Adaptive Chemical Agent Detection in Dynamic, Changing Environments

Anna Ladi

PhD

University of York

Electronics

November 2015

Abstract

In this thesis a framework for adaptive chemical detection is developed, considering the application scenario of autonomous, robot mounted chemical agent detection in dynamic, changing environments. Chemical detection is performed by the Receptor Density Algorithm (RDA), a previously developed immune-inspired anomaly detection algorithm, which suffers from a decrease in its performance when the background environment changes. Focusing on the software part of the system, the goal of this thesis is to adapt the RDA quickly and autonomously, without requiring user feedback. The approach adopted is to first detect a change in the background data generating distribution, also defined as concept drift, and adapt in response to this detected change. Statistical hypothesis testing is used to determine whether there has been concept drift in consecutive time windows of the incoming sensor data. Five different statistical methods are tested on mass spectrometry data, enhanced with artificial concept drift that signifies a changing environment. The results show that, while no one method is universally best, statistical hypothesis testing can detect concept drift in the context of chemical sensing and it can differentiate between anomalies and concept drift.

The adaptation of the system, which is triggered by the detection of concept drift, is achieved by switching to an ensemble (a set) of RDAs , created from a pool of pre-existing candidates. A novel mechanism for evaluating and selecting the members of the ensemble from this pool is proposed; it uses implicit performance information, extracted from the dynamics of the RDA, and does not require new user input to evaluate the candidates for the new environment. An ensemble of 5 members, selected in this way is found to be significantly better than a single RDA, the previously known best, reducing both the false detections and the number of missed anomalies. This method for selecting the ensemble members is also found significantly better than populating the ensemble based on their performance of the environment before concept drift. Finally, it is found that the ensemble can be created online, with its performance converging to the offline variant.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my supervisors, Jon Timmis and Andy Tyrrell, for giving me the opportunity to pursue this PhD, and for their valuable guidance and constant support during the last four years. I would also like to thank Dstl for funding this project and especially Peter J Hickey, for providing constructive feedback throughout this work.

I also want to express my gratitude and love to my family, for supporting me in every possible way and for always being there for me. Without them, none of this would be possible.

To my dear friends, Thanos, Eva, Kirk, Liana, Tina and Sotiris to mention a few, I owe a big thanks for all their help and patience during the last months, but mostly for all the fun we've had throughout the last years and for making York a home away from home. To my friend and lunch buddy Pratik, for all the fun and worries we shared, thank you.

Last, but definitely not least, I can't thank enough my partner Tasos, for his love, encouragement, proofreading, endless skyping and for always believing in me.

Declaration

I declare that the research described in this thesis is original work, which I undertook at the University of York during 2011 - 2015. Except where stated, all of the work contained within this thesis represents the original contribution of the author.

This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as References. Some parts of this thesis have been published in conference proceedings; where items were published jointly with collaborators, the author of this thesis is responsible for the material presented here. For each published item the primary author is the first listed author.

- Anna Ladi, Jon Timmis, Andy Tyrell, and Richard J Smith. An automated sniffer dog: Real-time adaptive molecular signature detection. *Sponsoring Institutions*, page 64, 2012. [1]

- Anna Ladi, Jon Timmis, Andy M. Tyrrell, and Peter J. Hickey. Statistical hypothesis testing for chemical detection in changing environments. In *Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2014 IEEE Symposium on*, pages 77-84. IEEE, 2014. [2]

Chapter 1

Introduction

The detection of chemicals of interest is essential in a number of domains including, but not limited to, safety and security, environmental monitoring and process monitoring [3]. Typically, a chemical detection task involves a number of chemical sensors or analytical chemistry instruments (e.g. mass spectrometers), which continuously sample their environment, and algorithms that are responsible for analysing data from those sensors.

In previous work, which this thesis will build on, the bio-inspired Receptor Density Algorithm (RDA) [4] was proposed. The RDA is inspired by the signalling mechanisms of T-cells of the immune system and is used for anomaly detection and generation of noise-free signatures. In previous work the RDA has been successfully used for the detection of chemicals of interest in mass spectrometry data [5].

