Table 5.2 is

$\mu \pm \sigma$	LED		RBF		RT		Waveform	
	MC	NBa	MC	NBa	MC	NBa	MC	NBa
Fading factors								
α_1=*0.9990*	- (10;0)	349±11	787±190	384±38	325±39	262±27	780±171	457±161
α_1=*0.9993*	969±255 (4;0)	281±8	563±100	309±29	261±33	211±25	541±65	348±89
α_1=*0.9994*	856±186 (3;0)	264±7	522±92	291±28	244±32	198±24	496±57 (2;0)	324±77 (1;0)
Sliding windows								
w_1=*3000*	1254±355 (1;0)	473±27	779±94	489±48	429±93 (0;1)	350±51 (0;1)	762±62	551±102
w_1=*4000*	1089±261	423±26	710±100 (0;1)	465±41 (0;1)	397±79 (0;2)	312±52 (0;1)	673±57	501±102
w_1=*5000*	1005±216	397±25	679±98 (0;2)	440±45 (0;2)	384±67 (0;3)	299±37 (0;3)	662±60 (0;5)	468±93

Table 5.3: Detection delay time, average and standard deviation, using PH test monitoring the ratio of error estimates. The learning algorithms are VFDT-MC (MC) and VFDT NBAdapt (NBa). The results report the average and standard deviation of 10 runs. In parenthesis is the number of runs, if any, where PH test misses the detection or signals a false alarm: they are in the form (Miss; False Alarm). With respect to the ratio using different fading factors, the value of the second fading factor was set to 0.9970 and the value of the first one varied from 0.9994 to 0.9990. For the ratio using different sliding windows, the length of the second window was set to 1k and the length of the first varied from 5k to 3k.

thousands of examples, while in Table 5.3 is hundreds of examples. Nevertheless, while monitoring the ratio of error estimates allow much faster detection, it is riskier. False alarms and missed detections are observed, mostly with VFDT-MC. Once more, in these experiments drift detection based on the ratio of fading error estimates is somewhat faster than with sliding windows.

5.5 Conclusions & Research Question

This chapter embraces the main problem with evaluation methods when learning from time-changing data streams: the most appropriate way of monitoring the evolution of the learning process.

Throughout this chapter the following answers to research question 3 were provided:

1. *How should the performance of stream learning algorithms be evaluated?*
   Sliding prequential and fading prequential error estimates are two feasible approaches to assess the performance of stream learning algorithms in the presence of evolving data.

2. *Can forgetting mechanisms provide reliable error estimates?*
   For consistent learning algorithms and examples generated by a stationary distribution, the prequential error estimated over a sliding window converge to the Bayes

error. Moreover, the holdout estimator and the prequential error also converge to the Bayes error, based on the same assumptions.

3. *How can the performance of learning algorithms in non-static environments be compared?*
   The McNemar test was made suitable for comparing stream learning algorithms by applying sliding windows or fading factors strategies.

4. *How can forgetting mechanisms be extended to cope with concept drift problems?*
   For concept drift detection purposes, monitoring drift using forgetting prequential error estimates or the ratio of forgetting prequential error estimates are advantageous approaches. Moreover, evaluating the detection delay time, the latter approach outperforms the former.

Overall, the fading factors strategies are worthwhile because they are memoryless, they do no require recent statistics to be stored in memory as in the case of the sliding windows approaches.

- 6 -

Application in a Clinical Environment

"I have been impressed with the urgency of doing.
Knowing is not enough; we must apply.
Being willing is not enough; we must do."
Leonardo da Vinci (1452 - 1519)

This chapter is devoted to research question 4:

How can a change detection method contribute to a decision support system based on Depth of Anesthesia (DoA) signals?

This research question is answered by proposing the application of a real-time algorithm to automatically detect changes in depth of anesthesia signals (Sebastião et al., 2013).

The method that was developed for change detection in DoA signals constitutes an enhancement of the Page-Hinkley Test (PHT) with a forgetting mechanism (PHT-FM): the samples are weighted according to their age so that more importance is given to recent samples. This enables the detection of the changes with less delay time than if no forgetting factor were used.

The performance of the PHT-FM was evaluated in a bi-fold approach. First, the algorithm was run offline in DoA signals previously collected during general anesthesia, allowing the adjustment of the forgetting mechanism. Second, the PHT-FM was embedded in a real-time software and its performance was validated online in operating room. This was done by asking the clinician that was present in the operating room to classify, in situ, the changes as true positives (TP), false positives (FP) or false negatives (FN).

This chapter is organized as follows. The first section introduces the problem of detecting

changes in Depth of Anesthesia (DoA) signals. Then in Section 6.1.1 the challenges faced while detecting changes in BISpectral index (BIS) signals are presented. Section 6.2 presents the methods used in clinical data collection and the classification procedure of the detected changes. The PHT-FM is advanced in Section 6.3 and Section 6.4 presents the results and discussion of the changes detected using this test. Section 6.5 addresses the limitations of the proposed decision support system. The last section presents the conclusions on research question 4 and further developments on the proposed decision support system.

6.1 Change Detection in Depth of Anesthesia (DoA) Signals

The automatic detection of changes in physiological signals has been a topic of research for some years now (Ansermino et al., 2009; Melek et al., 2005; Yang et al., 2006).

The majority of the indices for assess the DoA, e.g., the Index of Consciousness (IoC) (Jensen et al., 2008), Auditory Evoked Potentials (AEP) (Struys et al., 2002), Spectral Entropy (SE) (Viertiö-Oja et al., 2004) and BISpectral index (BIS) (Gan et al., 1997; Glass et al., 1997), range between a value close to 100, corresponding to fully awake state, and 0, corresponding to electrocortical activity suppression during a fully asleep and unconscious anesthetized state. In general anesthesia, values between 40 and 50 show to be ideal to perform surgical procedures (Luginbühl and Schnider, 2002; Rampil, 1998). However, during the course of surgery, this target range may change depending on the surgical protocols that are being used or on the overall state of the patient. A trained clinician judges the current observations and interprets them with sensitivity to the context and in comparison with previous observations to provide a warning of potential deterioration in the anesthetic state or to change the target values of the monitored physiological signals. Naturally, the thresholds for the clinician to trigger an alarm are dynamically dependent on all the available information and on the clinical environment. The automatic and correct identification of a change that needs to be alarmed can be considered a detection and a decision problem.

Interpreting the BIS signal behavior is of utmost importance to infer the overall anesthetic state of the patient. Surprisingly, no published work exists to date prior to this thesis, to address the problem of solving changes in the BIS signal to infer the correct amount of drugs to be administered. Nevertheless, this is a challenging problem to be solved since it is of paramount clinical relevance for the *a priori* information about the signal

trend. Nowadays the clinicians select the several drug delivery rates based on their clinical experience and on some *a priori* information about surgery procedures.

Therefore, the detection of changes in DoA signals is of paramount importance in the adaptation of the drug doses needed to achieve an optimal DoA level while avoiding undesirable side-effects (Mashour et al., 2009; Selbst, 2000). Changes in DoA signals may occur due either to intrinsic or extrinsic factors. Most of the DoA indices, namely the BIS, report the level of hypnosis of patients (related to unconsciousness), and analgesia (lack of pain) (Minto et al., 2000), which are the intrinsic factors. Intubation, incisions or other painful stimuli are the extrinsic factors.

Even though the DoA is the most widely index used in clinical practice, there still exists some controversy around the benefits of the use of the BIS in extensive clinical practice (Monk and Weldon, 2011). The reason for the use of BIS signals to assess the performance of the proposed change detection algorithm is bi-fold. First, BIS is the DoA index used in the operating rooms the author has collaborated with. Second, insights gained from this assessment are useful to extend this change detection algorithm to other DoA indices. In fact, it turns out, from clinical experience and anesthetic procedures, that the developed methodology can be easily adapted to other physiological signals used to measure the DoA.

6.1.1 Challenges Faced while Detecting Changes in BIS signals

In standard clinical practice, when the behavior of the BIS signal changes, the clinician manually adjusts the drug doses to control the overall anesthetic state of the patient. A quick reaction is crucial since a delay in this adjustment may compromise the well-being and general condition of the patient (Mashour et al., 2009), possibly leading to some undesirable side-effects (such as awareness experience during surgery and post-operative nausea, vomiting and muscle aches).

Figure 6.1 illustrates this scenario in a operating room. Apart from the change detector component, this figure shows the standard clinical procedure: under the supervision of the clinician, the drugs are delivered to the patient through the delivery devices (the amount of drugs that need to be given to the patient is decided by the clinician and manually established). Thereafter, the raw EEG data is collected through a sensor placed on the forehead of patients. The Aspect Medical - Covidien system processes the BIS information and computes a number between 0 and 100. This value is displayed along with the time on a monitor, providing an indication of the level of unconsciousness and sedation of the patient. Taking into account the BIS values presented, the clinician decides on the adjustment of

the drug doses. Accomplishing this standard procedure with a change detector component, the clinician is automatically alerted to changes in the BIS signal and then decides on the drugs adjustments.

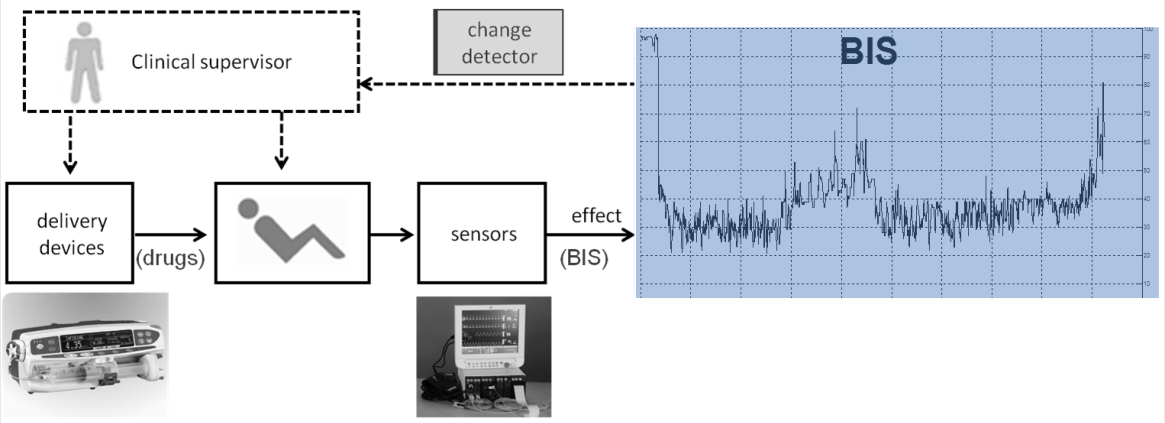

Figure 6.1: The change detector in the operating room setting.

The detection of initial phases of smooth changes is as challenging for the clinicians as it is for change detection algorithms. The enhancement of the Page-Hinkley test (PHT) with a forgetting mechanism aims at overcoming this difficulty. By giving more importance to recent samples, the initial phase of changes will be reinforced and most easily detected.

On the other hand, sudden changes are easily observed and detected. Hence, they do not pose great difficulties to change detection algorithms. However, these changes are the most critical ones because the behavior of the DoA signal changes abruptly and quickly. In these situations, it is of utmost importance to bring the DoA signal back to the desired range or level after the occurrence of the change. This fact reinforces the advantage of using an automatic change detection algorithm to give the clinician advanced warning so he can act even more promptly.

6.2 Methods

In this section the software for data communication and the procedures for the databases collection are presented. The classification of the detected changes by the clinician is also explained.

6.2.1 Software for Data Communication

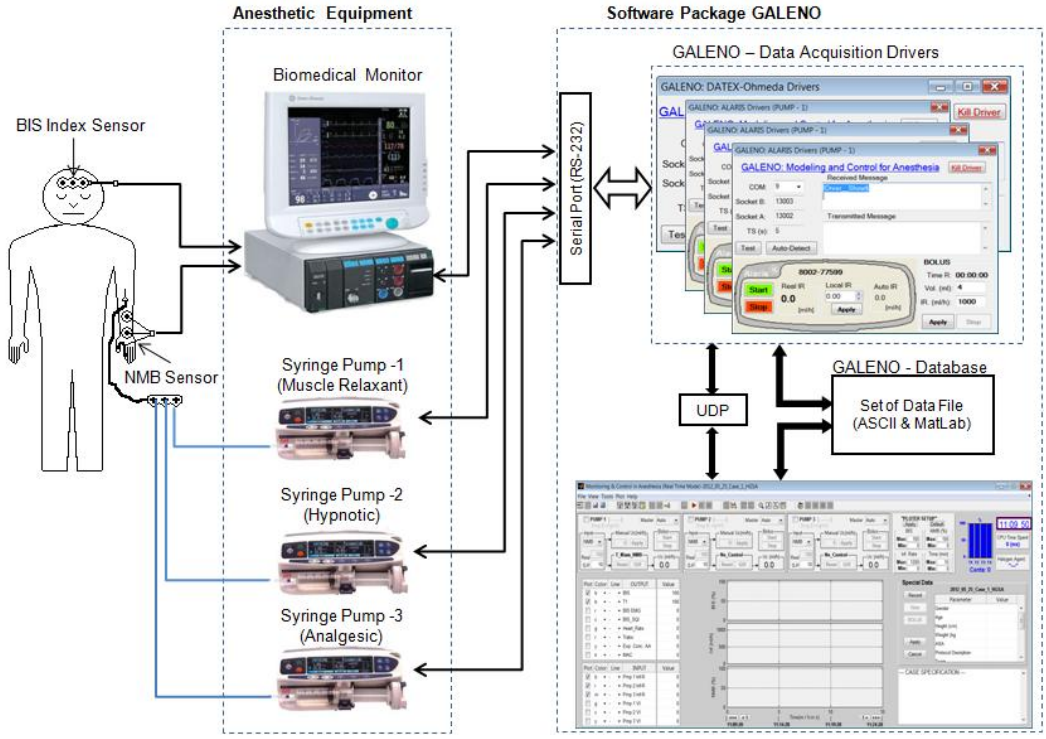

Figure 6.2: Software for data communication (Paz, 2013).

The software for data communication was developed by Paz (2013), in the scope of the project PTDC/SAU-BEB/103667/2008 GALENO - Modeling and Control for Personalized Drug Administration. This software was conceived to interact with two types of medical equipments: DATEX-Ohmeda S/5 monitor and ALARIS syringe pumps. Figure 6.2 illustrates the communication of data between different equipments. The BIS index sensor (as well as the NMB sensor) collects the raw data that is processed by the DATEX-OHMEDA system and displayed in a biomedical monitor. Thereafter, this data, and the drugs information collected by the ALARIS syringe pumps, is communicated to the GALENO platform, that works as the control system. This control system is implemented in MATLAB (MATLAB®& Simulink®, 2007) and performs all the data processing, graphical treatment and supports the implementation of the change detection test and of control algorithms with model simulation for control algorithm testing. Moreover it also allows the introduction of new modules, procedures or functions to increase the system performance. Figure 6.3 shows the MATLAB interface of the control system.

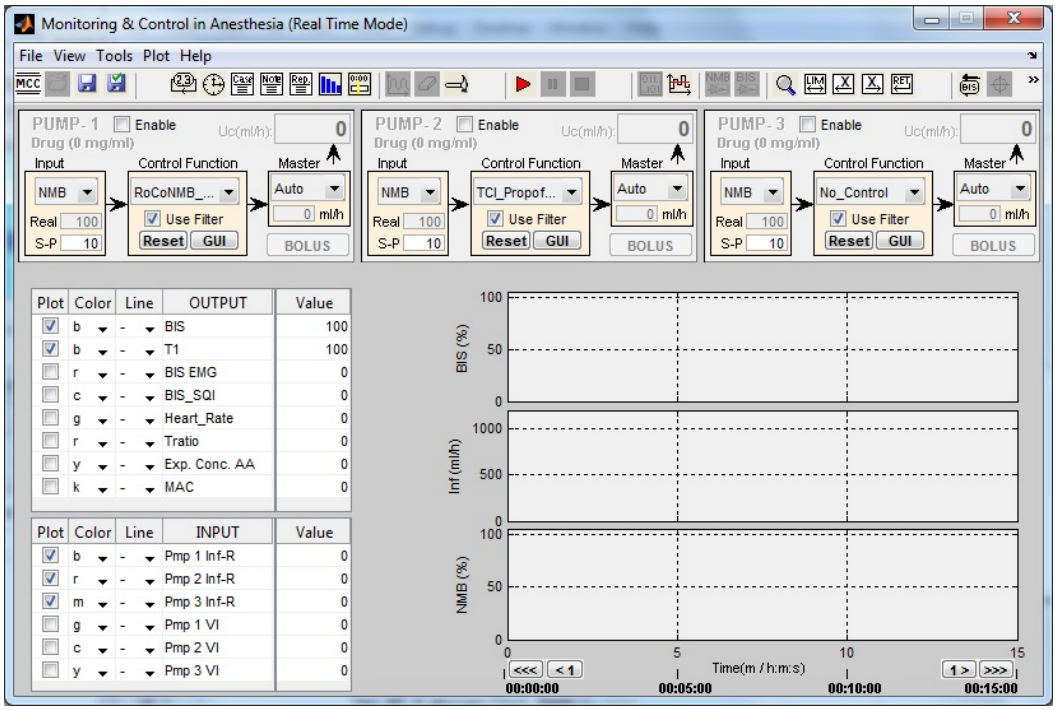

Figure 6.3: MATLAB interface of the GALENO platform.

6.2.2 Clinical Data Collection

In clinical practice, hypnotics and analgesics are the drugs that mainly influence the DoA. In the real cases collected, the hypnotic propofol and analgesic remifentanil were intravenously administrated to induce and maintain the DoA of patients, which was manually controlled by the clinician who changed the drug doses according to clinical requirements, using as a reference the vital signals of patients, e.g., the blood pressure and the heart rate, and the BIS values. The BIS, hemodynamic parameters and drug rates were recorded with a frequency of $1/5$ s^{-1}.

6.2.2.1 Offline Database

For the offline evaluation, data previously collected from 22 patients undergoing abdominal surgery was used. The patients were 60 ± 15 years old, 76.8 ± 17.7 kg and 13 female, and the surgeries lasted on average 144 ± 74 minutes.

The PHT-FM was run on each case of the database and the changes were classified afterward. The changes that were followed by a modification in the rate of propofol and/or

remifentanil were classified as TP. Similarly, if no modification in drugs rates occurred after the detection, the change was, in general, considered to be a FP. However, it should also be noted that during the course of surgery, when informed by the surgeons of the possibly painful procedures that are about to be performed, the clinician often avoids a possible increase in the DoA index value by anticipating the administration of a higher dose of the hypnotic and/or analgesic drugs. Some of the detected changes might therefore be the consequence of an increase in the administered drugs. Due to this, the changes that were detected by the PHT-FM were also considered TP if some modification in the propofol and/or remifentanil administered rates occurred right before the detection of a change.

6.2.2.2 Online Database

Due to the encouraging results that were obtained with the offline evaluation, the PHT-FM was implemented in the GALENO platform and run online in operating room, during 78 general anesthesia episodes, for different types of surgeries. The patients were 56 ± 14 years old, 71.8 ± 16.4 kg and 70 female, and the surgeries had an average duration of 123 ± 66 minutes.

The clinician was advised every time a change was detected by the PHT-FM and asked to classify it as TP or FP. Whenever the clinician considers that there is a true change in the BIS signal that was not detected by the PHT-FM, the clinician classifies it as a FN. The major advantage of this online assessment is that this classification is done in situ, and, consequently, based on the current observations and actions performed at the time when the change is detected. In general, cases where the rate of propofol and/or remifentanil was modified after the change was detected were classified as TP. Similarly, in most cases, if no action by the anesthesiologist was taken after the detection of a change, it was considered to be a FP. Exceptions were when, for example, the patient was subject to painful stimuli of limited duration, caused by a known source of stimulation, such as small incisions or patient repositioning. In such cases, when advised that a change had accurred, usually the clinician did not react by increasing the rate of propofol and/or remifentanil, because the DoA index value would decrease to its previous value as soon as the stimulus ended. Those changes were, however, classified as TP.

6.2.3 Classification of the Changes Detected by the PHT-FM

Both for the offline and the online databases, the changes detected by the proposed change detection algorithm were classified by a single clinician as TP or FP and the missed

changes by FN. The number of TN was not assessed because the TN are all the other data observations in the signal that were not classified as TP, FP or FN. For the online database, the changes that were detected but not classified by the clinician are marked as "nC". In most cases, these detections were not classified because the clinician was not available at the time the change was detected.

6.3 The Page-Hinkley Test with a Forgetting Mechanism

In order to detect changes in the BIS signals, the PHT was selected among other change detection algorithms, namely instead of the CWM, for the following reasons:

- The duration of the surgeries is expected to be brief. This implies that the collected BIS signals have small lengths not requiring synopses structures, avoiding the construction of histograms and saving time.
- The stable phases of the BIS signals are supposed to be short, which will not allow to provide a well supported representation of the reference window, which could compromise the success of comparison with the current window.
- The PHT is easy to implement and the time required to evaluate the samples is short.
- The change detection in the signals must be performed in the flow and the low computational complexity of the PHT makes it appropriate for use in this context.

With the intention of reducing the delay time of the changes detected and since in swift and evolving environments "old" data is usually less important than recent, this method was enhanced with a forgetting mechanism (PHT-FM). A fixed fading factor was shown to be suitable to forget outdated data in streaming scenarios. However, a time series (as BIS signals) is relatively small in size with respect to a data stream. Therefore, when applied to a time series, a fading factor must approximate a window with minimal length. Within this setting, it is worthwhile assigning an adaptive fading factor instead of a fixed one.

In this decay strategy, the samples are weighted according to their age so the PHT will focus more on recent samples, detecting the changes with low delay time. The resulting tests are the following:

For increase cases:	For decrease cases:
$U_0 = 0$	$L_0 = 0$
$U_T = \frac{T-1}{T} U_{T-1} + (x_T - \bar{x}_T - \delta)$	$L_T = \frac{T-1}{T} L_{T-1} + (x_T - \bar{x}_T + \delta)$
$m_T = min(U_t, t = 1 \dots T)$	$M_T = max(L_t, t = 1 \dots T)$
$PH_U = U_T - m_T$	$PH_L = M_T - L_T$

In these equations, the forgetting mechanism is the weight of the variables U_{T-1} and L_{T-1} in the update process. Since the ratio $\frac{T-1}{T}$ increases with time, the recent samples have more importance in the update process than the older ones. With this forgetting mechanism, while assigning more importance to recent observations, the algorithm will be able to detect both abrupt (sudden) and gradual (slow) changes earlier.