The basic scenario used for this thesis is that of a mobile robot with chemical sensing capabilities being deployed in an environment and performing detection of chemicals of interest. This can be achieved using the RDA, which detects chemical agents of interest in the form of anomalies, i.e. observations that deviate from what is expected to be encountered in this specific environment. This paradigm is challenged when considering that real world, possibly long term applications are often associated with changing environments and conditions, and that the RDA has no adaptation mechanism to respond to such changes.

The goal of this thesis is to develop an autonomous, adaptive system, keeping the RDA as the core anomaly detection unit, that will be able to detect when there is a need to change and adapt itself with minimal user feedback. This adaptive system can then be used by a robot equipped with chemical sensors, navigating through changing and dynamic environments and performing chemical detection autonomously.

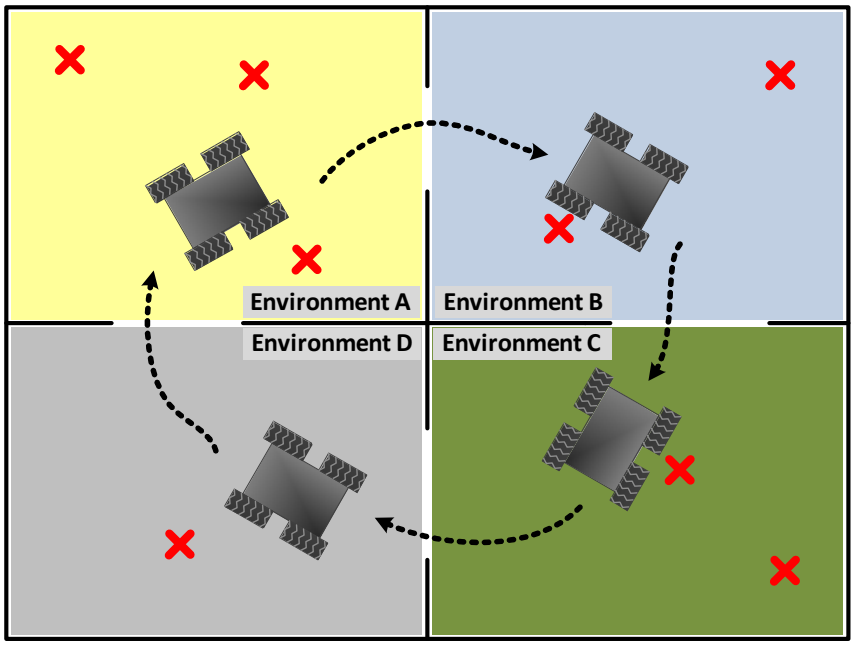


Figure 1.1: Example of chemical detecting robot navigating through changing environments. The robot should ideally detect accurately the anomalies (marked with a red x), in spite of the changing chemical background of the different environments. A robot is deployed in environment A (for example a room in a building) with the task of detecting chemicals that deviate from what is expected to be found in this environment. It is assumed that the RDA will be calibrated using some form of labelled data that reflect what is considered normal or anomalous in environment A. When the robot however moves to a different room, this is likely to have a new chemical background, for example a different set of background chemicals is expected to be found in the kitchen of a house than the living room. If the chemical background is sufficiently different the RDA is likely to fail in discriminating between what is normal and the anomalies present in environment B.

To demonstrate the necessity and importance of such a system that can deal with dynamic changing environments, two example scenarios of robot-mounted chemical detection are presented: the first scenario consists of the detection of possibly hazardous chemicals in a building with multiple environments/rooms (see figure 1.1). The robot is expected to detect chemicals that deviate from what is considered normal in a specific environment. Any training of the chemical detection algorithms, in this case the RDA, is based on information about both what is considered normal and what is considered anomalous. As the robot is deployed in a room of the building, say environment A, it will be calibrated and trained to detect chemicals within the chemical background of the specific room, i.e. a specific "normal" background. When the robot moves to a different room, environment B, the notion of normality changes with the change in the environment. For example, a different set of chemicals would be considered normal when the robot is in the living room

and when it is in the kitchen. This change in the model of normality can cause the system to fail in detecting anomalies, as the information it was trained on is now to some extent obsolete.