At every observation, the two PH statistics (PH_U and PH_L) are monitored and a change is reported whenever one of them rises above a given threshold λ. When a change is detected, all the variables and index of samples are cleaned and a new test is initialized.

The parameter δ is highly dependent on the characteristics of the signal under study. The value of this parameter is chosen to avoid false detections due to noise, taking into account the magnitude of changes that are allowed and should not trigger an alarm. The PHT relies on the difference between the observed value and its current average. Whenever the referred difference increases continuously, eventually exceeding the user predefined threshold (λ), the algorithm detects a change in the recorded BIS signal. The change threshold parameter (λ) is chosen considering a trade-off between admissible false alarm rates and detection delay times.

The pseudocode for the PHT-FM is presented in Appendix B.5.

Figure 6.4 illustrates how PHT-FM works. The upper plot shows the initial phase of a real BIS signal. As can be observed, two changes occur around minutes 14 and 17 due to a decrease in the signal. The bottom figure represents the evolution of the statistical test PH_L and the detection threshold (λ). The PHT-FM statistical test captures both the decreases presented at this stage of the signal. The λ parameter should guarantee that the algorithm, while being resilient to false alarms, can detect and react to changes as soon as they occur, reducing the detection delay time. By controlling this detection threshold parameter, a trade-off between the false positive alarms and the missed detections is established. Regarding both parameters (δ and λ), to the best of author's knowledge, it is not possible to automatically derive them through the characteristics of the signal under

study.

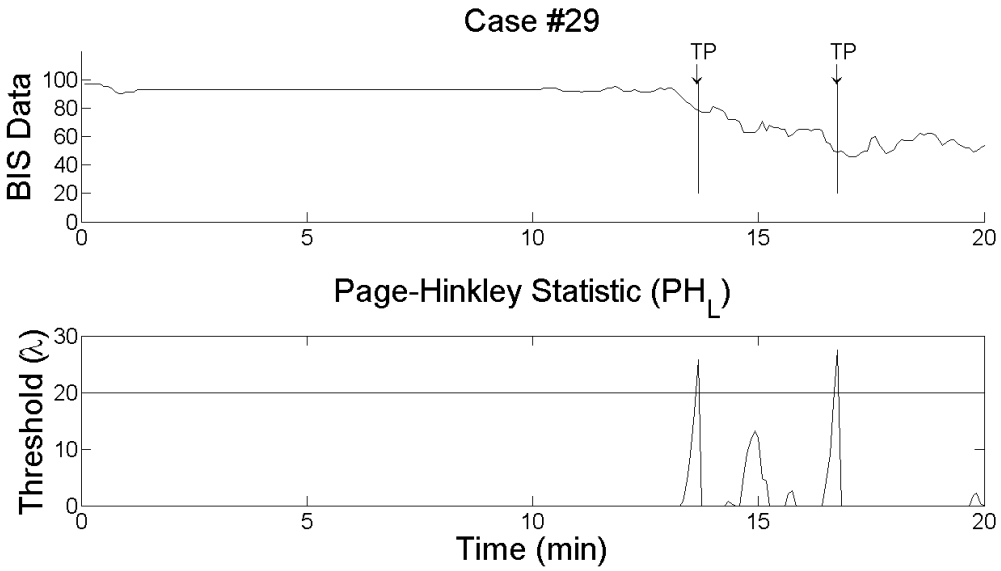

Figure 6.4: The upper figure shows the initial phase of a real BIS signal. The bottom figure represents the evolution of the PHT-FM statistic and the detection threshold λ.

6.3.1 Forgetting Mechanism

A comparative analysis was performed on the offline database to assess the effect of using the forgetting mechanism. Figure 6.5 shows the averaged delay time (in seconds) between accordant detections obtained by the original PHT and by the PHT enhanced with a forgetting mechanism (PHT-FM), for all the cases in the database. When using the PHT-FM, in 50% of the cases, the clinician is warned more than half a minute of earlier than when using the PHT without a forgetting mechanism. This is the main advantage of the PHT-FM, since reducing the delay time in detections gives the clinician more time to decide based on that information. The result of this assessment supports the use of the proposed forgetting mechanism.

6.3.2 Algorithm Input Parameters

To adjust the algorithm input parameters, an analysis was previously conducted on 106 BIS signals (Gambús et al., 2006, 2011) collected from patients undergoing similar anesthetic procedures to the ones described in Section 6.2.2. From that, the parameters λ and δ were set to 20 and 10, respectively. The admissible false alarm rate and the magnitude of

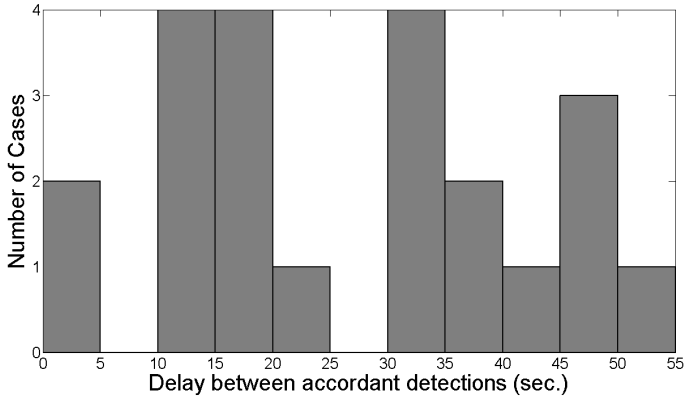

Figure 6.5: Average delay time (in seconds) between accordant detections obtained with the original PHT and with the enhanced PHT-FM, for all cases in the offline database.

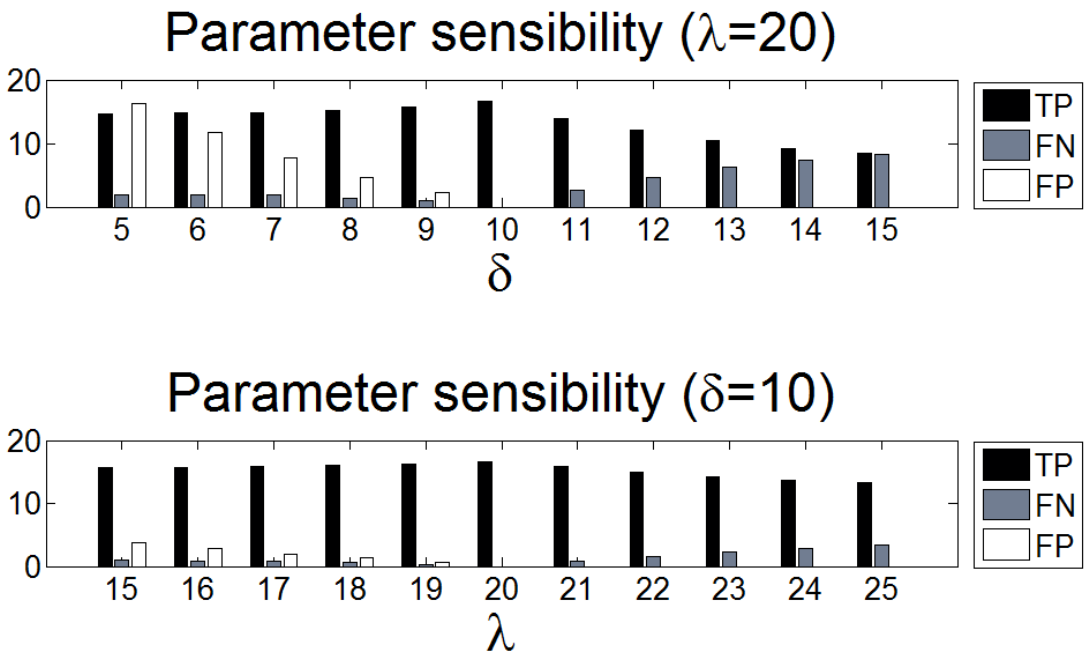

Figure 6.6: Parameters sensitivity.

the permitted changes were taken into account with regard to the intrinsic characteristics of the signal under study. The sensitivity of both parameters was also assessed by fixing one of the parameters and varying the other. The results obtained are shown in Figure 6.6. As is clear from the upper plot of Figure 6.6, where λ was equal to 20, values of δ below 10 give rise to an increasing number of FP. For values of δ above 10, the number of FN increases. The same behavior is found when δ was set to 10, and λ was made to vary between 15 and 25 (bottom plot of Figure 6.6).

6.4 Results and Discussion

The classifications by the clinician of the changes detected by the proposed PHT-FM are considered as ground truth. Naturally, they are highly dependent on the evaluation of a single clinician and are not 100% accurate. Nevertheless, these validations allow computing quality metrics of the PHT-FM performance, such as Precision and Recall.

6.4.1 Offline Evaluation

The main purpose of this validation is the adjustment of the forgetting mechanism. Since the changes that occur in the BIS signal were not simulated, but rather the consequence of changes in live operating settings, it was difficult to ascertain the exact times when the changes in the BIS signal occurred. Hence, a detection delay time evaluation could not be performed. A preliminary evaluation has however been performed scoring the offline algorithm detections.

Regarding the quality of measurements, for all cases in the database a precision of 0.87 and a recall of 0.98 were obtained. Table 6.1 shows the confusion matrix obtained for the offline database. In spite of the high number of false alarms, this fact does not represent a major concern because the clinician might be advised and then decide on an action based upon his expertise and on the vital signals of patients. Another important result of this evaluation is the low number of missed detections, which is evidence that the PHT-FM was able to detect almost all the changes that occurred in the BIS signals that induced the clinician to adjust the drug doses.

It should also be pointed out that most of the detections were classified as TP. These results supported the online implementation and evaluation of this algorithm, whose results are presented in Section 6.4.2.

Table 6.1: Confusion matrix obtained for the offline database.

		Real		Total
		Change	No Change	Total
Detected	Change	266	40	306
	No Change	4	X	4
Total		270	40	310

6.4.2 Online Evaluation

Figure 6.7 shows records of three patients from the online database. For each case, the upper plot shows the BIS signal and the changes detected by the PHT-FM (indicated by a vertical line and arrows pointing upwards if an increase in the BIS signal was detected and pointing downwards if a decrease in the BIS was detected).

As shown in Figure 6.7 (a), the change around minute 6, which is the consequence of the administration of the initial *bolus* of propofol was detected by the algorithm, as expected, and evaluated as TP by the clinician. The PHT-FM consistently detected the decreases of the BIS signal as a result of this propofol *bolus*, as can also be observed later at minute 10, both validated as TP. Although a TP validation by the clinician, around minute 50 the algorithm detected two ascendant changes that were neither the consequence of nor were they followed by any clinical action. Around minute 65, the clinician validated as FP an algorithm detection. These situations intend to illustrate the difficulties that the problem under study poses to the development of change detection algorithms, namely the false positive detections due to noise present in the BIS signals. Later, around minute 70, a detection of an increase in the BIS, marked as TP, followed by the administration of a propofol *bolus* by the clinician is noticeable. As expected, after this *bolus* the BIS decreased which was detected by the PHT-FM (despite the first detected change not having been classified by the clinician, the second one was validated as TP and both were clearly a result of the *bolus* administration). This is one example where the online use of this algorithm may be advantageous: advised by the algorithm of this increase the clinician could had acted more promptly. The last change detections, classified as TP, were the result of the end of the administration of the drugs.