A second possible scenario is the one of long-term field deployment. Long term field applications carry the inherent assumption that the environment will change, at least as far as temperature, humidity and pressure are concerned. In addition to the environmental changes, other problems might be encountered, related to the degradation of the hardware platform; the response of a sensor itself can change when exposed to the same chemicals as a result of environmental conditions, ageing or poisoning of the sensor [6].

The common aspect of both scenarios is that they represent dynamic situations, where a static system could possibly not cope with the change. The RDA, even though it has regulating mechanisms that allow for some tolerance, it is, in its current state, static; when the background changes significantly, it is expected to fail. On the contrary, an adaptive system is needed that will be able to adapt to change and remain functional. The **Research question** around which this thesis is built is the following:

To what extent can a system that autonomously detects a change in the environment and adapts in response to that detection address the decrease of the performance of an RDA-based chemical detection system, when it is deployed in changing environments?

1.1 Requirements

The main challenge associated with an autonomous chemical detection system is **autonomy** itself. Hence, the primary requirement is that the system makes decisions both on when to adapt and how to adapt autonomously, without relying on user directions and/or feedback. The availability of user feedback, for instance in the form of new labelled data for re-training, cannot be assumed to be available in a deployed chemical detection system such as the ones described. In addition to that there are two secondary challenges, which are **online** and **real-time** adaptation[1]. Although the online and real-time adaptation

[1]The challenges outlined here are in line with the requirements for this project, as discussed with the project DSTL (Defence Science and Technology Laboratory) collaborators. The requirements of online and real-time adaptation as considered secondary, because they depend to some extent on the computational resources available for any specific application.

requirements are relatively relaxed, the proposed systems will in principle attempt to address all three challenges.

An online system is one that uses data and information as it is being acquired, i.e. it must make decision on runtime, without knowledge of the future [7]. A real-time system associates the tasks to be performed with deadlines and the assessment of its operation depends not only upon the task output, but also upon the time in which it is completed [8]. A *hard real-time* system must ensure that all deadlines are met, but for *soft real-time* systems the goal becomes minimising the delay of completing some tasks, or meeting some deadlines within controlled timing bounds. In this work the system can be considered as soft real-time system (or near real-time); there is no hard deadline for the adaptation, but in principle it should be able to adapt in a timely manner.

1.2 Contributions and thesis novelty

This thesis addresses the software, algorithmic parts of a robotic adaptive chemical detection system; the hardware aspect, i.e. the robot itself or an embedded implementation is not in the scope of this thesis. Nonetheless, throughout the presented work, the limitations and requirements that are in principle associated with the application of interest, as outlined in section 1.1 are taken into account and addressed. The scope of this work is to create an anomaly detection system that can adapt to changes in the application data. While in this thesis mass-spectrometry chemical data is used, the principles of the developed system can apply to other domains that involve similar types of time-series data.
There are two novel contribution areas of this thesis: the first is the detection of concept drift (change in the context of the problem) and the second is the adaptation in response to that change, via the use of an ensemble classifier. In detail:

- ***Detection of concept drift.*** A changing background environment constitutes a change to the context of the problem, as the notion of what is normal and what is an anomaly can shift. In machine learning this is defined as concept drift [9]. To detect whether concept drift has happened, statistical hypothesis testing has been used in this work. Statistical hypothesis testing processes the data in two consecutive windows and tries to determine if they come from different distributions. If they do, concept drift is declared and an adaptation mechanism is triggered. One **contribution** of this thesis is that a number of different statistical hypothesis testing

methods have been compared on real chemical data, enhanced with artificial concept drift. The dataset used contains both anomalies and artificial concept drift. A second important contribution is that the ability and limitations of using statistical hypothesis testing to differentiate between concept drift and anomalies has been established.

- ***Adaptation using ensembles.*** The detection of concept drift is used to trigger the adaptation of the system. For the adaptation, the concept of ensemble learning has been employed. An ensemble is a set of classifiers (or anomaly detectors), whose decisions are combined into a single output. It has been shown that if the members of the ensemble are reasonably accurate and diverse, the probability of the ensemble making an error decreases, compared to the probability of error of its members [10]. In the work presented in this thesis, ensembles are used in order to adapt to a new unknown environment, by utilising existing anomaly detectors (RDAs) evolved for the previous, known environment. In particular, after concept drift is detected (environment changed), the suitability of the candidate RDAs is estimated based on implicit performance information, and the best are selected to form an ensemble that will perform anomaly detection in the new environment. The **contributions** of this analysis include: (a) The extension of the RDA, as the base learner of an ensemble. (b) The extraction of an implicit performance vector that can estimate the performance of a given parameter set for the RDA when used in an unknown environment. (c) A novel mechanism to select the members of the ensemble from a diverse pool of candidates, using this implicit performance vector.