Figure 6.7 (b) presents another clinical case. After the administration of the initial *bolus* of propofol, the PHT-FM detected two decreases and later (around minute 25) another one as the result of the accommodation of the BIS to this dose (all classified by the clinician as TP). After a stable period with values within the range of 40 to 60, the BIS signal reveals an increase which was detected by the algorithm. This change, evaluated as TP, contributed to an administration of a propofol *bolus* by the clinician. Resulting from this administration of propofol, a decrease detection evaluated as TP also occurs. From minute 40 to minute 60 the BIS signal remains stable around a mean value of 45. Around minute 60, the algorithm detected another increase, classified as TP, and leading to an administration of a propofol *bolus*. However, this dose was not enough to reduce the BIS signal which remained with an increasing behavior that was identified by the PHT-FM around minute 65. As a consequence, to avoid another BIS increase, the clinician

administrated another propofol *bolus*. After these administrations, the BIS recovered to the clinical reference range which was also detected by the PHT-FM (around minute 75). Later, two false positive detections can be observed (the algorithm alarmed these changes without any evident existence of a change). It should be noted that the noisy level of these signals poses difficulties for this algorithm and often noise can be confused with initial phases of a change, alarming a change when the signal remains stable and raising the rate of false positives. The last detections of the algorithm (validated as TP) were a consequence of the end of drugs administration due to the end of the surgery.

Figure 6.7 (c) shows a clinical case where the PHT-FM missed a change, identified as FN (around minute 60). The first three detected decreases, classified as TP by the clinician, were the result of the BIS adaptation after the initial dose of propofol. Around minute 25, the PHT-FM detected an increase in the BIS. The administration of a propofol *bolus* followed this increase in order to maintain the BIS in the predefined target window. A rise in the BIS around minute 40 was also detected by the algorithm and followed by a decrease in the signal despite no clinical action. These two detections were the result of the incision and did not report a relevant variation in the BIS behavior and as a consequence were classified as FP. Around minute 70, a TP detection led to a decrease in the propofol administration. Approaching the end of the surgery, the administration of the drugs was tuned off and the algorithm, as expectable, detected two increases in the BIS signal (both validated as TP).

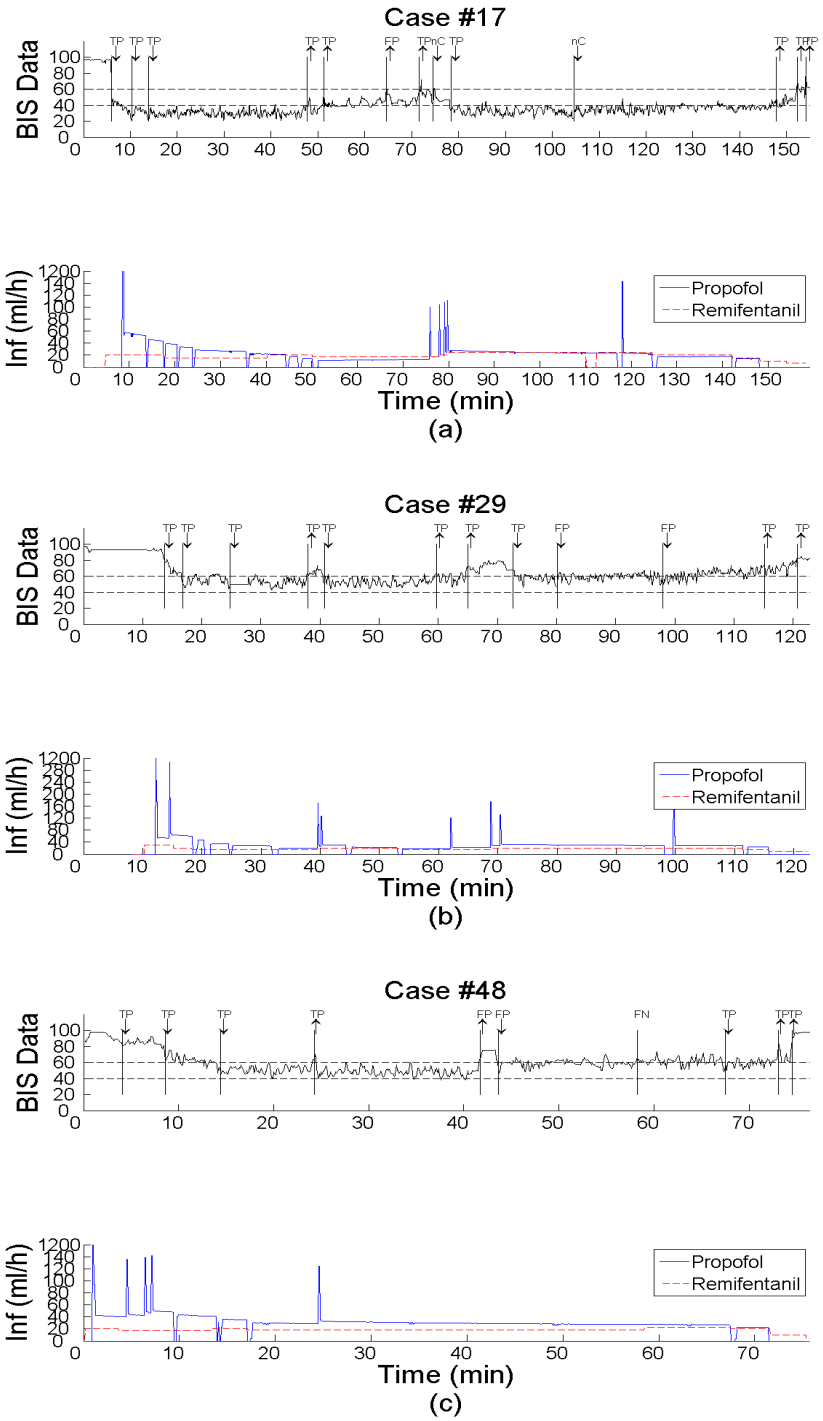

Figure 6.7: Example of the detected changes in a BIS signal of three clinical cases in the online database. The upper plots show the BIS signal, the detected changes (indicated by vertical lines and arrows) and the validation by the clinician. The bottom plots show the propofol and remifentanil dosages (ml/h), respectively.

So too with the offline analysis, the precision and the recall were computed for all cases in the database, obtaining 0.72 and 0.93, respectively. The high accuracy achieved by the proposed PHT-FM supports the argument for its inclusion in a real-time decision support system for routine use in clinical practice. Table 6.2 shows the evaluation metrics corresponding to the online validation by the clinician. It must be stressed that the number of false alarms are not a major concern for this problem. A false alarm will alert the clinician, who decides if an action should be taken, based upon his expertise and on the vital signals of patients. The missed detections might be more problematic. If the PHT-FM misses a change that occurred in the DoA signal, and if the clinician is unable to infer it by looking at the available monitors, the drugs doses will not be adjusted in order to provide more comfort to the patient. Both situations can disturb and/or potentially mislead the clinician but do not cause any immediate wrong action.

Table 6.2: Confusion matrix obtained for the online database.

		Real		Total
		Change	No Change	Total
Detected	Change	660	255	915
	No Change	48	X	48
Total		708	255	963

Since the change detection algorithm is implemented online it is not possible to avoid the presence of noise and sensor faults. Therefore, it should be pointed out, that the validations of the clinician were only taken into account if the quality of the BIS signal was greater or equal to 50%. As well as this, some of the changes detected by the algorithm were not classified by the clinician (and were not considered in these measurements).

6.5 Limitations of the Proposed Decision Support System

The presence of noise and outliers in the collected signals of BIS are the main limitations when incorporating the proposed PHT-FM in a decision support system.

The corruption of the BIS signals by noise is as challenging for the application domain as for the change detection algorithms. For the application domain, noise can confuse the clinician in the observation of the signal behavior, affecting and potentially confusing his judgments. While for the change detection algorithms the noise can be easily confused with an initial phase of a change. The noise in the signals could be addressed by performing some sort of filter strategy before presenting the signal to the change detection algorithm.

However, this was not in the scope of this assignment, but it is a task to which some effort should be devoted in future research.

Figure 6.8 shows a typical BIS signal collected in the operating room, illustrating the level of noise present in the BIS signals. Especially after minute 50, along with decreases and increases in the BIS behavior, the presence of noise is noticeable in the high variation of the BIS signal.

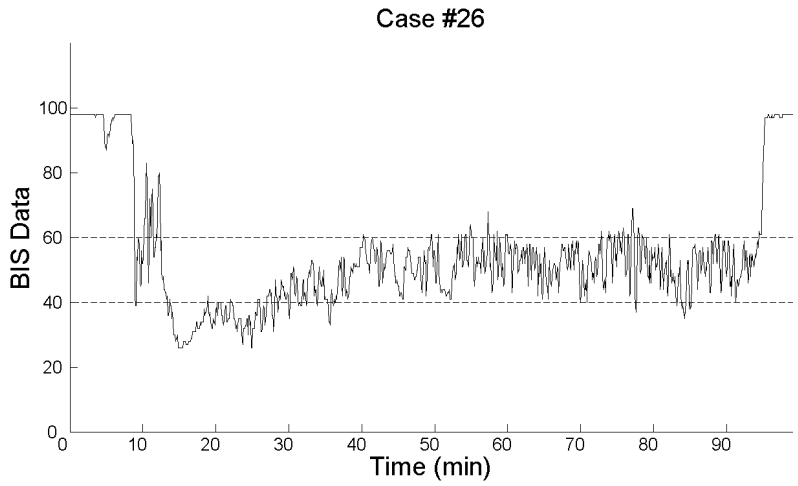

Figure 6.8: Example of a BIS signal with typical noise level, collected in the operating room.

Outliers and sensor faults also pose difficulties to both domains. Sensor faults, which are impossible to avoid, often cause outliers and/or missed data. These constraints were not addressed within this application.

6.6 Conclusions on Research Question and Further Developments

The proposed PHT-FM has been found to be suitable for answering research question 4. The online evaluations obtained so far support the advantage of this change detection algorithm when monitoring DoA signals under general anesthesia procedures.

This contribution has a high impact in clinical practice since the PHT-FM alerts the clinician in advance to changes in the anesthetic state of the patient, allowing actions to be taken earlier. The results support the inclusion of the proposed PHT-FM in a robust and reliable real-time decision support system for routine use in clinical practice. This system would help clinicians to determine and administer the drug doses needed to achieve an optimum degree

of comfort while avoiding undesirable side-effects, leading to increased patient anesthesia satisfaction.

However, it should be noted that the environment of the application and the specific features of the BIS signal, namely the high level of noise present in the measurements, point to the need for further improvements in this detection algorithm. The development of a dedicated online filter to smoother the BIS signals is a task that should be addressed to enhance the detection algorithm results.

Due to the lack of reliable sensors to directly and quantitatively measure the level of analgesia, and considering the controversy around the use of BIS to quantify the DoA of patients, further research on other vital signals such as the electrocardiogram, blood pressure and heart rate should be carried out to improve the decision support system. Results should naturally also be correlated with those obtained using BIS signals, or other DoA signals, since the extension of the PHT-FM to them is straightforward.