The novelty of this thesis is that an ensemble adaptation module is **autonomously triggered** when there is concept drift and **reuses existing resources** and **implicit performance** estimates in order to minimise the decrease in the performance of the RDA under concept drift, without requiring additional user feedback.

1.3 Thesis outline

This thesis addresses the adaptation of a chemical detection systems under the presence of concept drift. It is outlined as follows:

Chapter 2 introduces the domain of chemical agent detection. In this review key challenges are identified, related to the application of robot mounted chemical detection

and existing strategies that can be used to handle these challenges are discussed. The RDA is introduced in this chapter and its advantages compared to other existing methods are discussed.

Chapter 3 discusses the problem of concept drift and adaptation. Relevant definitions, challenges and adaptation strategies are reviewed, including the paradigm of ensemble learning and its applications in adaptation in response to concept drift.

Chapter 4 addresses the detection of a change in the environment. Statistical hypothesis testing is applied in chemical data enhanced with artificial concept drift. Five statistical hypothesis testing methods are compared. Additionally, the combination of these methods with the existing anomaly detection system is studied.

Chapter 5 addresses the adaptation in response to concept drift. Evolutionary ensembles are selected to combine solutions evolved for a known environment into an ensemble that will perform well on the unknown environment B. A novel mechanism is proposed for the selection of the members of the ensemble, based on implicit performance information.

Chapter 6 concludes the thesis and outlines the contributions of this work, as well as possible future research directions.

Chapter 2

Detection and Identification of Chemicals of Interest

The task of detecting chemicals of interest can be broken down into two main fields of research: sensing technology (hardware) and data processing (software). The algorithms and techniques that are used for the processing of the data collected by chemicals sensors and instruments are closely related to the output of the specific sensor or instrument. For instance, analytical chemistry instruments produce high resolution spectral signatures, therefore algorithms that suffer from the curse of dimensionality are not suitable for processing the data. On the other hand, arrays of simple, cross selective gas sensors, usually require more advanced pattern recognition approaches to fully exploit the acquired data.

For this reason, although this thesis focuses on the software aspect of the system, it is considered useful to provide at least an overview of the sensing technology available and commonly used in the research community. The main focus of this chapter, however, will be on the software part of the system, the intelligent processing of information acquired from chemical sensors in order to detect and identify chemicals of interest. Typically this is viewed strictly as an identification task; A chemical fingerprint is extracted through some processing of the sensor data and it is classified or matched against a library of chemical fingerprints of known compounds.

The detection of chemicals of interest can also be viewed as an anomaly detection task, rather than a pattern recognition problem. The reason for this is that in a real-time chemical detection task, the detection of any "out of context" odour. i.e. an anomaly, can be worth investigating. Moreover there can be unknown target chemicals (without an existing library entry) in a specific environment, or the target chemicals can be masked.

The Receptor Density Algorithm [4], which is the anomaly detection algorithm that will be used here, performs anomaly detection on incoming chemical sensor data, it extracts signatures for the detected anomalies and then, as a post processing step, matches these signatures against a library of known analytes.

2.1 Structure of this chapter

In section 2.2 a brief review of chemical sensing technologies will be presented, and in section 2.3 the relevant research on chemical detection, seen as an identification problem will be discussed, along with challenges and limitations related to real-time robotic systems. Section 2.4 will outline the basic concepts related to anomaly detection, as an alternative way to treat the problem of chemical detection and relevant work. Last, section 2.5 will introduce the bio-inspired Receptor Density Algorithm, an anomaly detection algorithm that has been successfully applied to the detection of chemicals of interest, and will discuss its advantages and the reasons it has been selected for the work carried out in this thesis.