- 7 -

Concluding Remarks

"It is good to have an end to journey toward;
but it is the journey that matters, in the end."
Ernest Hemingway (1899 - 1961)

This final chapter provides a summary of the main contributions of this thesis, drawing some conclusions from the research accomplished during this PhD project. An outline of possible future developments is also presented.

7.1 Main Contributions

During the course of this PhD project, the main concern was to focus on the research outcomes: would the results be indelible?

To accomplish such an ambitious and challenging assignment, the research conducted during the PhD project was concerned with the identification of the appropriate methods to address the challenges established and to fulfill the research questions proposed.

The efforts to answer the proposed research questions led to the contributions summarized in the following sections.

7.1.1 Online Equi-width Histograms under Error Constraints

It is impractical to accumulate and archive in memory the massive amount of information produced at a high-speed rate and gathered in the form of transient and infinite data streams. In practice, data is promptly processed and discarded immediately. Therefore,

it is necessary to create compact summaries of data, keeping only a small and finite representation of the received information. Such structures allow the discarded data to be remembered.

Within this context, online equi-width histograms, under mean square error constraints, are proposed to construct compact representations of data: the number of buckets in the histogram is defined with respect to a user-established bound on the mean square error of the histogram.

7.1.2 Online Fading Histograms

When dealing with data streams in evolving environments, in addition to the remembering approach, it is also necessary to forget outdated data: old observations do not describe the current state of nature and therefore are useless.

This issue is dealt with by using fading factors in the construction of histograms, weighting data examples according to their age: recent observations (with high weight) contribute more to the fading histogram than old observations (with low weight).

Such fading histograms, besides allowing discarded data to be remembered, also allow outdated data to be gradually forgotten. Therefore, the data representation provided by these fading histograms is more up-to-date.

7.1.3 Cumulative Windows Model (CWM) for Change Detection

Fading histograms forget data gradually, by assigning different importance to observations. Regarding the detection of changes, after the occurrence of a change, all past observations are forgotten. This is the abrupt forgetting brought on when using a change detection method.

Within the scope of change detection, a windowing scheme to address both distribution and concept changes is proposed. The Cumulative Windows Model (CWM) for detecting changes is based on the online monitoring of data distributions provided by the histograms mentioned before, which are compared using the Kullback-Leibler divergence. This change detection method can be performed using an adaptive evaluation step, allowing to reduce the detection delay time.

7.1.4 Forgetting Error Estimates

Learning from data streams differs in various aspects from learning in batch mode. This is mainly because the learning process must be done sequentially. Therefore, standard techniques to evaluate batch learning algorithms, such as cross-validation and variants, are useless to evaluate the learning of stream algorithms. The evaluation of such learning algorithms must be performed along with the incoming data. Prequential and holdout strategies are suitable for use in this context.

However, in most cases, the learning scenario is not static; rather it evolves with time. Thus, stream learning models must accommodate the evolving data. In this time-changing context, the prequential and holdout strategies become inappropriate. Therefore, as a contribution of this thesis, new criteria for effectively evaluating algorithms when learning from evolving data streams are advanced. These evaluation metrics are based on computing the prequential error estimate using forgetting mechanisms: either sliding windows or fading factors. Moreover, for consistent learners, convergence proof to the Bayes error of the error estimates obtained through the prequential method, through the holdout strategy and over sliding windows are presented.

Regarding the forgetting estimates, the fading factor strategies are useful because they are memoryless, not requiring to store in memory recent statistics, as is the case with the sliding windows approaches.

7.1.5 Decision Support System

The last but not the least contribution of this thesis is presented in a clinical environment: a real-time algorithm for change detection in depth of anesthesia signals of patients undergoing surgery.

The Page-Hinkley Test (PHT) was enhanced with a forgetting mechanism (PHT-FM) in order to detect changes in DoA signals with less detection delay time. The use of an effective change detection method has a high impact in clinical practice, since it alerts the clinician in advance to changes in the anesthetic state of the patient, allowing a more prompt action to be taken.

The results of the online evaluation of the ability of the PHT-FM to detect changes in DoA signals are remarkably encouraging. Therefore, the argument that the PHT-FM should be embedded in a real-time decision support system for routine use in clinical practice is sustained.

7.2 Directions for Further Research

Research is endless: in most cases, the main findings in research lead to new and unsettling questions.

This section presents directions for further developments and the issues raised while working on the main contributions of this thesis.

7.2.1 Data Synopsis

Developments in technology have led to high capacity and low cost smart devices. These are available almost everywhere, producing and collecting, at a high-speed rate, huge amounts of information. Moreover, the flourishing of networks prompts the exchange of information between devices. Therefore, real world problems must be addressed from a multidimensional perspective. Although the broad applicability of histograms relies on their simplicity, efficiency and effectiveness, new construction algorithms are needed to conveniently handle such data.

7.2.2 Evolving Scenarios

The world is far from being static and data collected from real applications is becoming increasingly evolved. Therefore, it is of paramount interest to go beyond the detection of distribution and concept changes. Change analysis must be carefully carried out, developing general groundwork concerned with the description of changes between data distributions or between target concepts.

Moreover, the learning ability of evolving models that deal with evolving data must be improved. Forgetting error estimates must be used to assess performance, allowing fast adaptability in evolving environments.

7.2.3 Personalized Drug Administration System

A real-time decision support system, with the embedded PHT-FM, would help clinicians to determine and administer the drug doses needed to achieve an optimum degree of comfort while avoiding undesirable side-effects, leading to increased patient anesthesia satisfaction.

Moreover, this decision support system should integrate a broader personalized drug administration system for routine use in clinical practice, like general anesthesia, deep sedation, and intensive care units.

This is a huge task and must be embraced by a multidisciplinary research group in the scope of a research project.

- A -

Fading Histograms

This appendix presents some computations on the error of approximating the fading sliding histogram with the fading histogram. Considering the definition of histogram frequencies (3.1), the frequencies of a sliding histogram (with k buckets) constructed over a sliding window of length w and computed at observation i with an exponential fading factor α ($0 \ll \alpha < 1$), can be defined as follow:

$$F_{\alpha,w,j}(i) = \frac{\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)}{\sum_{j=1}^{k} \sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)}, \forall j = 1, \ldots, k, \tag{A.1}$$

To approximate a fading sliding histogram by a fading histogram, the older data than that within the most recent window $W = \{x_l : l = i - w + 1, \ldots, i\}$ must be taken into consideration. Therefore, for each bucket $j = 1, \ldots, k$, the proportion of weight given to old observations (with respect to W) in the computation of the fading histogram is defined as the bucket ballast weight:

$$B_{\alpha,w,j}(i) = \frac{\sum_{l=w}^{i-1} \alpha^{l}}{N_\alpha(i)}, \forall j = 1, \ldots, k, \tag{A.2}$$

where $N_\alpha(i)$ is the fading increment defined as $N_\alpha(i) = \sum_{j=1}^{k} \sum_{l=1}^{i} \alpha^{i-l} C_j(l)$.

As with the old observations, for each bucket $j = 1, \ldots, k$, the proportion of weight given

to observations within the most recent window W is defined by:

$$B'_{\alpha,w,j}(i) = 1 - B_{\alpha,w,j}(i) = \frac{\sum_{l=0}^{w-1} \alpha^l}{N_\alpha(i)}, \forall j = 1, \ldots, k. \tag{A.3}$$

Hence, the error of approximating the fading sliding histogram with the fading histogram, both with k buckets, can be defined as:

$$\Delta_{\alpha,w}(i) = \sum_{j=1}^{k} \Delta_{\alpha,w,j}(i) = \sum_{j=1}^{k} \left\| F_{\alpha,w,j}(i) - F_{\alpha,j}(i) \right\|. \tag{A.4}$$

Theorem A.1. *Let $\varepsilon < 1$ be an admissible ballast weight for the fading histogram. Then, $\Delta_{\alpha,w}(i) \leq 2\varepsilon$.*

Proof. From the respective histogram frequencies definitions comes that the approximation error in each bucket is:

$$\Delta_{\alpha,w,j}(i) = \left\| \frac{\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)}{\sum_{j=1}^{k} \sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)} - \frac{\sum_{l=1}^{i} \alpha^{i-l} C_j(l)}{\sum_{j=1}^{k} \sum_{l=1}^{i} \alpha^{i-l} C_j(l)} \right\|, \forall j = 1, \ldots, k$$

Splitting each of these errors considering the frequencies inside and outside the most recent window of size w:

$$\Delta_{\alpha,w,j}(i) = \left\| \Delta_{\alpha,w,j}(i)^{in} - \Delta_{\alpha,w,j}(i)^{out} \right\|,$$

where

$$\Delta_{\alpha,w,j}(i)^{in} = \frac{\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)}{\sum_{j=1}^{k} \sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)} - \frac{\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)}{\sum_{j=1}^{k} \sum_{l=1}^{i} \alpha^{i-l} C_j(l)},$$

and

$$\Delta_{\alpha,w,j}(i)^{out}(i) = \frac{\sum_{l=1}^{i-w} \alpha^{i-l} C_j(l)}{\sum_{j=1}^{k} \sum_{l=1}^{i} \alpha^{i-l} C_j(l)}.$$

Looking for an upper bound on the error, the worst case scenario is that these two sources

of error do not cancel out, rather adding up their effect:

$$\Delta_{\alpha,w,j}(i) \leq \left\|\Delta_{\alpha,w,j}(i)^{in}\right\| + \left\|\Delta_{\alpha,w,j}(i)^{out}\right\|.$$

Hence

$$\Delta_{\alpha,w,j}(i) \leq \left\| \frac{\left(\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)\right)\left(\sum_{j=1}^{k}\sum_{l=1}^{i-w} \alpha^{i-l} C_j(l)\right) - \left(\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)\right)\left(\sum_{j=1}^{k}\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)\right)}{\left(\sum_{j=1}^{k}\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)\right)\left(\sum_{j=1}^{k}\sum_{l=1}^{i} \alpha^{i-l} C_j(l)\right)} \right\| + \left\| \frac{\sum_{l=1}^{i-w} \alpha^{i-l} C_j(l)}{\sum_{j=1}^{k}\sum_{l=1}^{i} \alpha^{i-l} C_j(l)} \right\| \Leftrightarrow$$

$$\Leftrightarrow \Delta_{\alpha,w,j}(i) \leq \left\| \frac{\left(\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)\right)\left(\sum_{j=1}^{k}\sum_{l=1}^{i-w} \alpha^{i-l} C_j(l)\right)}{\left(\sum_{j=1}^{k}\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)\right)\left(\sum_{j=1}^{k}\sum_{l=1}^{i} \alpha^{i-l} C_j(l)\right)} \right\| + \left\| \frac{\sum_{l=1}^{i-w} \alpha^{i-l} C_j(l)}{\sum_{j=1}^{k}\sum_{l=1}^{i} \alpha^{i-l} C_j(l)} \right\| \Leftrightarrow$$

$$\Leftrightarrow \Delta_{\alpha,w,j}(i) \leq \left\| \frac{\left(\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)\right)\left(\sum_{j=1}^{k}\sum_{l=1}^{i-w} \alpha^{i-l} C_j(l)\right)}{\left(\sum_{j=1}^{k}\sum_{l=i-w+1}^{i} \alpha^{i-l} C_j(l)\right) N_\alpha(i)} \right\| + \left\| \frac{\sum_{l=1}^{i-w} \alpha^{i-l} C_j(l)}{N_\alpha(i)} \right\|$$

The upper bound on the error is given by considering all $C_j(l) = 1$:

$$\Delta_{\alpha,w,j}(i) \leq \left\| \frac{\left(\sum_{l=i-w+1}^{i} \alpha^{i-l}\right)\left(k\sum_{l=1}^{i-w} \alpha^{i-l}\right)}{\left(k\sum_{l=i-w+1}^{i} \alpha^{i-l}\right) N_\alpha(i)} \right\| + \left\| \frac{\sum_{l=1}^{i-w} \alpha^{i-l}}{N_\alpha(i)} \right\| = 2\left\| \frac{\sum_{l=1}^{i-w} \alpha^{i-l}}{N_\alpha(i)} \right\|$$

Then, from bucket ballast weight definition comes that:

$$\Delta_{\alpha,w,j}(i) \leq 2\left\|B_{\alpha,w,j}(i)\right\|$$

Considering in each bucket $j = 1, \ldots, k$ an admissible ballast weight, at most, of ε/k

comes that:

$$\Delta_{\alpha,w}(i) = \sum_{j=1}^{k} \Delta_{\alpha,w,j}(i) \leq \sum_{j=1}^{k} 2\varepsilon/k = 2\varepsilon.$$

□

- B -

Algorithms

In this appendix, the pseudocode from the different change detection methods addressed along the thesis is presented.