2.2 Chemical sensing technology for airborne chemicals

Several chemical sensing instruments can be used for chemical monitoring and detection. Some of the available technologies that are the most commonly used in the literature will be reviewed in the following analysis. In this thesis only sensing a chemical in its gas phase is considered. This brief analysis will outline the different categories of sensors, their principle physics and operating mechanisms, as well as the advantages and limitations associated with each. Liu et al. present the key gas sensing technologies in [11] and they use some indicators to evaluate the performance of the available sensors, which are summarised in table 2.1.

One of the technologies that can be used for chemical sensing is **chromatography and spectrometry**. Typically a chromatograph separates a chemical mixture into individual compounds. A chromatograph is equipped with an appropriate column in which the mixture travels. Due to their different properties, the components of the mixture reach the detector at the end of the column at different rates [12]. A subclass of this is Ion Mobility Spectrometry (IMS). In this case the mixture that is injected into the spectrometer is ionised. The transit time of the ions under the presence of an electrical field, as they pass through a drift region is measured [13]. In contrast to IMS, mass

Indicators used for sensor evaluation	
Sensitivity	The minimum gas concentration at which the sensor can detect a chemical. High sensitivity means that the chemical can be detected even in low concentrations.
Selectivity (or specificity)	The ability identify and react to a specific gas in a gas mixture.
Response time	The time between the increase in the concentration of a gas and the change in the output of the sensor.
Reversibility	The ability of the sensor to return to its original condition after the detection.
Adsorptive capacity	Surface phenomenon that affects the sensitivity and selectivity.
Miniaturisation and portability.	
Energy consumption.	
Fabrication cost.	

Table 2.1: Indicators used for sensor evaluation, adapted from [11]. Miniaturisation and portability is not included in the original indicators list, but it has been added here, as it is commonly used in reviews as an indication of sensor capabilities, especially when in situ or mobile monitoring applications are considered.

spectrometry requires a vacuum and measures the mass to charge ratio of the molecular fragments [12]. Both IMS and mass spectrometry produce a high-dimensional output with many channels corresponding to drift times or mass to charge ratios respectively. In general, gas chromatography (GC) is a highly sensitive and selective, typical laboratory analytical technique; however, the cost of GC is high, and its miniaturization for portable application "needs more technological breakthrough" [11].

Optical methods are also mainly based on spectroscopy; the two most commonly used instruments are infrared (IR) source gas sensors and Raman detectors. IR sensors can detect gases with unique infrared absorption signatures in the 2-14 μm range [12], while Raman spectroscopy exploits the vibrational structure of analytes [14]. In [11] it is pointed out that optical methods have relatively short response times and stable performance in changing environmental conditions, which makes them suitable for real-time detection tasks. Their application on portable gas sensors is limited, however, due to miniaturisation issues and relatively high cost.

When real world field applications are considered, like the detection of explosives and explosive related illicit materials in soil, water, or as hidden material, common limitations of the instruments described are that they can be too big, too sophisticated to handle, or lacking the necessary sensitivity [15]. Another family of sensors that can address such limitations, uses portable, easy to operate and low cost gas sensors [15]. Gas sensors operate on the principle that their physical properties are altered in some measurable, characteristic way, upon coming in contact with specific gaseous chemicals in the ambient atmosphere [16]. Electrochemical sensors and mass sensors are the two most commonly used types belonging in this category.

Electrochemical sensors output a single value at one timestep, which corresponds to conductivity, current or voltage. The common principle is that these sensors are equipped with a chemically sensitive material, whose interaction with the chemical of interest causes a change in its electrical characteristics [14]. The chemically sensitive material is selected according to which gas needs to be detected. Two commonly used types of electrochemical sensors are metal oxide semiconductors (MOX) and polymer absorption chemiresistors. MOX sensors sense combustible and reducing gases that interact with oxygen at the surface of the sensors at elevated temperatures; this interaction causes a change in resistance. Polymer absorption chemiresistors measure the change in resistance that is caused by the absorption of an analyte into a polymer film [14]. MOX have several advantages, such as low cost and high sensitivity. Moreover, they have a long lifespan and some tolerance to changing environmental conditions [16]. Nonetheless, they operate on a high temperature, which increases their power consumption and they have long recovery period, which can limit their use in real-time applications [11]. Polymer sensors can detect some inorganic gases and Volatile Organic Compounds (VOC) which can be hazardous to humans, that cannot be detected by MOX sensors. Their advantages include high sensitivity, short response time and low power consumption, as they operate in room temperature. Their disadvantages are long term instability, irreversibility and poor selectivity.