B.1 Drift Detection Method

Algorithm 3 Drift Detection Method (DDM).

```
Input: Φ                                                          % online classifier
       {x_t; y_t}^n                                               % stream of examples
  Initialize n ← 0                                                % number of examples
             r ← 0                                                % number of wrong classifications
             p_min ← 0, p_min ← 0
             B ← {}                                               % short-term buffer
  for each example (x_t, y_t) do
      ŷ_t ← Φ(x_t)
      increment n                                                 % number of examples
      if Φ(x_t) ≠ y_t then
          increment r                                             % wrong classification
      end if
      p_t ← r/n                                                   % probability of misclassification
      s_t ← sqrt(p_t(1−p_t)/n)                                    % standard deviation
      if p_t + s_t < p_min + s_min then
          p_min ← p_t; s_min ← s_t                                % update registers
      end if
      if n ≥ 30 then
          if p_t + s_t < p_min + 2s_min then
              Warning? ← FALSE                                    % In-Control
              Update Φ with example {x_t, y_t}
              B ← {}
          else
              if p_t + s_t < p_min + 3s_min then
                  if NOT Warning? then
                      B ← B ∪ {x_t, y_t)}                         % Warning zone
                      Warning? ← TRUE
                  else
                      B ← B ∪ {x_t, y_t)}
                  end if
              else
                  Relearn a new classifier using the examples in the B   % Out-control
                  Warning ← FALSE
                  reinitialize n, r, p_min, s_min, B
              end if
          end if
      end if
  end for
```

B.2 ADaptive WINdowing Method

Algorithm 4 ADaptive WINdowing method (ADWIN2).

Input: **S** % data stream of examples or errors
M % bucket's parameter
Initialize W as an empty list of buckets
WIDTH, VARIANCE and TOTAL

for all x_t in **S do**
SETINPUT (x_t, W)
Output: $\hat{u}_W = TOTAL/WIDTH$ and $ChangeAlarm$
end for

SETINPUT (item **e**, list W)
`InsertElement`(**e**,W)
repeat DELETEELEMENT(W)
until $|\hat{u}_{W_0} - \hat{u}_{W_1}| < \epsilon_{cut}$ holds
for every split of W into $W = W_0 \cdot W_1$

INSERTELEMENT(**e**,W)
create a new bucket **b** with content **e** and capacity 1
$W \leftarrow W \cup$ **b** % add **e** to the head of W
updata WIDTH, VARIANCE and TOTAL
CompressBuckets(W)

DELETEELEMENT(list W)
remove a bucket from the tail of list W
update WIDTH, VARIANCE and TOTAL
$ChangeAlarm \leftarrow TRUE$

COMPRESSBUCKETS(list W)
Transverse the list of buckets in increasing order
do If there are more than M buckets of the same capacity
do Merge buckets
COMPRESSBUCKETS(sublist of W not transversed)

B.3 Page-Hinkley Test

Algorithm 5 Page-Hinkley Test (PHT).

Input: **S** % data stream of examples or errors
δ % parameter for admissible changes
λ % change threshold
Output: $DriftAlarm \in \{TRUE, FALSE\}$
Time of the detected changes: t^*

% Initialize the error estimators
$S(0) \leftarrow 0$;
$U(0) \leftarrow 0$; $m(0) \leftarrow 0$;
$L(0) \leftarrow 0$; $M(0) \leftarrow 0$;

% Update the error estimators
for all x_t in **S do**
$S(t) \leftarrow S(t-1) + x(t)$;
$U(t) \leftarrow U(t-1) + x(t) - \frac{S(t)}{t} - \delta$;
$L(t) \leftarrow L(t-1) + x(t) - \frac{S(t)}{t} + \delta$
$m(t) \leftarrow min(m(t), U(t))$;
$M(t) \leftarrow max(M(t), L(t))$;

% Page Hinkley test
if $PH_U \leftarrow U(t) - m(t) \geq \lambda$ **then**
report a change at time t: $t^* = t$ % increase
$DriftAlarm \leftarrow TRUE$
else
$DriftAlarm \leftarrow FALSE$
end if
if $PH_L \leftarrow M(t) - L(t) \geq \lambda$ **then**
report a change at time t: $t^* = t$ % decrease
$DriftAlarm \leftarrow TRUE$
else
$DriftAlarm \leftarrow FALSE$
end if
end for

B.4 Page-Hinkley Test Based on the Ratio of two Fading Factors

Algorithm 6 Page-Hinkley test based on the ratio of two fading factors.

Input: **S** % data stream of examples or errors
δ % parameter for admissible changes
λ % change threshold
Fading factor α_1 $(0 \ll \alpha_1 \leq 1)$
Fading factor α_2 $(0 \ll \alpha_2 < \alpha_1)$
Output: $DriftAlarm \in \{TRUE, FALSE\}$
Time of the detected changes: t^*

% Initialize the error estimators

$S_{\alpha_1}(0) \leftarrow 0$; $S_{\alpha_2}(0) \leftarrow 0$;
$SR(0) \leftarrow 0$; $m_T(0) \leftarrow 0$; $M_T \leftarrow 1$;

% Update the error estimators

for all x_t in **S do**
$S_{\alpha_1}(i) \leftarrow e_i + \alpha_1 * S_{\alpha_1}(i-1)$
$N_{\alpha_1}(i) \leftarrow 1 + \alpha * N_{\alpha_1}(i-1)$
$M_{\alpha_1} \leftarrow \frac{S_{\alpha_1}(i)}{N_{\alpha_1}(i)}$
$S_{\alpha_2}(i) \leftarrow e_i + \alpha_2 * S_{\alpha_2}(i-1)$
$N_{\alpha_2}(i) \leftarrow 1 + \alpha * N_{\alpha_2}(i-1)$
$M_{\alpha_2} \leftarrow \frac{S_{\alpha_2}(i)}{N_{\alpha_2}(i)}$
$R(i) = log(\frac{M_{\alpha_2}}{M_{\alpha_1}})$

% Page Hinkley test

$SR(i) \leftarrow SR(i-1) + R(i)$
$m_T(i) \leftarrow m_T(i-1) + R(i) - \frac{SR(i)}{i} - \delta$
$M_T \leftarrow min(M_T, m_T(i))$
if $m_T(i) - M_T \geq \lambda$ **then**
report a change at time t: $t^* = t$ % time of detected drift
$DriftAlarm \leftarrow TRUE$
else
$DriftAlarm \leftarrow FALSE$
end if
end for

B.5 Page-Hinkley Test based on Fading Factors (adaptive)

Algorithm 7 Page-Hinkley test based on fading factors (adaptive).

Input: **S** % data stream of examples or errors
δ % parameter for admissible changes
λ % change threshold
Output: $DriftAlarm \in \{TRUE, FALSE\}$
Time of the detected changes: t^*

% Initialize the error estimators
$S(0) \leftarrow 0$;
$U(0) \leftarrow 0$; $m(0) \leftarrow 0$;
$L(0) \leftarrow 0$; $M(0) \leftarrow 0$;

% Update the error estimators
for all x_t in **S do**
$S(t) \leftarrow S(t-1) + x(t)$;
$U(t) \leftarrow \frac{T-1}{T} U(t-1) + x(t) - \frac{S(t)}{t} - \delta$;
$L(t) \leftarrow \frac{T-1}{T} L(t-1) + x(t) - \frac{S(t)}{t} + \delta$
$m(t) \leftarrow min(m, U(t))$;
$M(t) \leftarrow max(M, L(t))$;

% Page Hinkley test
if $PH_U = U(t) - m \geq \lambda$ **then**
report a change at time t: $t^* = t$ % increase
$DriftAlarm \leftarrow TRUE$
else
$DriftAlarm \leftarrow FALSE$
end if
if $PH_L = M - L(t) \geq \lambda$ **then**
report a change at time t: $t^* = t$ % decrease
$DriftAlarm \leftarrow TRUE$
else
$DriftAlarm \leftarrow FALSE$
end if
end for

References

Ansermino, J., Daniels, J., Hewgill, R., Lim, J., Yang, P., Brouse, C., Dumont, G., and Bowering, J. (2009). An evaluation of a novel software tool for detecting changes in physiological monitoring. *Anesth Analg*, 108(3):873–80.

Ayres-de Campos, D., Bernardes, J., Garrido, A., Marques-de Sá, J., and Pereira-Leite, L. (2000). Sisporto 2.0: a program for automated analysis of cardiotocograms. *J Matern Fetal Med*, 9(5):311–8.

Ayres-de Campos, D., Sousa, P., Costa, A., and Bernardes, J. (2008). Omniview-SisPorto 3.5 - a central fetal monitoring station with online alerts based on computerized cardiotocogram+ST event analysis. *Journal of Perinatal Medicine*, 36(3):260–264.

Babcock, B., Babu, S., Datar, M., Motwani, R., and Widom, J. (2002). Models and issues in data stream systems. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS '02, pages 1–16, New York, NY, USA. ACM.

Bach, S. and Maloof, M. (2008). Paired learners for concept drift. In *Eighth IEEE International Conference on Data Mining, 2008. ICDM '08*, pages 23 –32.

Bache, K. and Lichman, M. (2013). UCI machine learning repository.

Baena-García, M., Campo-Ávila, J. D., Fidalgo, R., Bifet, A., Gavaldà, R., and Morales-Bueno, R. (2006). Early drift detection method. In *In 4th ECML PKDD International Workshop on Knowledge Discovery from Data Streams*, pages 77–86.

Barbará, D. (2002). Requirements for clustering data streams. *SIGKDD Explor. Newsl.*, 3(2):23–27.

Basseville, M. and Nikiforov, I. (1993). *Detection of Abrupt Changes: Theory and Applications*. Prentice-Hall, Englewood Cliffs, NJ, USA.

Berthold, M. and Hand, D. J., editors (1999). *Intelligent Data Analysis: An Introduction*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition.

Bifet, A. and Gavaldà, R. (2006). Kalman filters and adaptive windows for learning in data streams. In *Proceedings of the 9th international conference on Discovery Science*, DS'06, pages 29–40, Berlin, Heidelberg. Springer-Verlag.