Finally, **mass sensors** measure the mass of materials that stick to the surface of the device. Main examples of this category are the Surface Acoustic Wave (SAW) and the Quartz micro balance (QMB) sensors [13]. SAW sensors are small, low power sensors with high sensitivity, but have limited selectivity and can react strongly to water vapor [12]. QMB sensors have a rapid response and short recovery time, also low power consumption and long term stability and lifetime [16]. Their limitations include comparatively low sen-

sitivity to the target gas, poor signal to noise performance and sensitivity to variations in humidity [16]. Both electrochemical and mass sensors show limited selectivity and cross selectivity. To address that, but also to increase the information content of these sensors, it is very common to combine more than one of them (each one designed for a different analyte) in a sensor array.

The sensors and instruments described above, especially the simpler devices (electrochemical and mass sensors) are not totally selective, i.e. they do not respond to a single chemical compounds, ignoring all the rest. Moreover they are sensitive to changes in the environment and conditions such as humidity and temperature. The coupling of chemical sensors or instruments with pattern recognition, or anomaly detection approaches can increase their detection capabilities, and it is studied in the next section.

2.3 Chemical identification/classification

After the selection of the appropriate sensors or instruments for the problem at hand, the processing of the data collected by the sensors follows. This is commonly treated as an identification, or classification problem. A chemical signature, or fingerprint is extracted from the sensor data, and then it is matched against a library of signatures of known compounds. The development of detection algorithms is directly related to the instrumentation and factors such as the dimensionality and modelling complexity of the data. In this section the basic families of approaches both for spectral data and sensor arrays are reviewed, along with robotics applications that use machine olfaction (chemical sensing).

Shaffer in [17] outlines 6 requirements for pattern recognition algorithms that are part of a chemical detection system. These are: (1)*High accuracy* - for field measurements and safety and security applications low false alarm rate is required for ideally no missed classifications. (2)*Speed* - especially if real-time analysis is needed, computationally intensive algorithms may not be suitable for this application. (3)*Simple to train* - when the library is expected to be updated, the classifier will need to be retrained. (4)*Low memory requirements* - for field portable sensor application, where the algorithms might be implemented in micro-controllers with limited memory resources. (5)*Robust to outliers* when running on unknown, uncontrolled environments the system should only recognise what it is trained for and not produce high false positives, by misclassifying noise or new ambiguous sensor signals. (6)*Measure of uncertainty* - confidence measure associated with each classification.

2.3.1 Using spectrometry and/or analytical chemistry

Chemical detection methods that use data gathered from spectrometers can employ one or a combination of the following to extract knowledge about chemicals of interest: domain specific analytical chemistry methods, statistical techniques, or pattern recognition methods / machine learning methodologies. In [18] two approaches are compared, a pure analytical, domain knowledge based classifier and a chemometric classifier. The application is the classification of aerosol data obtained through GC-MS, a common pairing of Gas Chromatography and Mass Spectrometry. A domain knowledge classifier is a group of classification rules set manually by a domain expert. These rules are based on the presence and/or relative intensity of characteristic masses in the mass spectra recorded for each peak or relevant retention times. A chemometric classifier uses statistical methods for spectral transformations and analysis. For the chemometric classifier used in [18] weighted sums of spectral features, or combinations of spectral transformations are thresholded and used to classify the data. The difference is that training data and multivariate statistical analysis, instead of expert knowledge, is used to determine the optimal weights or the optimal transformations in order to identify a specific class. Comparing the performance of the two, they conclude that expertise and domain knowledge can lead to accurate and efficient classifiers, especially in the case of well defined classes, and is often the easiest strategy to implement; it does not require analysis and training to determine the equation forms and classification thresholds, as does the chemometric classifier. On the other hand chemometric classifiers achieve comparable performance and they have the advantage of not requiring expert knowledge; additionally, they can prove beneficial when it is unclear which mass spectral characteristics are important to distinguish a specific compound class. It is added that the combination of the two provides an even better method of reducing manual evaluation of spectra and increase confidence in the classification.