Bifet, A. and Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. In *In SIAM International Conference on Data Mining*, Berlin, Heidelberg.

Bifet, A., Holmes, G., Kirkby, R., and Pfahringer, B. (2010). MOA: massive online analysis. *Journal of Machine Learning Research*, 11:1601–1604.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., New York, NY, USA.

Blair, R. C. and Higgins, J. J. (1980). A Comparison of the Power of Wilcoxon's Rank-Sum Statistic to That of Student's t Statistic under Various Nonnormal Distributions. *Journal of Educational Statistics*, 5(4):309+.

Chakrabarti, K., Garofalakis, M. N., Rastogi, R., and Shim, K. (2000). Approximate query processing using wavelets. In Abbadi, A. E., Brodie, M. L., Chakravarthy, S., Dayal, U., Kamel, N., Schlageter, G., and Whang, K.-Y., editors, *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt*, pages 111–122. Morgan Kaufmann.

Cormode, G. and Garofalakis, M. (2007). Sketching probabilistic data streams. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, SIGMOD '07, pages 281–292, New York, NY, USA. ACM.

Cormode, G. and Muthukrishnan, S. (2005a). An improved data stream summary: the count-min sketch and its applications. *J. Algorithms*, 55(1):58–75.

Cormode, G. and Muthukrishnan, S. (2005b). What's hot and what's not: tracking most frequent items dynamically. *ACM Trans. Database Syst.*, 30(1):249–278.

Correa, M., Bielza, C., and Pamies-Teixeira, J. (2009). Comparison of bayesian networks and artificial neural networks for quality detection in a machining process. *Expert Syst. Appl.*, 36(3):7270–7279.

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

Dawid, A. P. (1984). Statistical theory: the prequential approach. *Journal of the Royal Statistical Society. Series A*, 147:278–292.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comput.*, 10(7):1895–1923.

Domingos, P. and Hulten, G. (2001). Catching up with the data: Research issues in mining data streams. In *In Workshop on Research Issues in Data Mining and Knowledge Discovery*.

Duda, R. O. and Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. John Wiley & Sons Inc, 1 edition.

Çelik, T. (2011). Bayesian change detection based on spatial sampling and gaussian mixture model. *Pattern Recognition Letters*, 32(12):1635–1642.

Freedman, D. and Diaconis, P. (1981). On the histogram as a density estimator:L2 theory. *Probability Theory and Related Fields*, 57(4):453–476.

Frisén, M. (2008). Introduction to financial surveillance. Research Reports 2008:1, Statistical Research Unit, Department of Economics, School of Business, Economics and Law, University of Gothenburg.

Gama, J. (2010). *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC, 1st edition.

Gama, J. and Kosina, P. (2011). Learning decision rules from data streams. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1255–1260. AAAI Press.

Gama, J., Medas, P., Castillo, G., and Rodrigues, P. (2004). Learning with drift detection. In *In SBIA Brazilian Symposium on Artificial Intelligence*, pages 286–295. Springer Verlag.

Gama, J. and Pinto, C. (2006). Discretization from data streams: applications to histograms and data mining. In *Proceedings of the 2006 ACM symposium on Applied computing*, SAC '06, pages 662–667, New York, NY, USA. ACM.

Gama, J., Rocha, R., and Medas, P. (2003). Accurate decision trees for mining high-speed data streams. In *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 523–528, New York, NY, USA. ACM.

Gama, J. and Rodrigues, P. P. (2007). *Data stream processing*, chapter Learning from data streams - processing techniques in sensor networks, pages 25–39. Springer Verlag.

Gama, J., Sebastião, R., and Rodrigues, P. P. (2009). Issues in evaluation of stream learning algorithms. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 329–338, New York, NY, USA. ACM.

Gama, J., Sebastião, R., and Rodrigues, P. P. (2013). On evaluating stream learning algorithms. *Machine Learning*, 90(3):317–346.

Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., and Bouchachia, H. (2014, in press). A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4).

Gambús, P., A Jensen, E., Martínez Pallí, G., Fidler, M., and Kern, S. (2006). Modelling the interaction of propofol and remifentanil by means of an adaptive neuro fuzzy inference system (anfis). *European Journal of Anaesthesiology*, 23:134.

Gambús, P., Jensen, E., Jospin, M., Borrat, X., Martínez Pallí, G., Fernández-Candil, J., Valencia, J., Barba, X., Caminal, P., and Trocóniz, I. (2011). Modeling the effect of propofol and remifentanil combinations for sedation-analgesia in endoscopic procedures using an adaptive neuro fuzzy inference system (anfis). *Anesth Analg*, 112(2):331–9.

Gan, T., Glass, P., Windsor, A., Payne, F., Rosow, C., Sebel, P., and Manberg, P. (1997). Bispectral index monitoring allows faster emergence and improved recovery from propofol, alfentanil, and nitrous oxide anesthesia. bis utility study group. *Anesthesiology*, 87(4):808–15.

Gao, J., Fan, W., Han, J., and Yu, P. S. (2007). A general framework for mining concept-drifting data streams with skewed distributions. In *In Proc. SDM'07*.

Gibbons, J. D. and Chakraborti, S. (2003). *Nonparametric Statistical Inference (Statistics: a Series of Textbooks and Monogrphs)*. CRC, 4 edition.

Gibbons, P. B. and Matias, Y. (1999). Synopsis data structures for massive data sets. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 909–910.

Gilbert, A. C., Kotidis, Y., Muthukrishnan, S., and Strauss, M. J. (2003). One-pass wavelet decompositions of data streams. *IEEE Trans. on Knowl. and Data Eng.*, 15(3):541–554.

Glass, P. S., Bloom, M., Kearse, L., Rosow, C., Sebel, P., and Manberg, P. (1997). Bispectral Analysis Measures Sedation and Memory Effects of Propofol, Midazolam, Isoflurane, and Alfentanil in Healthy Volunteers. *Anesthesiology*, 86:836–847.

Gonçalves, H., Bernardes, J., Paula Rocha, A., and Ayres-de Campos, D. (2007). Linear and nonlinear analysis of heart rate patterns associated with fetal behavioral states in the antepartum period. *Early Hum Dev*, 83(9):585–591.

Guha, S., Koudas, N., and Shim, K. (2006). Approximation and streaming algorithms for histogram construction problems. *ACM Trans. Database Syst.*, 31(1):396–438.

Guha, S., Shim, K., and Woo, J. (2004). Rehist: Relative error histogram construction algorithms. In *In proceedings of the 30th International Conference on. Very Large Data Bases*, pages 300–311.

Guralnik, V. and Srivastava, J. (1999). Event detection from time series data. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 33–42, New York, NY, USA. ACM.

Hadjiliadis, O., del Valle, G., and Stamos, I. (2009). A comparison of 2-cusum stopping rules for quickest detection of two-sided alternatives in a brownian motion model. *Sequential Anal.*, 28(1):92–114.

Hartland, C., Baskiotis, N., Gelly, S., Teytaud, O., and Sebag, M. (2006). Multi-armed bandit, dynamic environments and meta-bandits. In *Online Trading of Exploration and Exploitation Workshop, NIPS*, Whistler, Canada.

Hinkley, D. V. (1971a). Inference about the change-point from cumulative sum tests. *Biometrika*, 58(3):509–523.

Hinkley, D. V. (1971b). Inference in Two-Phase Regression. *Journal of The American Statistical Association*, 66:736–743.

Hinkley, D. V. and Hinkley, E. A. (1970). Inference about the change-point in a sequence of binomial variables. *Biometrika*, 57(3):477–488.

Ho, S.-S. (2005). A martingale framework for concept change detection in time-varying data streams. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 321–327, New York, NY, USA. ACM.

Ho, S.-S. and Wechsler, H. (2010). A martingale framework for detecting changes in data streams by testing exchangeability. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2113–2127.

Hulten, G. and Domingos, P. (2003). VFML – a toolkit for mining high-speed time-changing data streams.

Hulten, G., Spencer, L., and Domingos, P. (2001). Mining time-changing data streams. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '01, pages 97–106, New York, NY, USA. ACM.

Ikonomovska, E., Gama, J., Sebastião, R., and Gjorgjevik, D. (2009). Regression trees from data streams with drift detection. In *Discovery Science*, pages 121–135.

Ioannidis, Y. (2003). The history of histograms (abridged). In *Proceedings of the 29th international conference on Very large data bases - Volume 29*, VLDB '03, pages 19–30. VLDB Endowment.

Ioannidis, Y. E. and Poosala, V. (1995). Balancing histogram optimality and practicality for query result size estimation. In Carey, M. J. and Schneider, D. A., editors, *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, San Jose, California, May 22-25, 1995*, pages 233–244. ACM Press.

Jagadish, H. V., Koudas, N., Muthukrishnan, S., Poosala, V., Sevcik, K. C., and Suel, T. (1998). Optimal histograms with quality guarantees. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, VLDB '98, pages 275–286, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Jensen, E., Jospin, M., Gambús, P., Vallverdu, M., and Caminal, P. (2008). Validation of the index of consciousness (ioc) during sedation/analgesia for ultrasonographic endoscopy. In *Conf Proc IEEE Eng Med Biol Soc.*, volume 2008, pages 5552–5555.

Karras, P. and Mamoulis, N. (2008). Hierarchical synopses with optimal error guarantees. *ACM Transactions on Database Systems*, 33:1–53.

Katakis, I., Tsoumakas, G., Banos, E., Bassiliades, N., and Vlahavas, I. (2009). An adaptive personalized news dissemination system. *J. Intell. Inf. Syst.*, 32(2):191–212.

Kearns, M. J. and Vazirani, U. V. (1994). *An introduction to computational learning theory*. MIT Press, Cambridge, MA, USA.

Kelly, M. G., Hand, D. J., and Adams, N. M. (1999). The impact of changing populations on classifier performance. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 367–371, New York, NY, USA. ACM.

Kifer, D., Ben-David, S., and Gehrke, J. (2004). Detecting change in data streams. In *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30*, VLDB '04, pages 180–191. VLDB Endowment.

Kim, H., Rozovskii, B. L., and Tartakovsky, A. G. (2004). A nonparametric multichart cusum test for rapid detection of dos attacks in computer networks. In *International Journal of Computing and Information Sciences*, volume 2, pages 149–158.

Klinkenberg, R. (2004). Learning drifting concepts: Example selection vs. example weighting. *Intell. Data Anal.*, 8(3):281–300.

Klinkenberg, R. and Renz, I. (1998). Adaptive information filtering: Learning in the presence of concept drifts. In *Workshop Notes of the ICML/AAAI-98 Workshop Learning for Text Categorization*, pages 33–40. AAAI Press.

Kolter, J. Z. and Maloof, M. A. (2003). Dynamic weighted majority: A new ensemble method for tracking concept drift. In *Proceedings of the Third IEEE International Conference on Data Mining*, ICDM '03, pages 123–, Washington, DC, USA. IEEE Computer Society.

Kosina, P., Gama, J., and Sebastião, R. (2010). Drift severity metric. In *ECAI 2010 - Proceedings of the 19th European Conference on Artificial Intelligence*, pages 1119–1120.

Koychev, I. (2000). Gradual forgetting for adaptation to concept drift. In *In Proceedings of ECAI 2000 Workshop Current Issues in Spatio-Temporal Reasoning*, pages 101–106.

Kubat, M. (1989). Floating approximation in time-varying knowledge bases. *Pattern Recognition Letters*, 10(4):223–227.

Kuncheva, L. I. (2004). Classifier Ensembles for Changing Environments. In *Multiple Classifier Systems*, pages 1–15.

Kuncheva, L. I. (2008). Classifier ensembles for detecting concept change in streaming data: Overview and perspectives. In *2nd Workshop SUEMA 2008 (ECAI 2008)*, pages 5–10.

Last, M. (2002). Online classification of nonstationary data streams. *Intell. Data Anal.*, 6(2):129–147.

Lin, M.-Y., Hsueh, S.-C., and Hwang, S.-K. (2007). Interactive mining of frequent itemsets over arbitrary time intervals in a data stream. In *Proceedings of the nineteenth conference on Australasian database - Volume 75*, ADC '08, pages 15–21, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

Luginbühl, M. and Schnider, T. (2002). Detection of awareness with the bispectral index: two case reports. *Anesthesiology*, 96(1):241–3.

Marques de Sá, J., Sebastião, R., and Gama, J. (2011a). Tree classifiers based on minimum error entropy decisions. *Canadian Journal on Artificial Intelligence, Machine Learning & Pattern Recognition*, 2(3):41–55.

Marques de Sá, J., Sebastião, R., Gama, J., and Fontes, T. (2011b). New results on minimum error entropy decision trees. In *CIARP*, pages 355–362.

Mashour, G., Esaki, R., Vandervest, J., Shanks, A., and Kheterpal, S. (2009). A novel electronic algorithm for detecting potentially insufficient anesthesia: implications for the prevention of intraoperative awareness. *Journal of Clinical Monitoring and Computing*, 23:273–277.

Massey, F. J. (1951). The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78.

MATLAB®& Simulink® (2007). *Student Version R2007a*. The MathWorks Inc., Natick, Massachusetts.

Melek, W., Lu, Z., Kapps, A., and Fraser, W. (2005). Comparison of trend detection algorithms in the analysis of physiological time-series data. *IEEE Trans. Biomed. Engineering*, 52:639–651.

Mierswa, I., Scholz, M., Klinkenberg, R., Wurst, M., and Euler, T. (2006). YALE: rapid prototyping for complex data mining tasks. In *In Proceedings of the 12th ACM SIGKDD International PONZETTO & STRUBE Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 935–940, New York, NY, USA. ACM Press.

Minku, L. L. and Yao, X. (2012). Ddd: A new ensemble approach for dealing with concept drift. *IEEE Transactions on Knowledge and Data Engineering*, 24(4):619–633.

Minto, C., Schnider, T., Short, T., Gregg, K., Gentilini, A., and Shafer, S. (2000). Response surface model for anesthetic drug interactions. *Anesthesiology*, 92(6):1603–16.

Misra, J. and Gries, D. (1982). Finding Repeated Elements. *Science of Computer Programming*, 2:143–152.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition.

Monk, T. and Weldon, B. (2011). Does depth of anesthesia monitoring improve postoperative outcomes? *Curr Opin Anaesthesiol*, 24(6):665–9.

Montgomery, D. (2009). *Introduction to statistical quality control*. Wiley, New York, NY [u.a.], 6. ed edition.

Mouss, H., Mouss, D., Mouss, N., and Sefouhi, L. (2004). Test of page-hinckley, an approach for fault detection in an agro-alimentary production system. In *Control Conference, 2004. 5th Asian*, volume 2, pages 815 –818.

Muthukrishnan, S., van den Berg, E., and Wu, Y. (2007). Sequential change detection on data streams. In *Seventh IEEE International Conference on Data Mining Workshops, 2007. ICDM Workshops 2007.*, pages 551 –550.

Nishida, K. and Yamauchi, K. (2007). Detecting concept drift using statistical testing. In *Proceedings of the 10th international conference on Discovery science*, DS'07, pages 264–269, Berlin, Heidelberg. Springer-Verlag.

Page, E. S. (1954). Continuous inspection schemes. *Biometrika*, 41(1-2):100–115.

Paz, L. (2013). Galeno- modeling and control for personalized drug administration. User's manual, Departamento de Matemática da Faculdade de Ciencias da Universidade do Porto.

Petzold, M., Sonesson, C., Bergman, E., and Kieler, H. (2004). Surveillance in longitudinal models: Detection of intrauterine growth restriction. *Biometrics*, 60(4):1025–1033.

Pinto, C. and Gama, J. (2007). Incremental discretization, application to data with concept drift. In *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing*, pages 467–468, New York, NY, USA. ACM.

Poor, H. V. and Hadjiliadis, O. (2009). *Quickest Detection*. Cambridge University Press, New York, USA.

Poosala, V., Ioannidis, Y. E., Haas, P. J., and Shekita, E. J. (1996). Improved histograms for selectivity estimation of range predicates. In *SIGMOD Conference*, pages 294–305.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Rampil, I. J. (1998). A Primer for EEG Signal Processing in Anesthesia. *Anesthesiology*, 89:980–1002.

Roberts, S. W. (1959). Control chart tests based on geometric moving averages. *Technometrics*, 42(1, Special 40th Anniversary Issue):97–101.

Rodrigues, P., Gama, J., and Sebastião, R. (2010). Memoryless fading windows in ubiquitous settings. In *Proceedings of Ubiquitous Data Mining (UDM) Workshop, in conjunction with the 19th European Conference on Artificial Intelligence - ECAI 2010*, pages 27–32.

Rodrigues, P. P., Gama, J., and Pedroso, J. (2008). Hierarchical clustering of time-series data streams. *IEEE Trans. on Knowl. and Data Eng.*, 20(5):615–627.

Rodrigues, P. P., Sebastião, R., and Santos., C. C. (2011). Improving cardiotocography monitoring: a memory-less stream learning approach. In *Workshop on Learning from Medical Data Streams*, volume 765 - paper 7.

Ross, G. J., Adams, N. M., Tasoulis, D. K., and Hand, D. J. (2012). Exponentially weighted moving average charts for detecting concept drift. *Pattern Recogn. Lett.*, 33(2):191–198.

Ross, G. J., Tasoulis, D. K., and Adams, N. M. (2009). Online annotation and prediction for regime switching data streams. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, pages 1501–1505, New York, NY, USA. ACM.

Sayad, S. (2011). *Real Time Data Mining*. Self-Help Publishers.

Scott, D. W. (1979). On optimal and data-based histograms. *Biometrika*, 66(3):605–610.

Sebastião, R. and Gama, J. (2007). Change detection in learning histograms from data streams. In *Proceedings of the 13th Portuguese Conference on Artificial Intelligence*, EPIA'07, pages 112–123, Berlin, Heidelberg. Springer-Verlag.

Sebastião, R. and Gama, J. (2009). A study on change detection methods. In Lopes, L. S., Lau, N., Mariano, P., and Rocha, L. M., editors, *New Trends in Artificial Intelligence, 14th Portuguese Conference on Artificial Intelligence*, EPIA'09, pages 353–364.

Sebastião, R., Gama, J., and Mendonça, T. (2008). Learning from data streams: Synopsis and change detection. In Cesta, A. and Fakotakis, N., editors, *STAIRS*, volume 179 of *Frontiers in Artificial Intelligence and Applications*, pages 163–174. IOS Press.

Sebastião, R., Gama, J., Rodrigues, P., and Bernardes, J. (2010). Monitoring Incremental Histogram Distribution for Change Detection in Data Streams. In Gaber, M. M., Vatsavai, R. R., Omitaomu, O. A., Gama, J., Chawla, N. V., and Ganguly, A. R., editors, *Knowledge Discovery from Sensor Data*, volume 5840, chapter 2, pages 25–42. Springer Berlin Heidelberg, Berlin, Heidelberg.

Sebastião, R., Martins da Silva, M., Rabiço, R., Gama, J., and Mendonça, T. (2012). Online evaluation of a changes detection algorithm for depth of anesthesia signals. In *Proc. 8th IFAC Symposium on Biological and Medical Systems*, pages 343–348.

Sebastião, R., Silva, M., Rabiço, R., Gama, J., and Mendonça, T. (2013). Real-time algorithm for changes detection in depth of anesthesia signals. *Evolving Systems*, 4(1):3–12.

Selbst, S. M. (2000). Adverse sedation events in pediatrics: A critical incident analysis of contributing factors. *Pediatrics*, 105(4):864–865.

Severo, M. and Gama, J. (2006). Change detection with kalman filter and cusum. In *Proceedings of the 9th international conference on Discovery Science*, DS'06, pages 243–254, Berlin, Heidelberg. Springer-Verlag.

Shewhart, W. A. (1925). The application of statistics as an aid in maintaining quality of a manufactured product. *Journal of the American Statistical Association*, 20(152):546–548.

Shewhart, W. A. (1931). *Economic Control of Quality Manufactured Product*. D.˜Van Nostrand Reinhold, Princeton, NJ.

Silva, M. M., Sousa, C., Sebastião, R., Gama, J., Mendonça, T., Rocha, P., and Esteves, S. (2009). Total mass TCI driven by parametric estimation. In *Proceedings of the 2009 17th Mediterranean Conference on Control and Automation*, MED '09, pages 1149–1154, Thessaloniki, Greece. IEEE Computer Society.

Street, W. N. and Kim, Y. (2001). A streaming ensemble algorithm (sea) for large-scale classification. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 377–382. ACM Press.

Struys, M., Jensen, E., Smith, W., Smith, N., Rampil, I., Dumortier, F., Mestach, C., and Mortier, E. (2002). Bispectral index monitoring allows faster emergence and improved recovery from propofol, alfentanil, and nitrous oxide anesthesia. bis utility study group. *Anesthesiology*, 96(4):803–16.

Sturges, H. A. (1926). The choice of a class interval. *American Statistical Association*, 21:65–66.

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

Viertiö-Oja, H., Maja, V., Särkelä, M., Talja, P., Tenkanen, N., Tolvanen-Laakso, H., Paloheimo, M., Vakkuri, A., Yli-Hankala, A., and Meriläinen, P. (2004). Description of the entropy algorithm as applied in the datex-ohmeda s/5 entropy module. *Acta Anaesthesiol Scand*, 48(2):154–61.

Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11(1):37–57.

Wald, A. (1947). *Sequential Analysis*. John Wiley and Sons, 1st. ed edition.

Widmer, G. and Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Mach. Learn.*, 23(1):69–101.

Yang, M. C. K., Namgung, Y. Y., Marks, R. G., Magnusson, I., and Clark, W. B. (1993). Change detection on longitudinal data in periodontal research. *Journal of Periodontal Research*, 28:152–160.

Yang, P., Dumont, G. A., and Ansermino, J. M. (2006). Adaptive change detection in heart rate trend monitoring in anesthetized children. *IEEE Trans. Biomed. Engineering*, 53(11):2211–2219.

Zliobaite, I. (2007). Ensemble learning for concept drift handling - the role of new expert. In Perner, P., editor, *Machine Learning and Data Mining in Pattern Recognition, 5th International Conference, MLDM 2007, Leipzig, Germany, July 18-20, 2007, Poster Proceedings*, pages 251–260. IBal publishing.

Zliobaite, I. (2009). Learning under Concept Drift: an Overview. Technical report, Vilnius University, Faculty of Mathematics and Informatic.