Using domain specific and specialised spectral decomposition methodologies has an important disadvantage; it limits the generalisation of the system. The system is very specifically designed to deal with limited classes and compounds related to a particular application and it relies on data interpretation from a human expert. Varmuza et al. in [19] argue in favour of statistical, chemometric techniques over domain specific analytics. They note that the GC-MS combination is powerful but the interpretation of the huge amount of data which is typically produced during a GC-MS analysis poses a challenge, as does the fragmentation process of the ions occurring in a mass spectrometer, which

is too complex to be modelled comprehensively. A statistical approach can be used to develop algorithms that recognise the presence or absence of certain substructures from a low resolution mass spectrum. This paper introduces software that allows the use of chemometric mass spectral classifiers that can indicate some of the substructures that are present or absent in the molecule of interest. It is suggested that the purpose of computer assisted spectra interpretation is to support the non-specialist to automate some parts of data evaluation rather than replace human experts.

Gardner et al. in [20] propose a robot-based Raman detector for chemical, biological and explosive chemical agents, using a combination of analytical methods and basic pattern recognition. Raman spectrometry, which reveals the molecular composition of materials, extracts multiple spectra for one sample. If the variability of the spectra indicates a mixture, an un-mixing algorithm identifies its components; if not a mean spectrum is used as the descriptor of the sample. Either the mean spectrum or the components are matched against a library of known signatures using Euclidean or Machalanobis distance. This method is tested on only four target compounds and some mixtures of those, which makes it very specifically tuned for the detection of only a few targets. Additionally, in the setup proposed, they guide the sensor to the sample (no continuous sampling/monitoring) and the sampling/processing phase takes up to 10 minutes. Although this can be used if only specific, identifiable targets are to be assessed, it limits the continuous detection scenario.

Feature extraction - dimensionality reduction. The high dimensionality of IR spectroscopy, mass spectroscopy and gas chromatography can seriously limit their application, as the analysis of pattern vectors of 100 or more dimensions is often required [17]. Consequently, reducing the dimensionality in spectral data is a common element of the detection algorithms. A common approach that addresses this issue, is dimensionality reduction or feature extraction. The opinion of computerisation of spectral analysis and solutions that remove or limit the necessity of using domain specific techniques and experts is shared by Praisler [21] and extended to the problem of reducing the dimensionality of spectral data. In [21] Principle Components Analysis (PCA) is used for feature extraction from GC-FTIR (Gas Chromatogrphy - Fourier Transform Infrared Spectroscopy) data. The principle components are determined using training samples, associated with known identities. Then, a SIMCA (Soft Independent Modelling of Class Analogies) classification is

initiated: the class boundaries are modelled based on the similarity of the samples within the class, taking into account the Euclidean distances between the class members in residual PC space, instead of the original feature space. The resulting system is applied in forensic toxicology, to discriminate between illicit amphetamines and legal compounds. It is highly specific and sensitive, achieving a classification rate of 93.92% , while it is argued to be an inexpensive, rapid and easy approach that does not require extensive training.

In [22] IMS is used to detect VOCs that correspond to lung cancer, from breath analyser samples. After some pre-processing and spectra correction, statistical methods are used to analyse the obtained sample spectra. A merging regions algorithm, typically used in image segmentation, is used to discriminate and localise the different peaks and then the peak positions are clustered. This peak pattern analysis produces a feature vector for each sample of 23 VOC peak variables. Using this feature vector, linear discriminant analysis (LDA) is used to find a linear combination of features which characterizes or separates two or more classes. Breath analysis by IMS, using 23 discriminating peak regions, provided an excellent classification rate in patients with lung cancer and healthy controls. One limitation of this approach is that the number of discriminating peaks needed is likely to rise, if more classes are added (for instance subclasses of non-healthy patients).

Bio-inspired methods (ANNs and AIS) for chemical detection. An approach often used for detection and classification tasks is Artificial Neural Networks (ANNs). ANNs is a paradigm inspired from biology, which is used in machine learning, typically as a pattern recognition or classification scheme [23]. An ANN comprises of a set of nodes called neurons, which are simple processing units and a set of connections between them. The connections are associated with weights, which determine the degree of participation of a neuron's output to the rest of the network's calculations. A subset of neurons act as input nodes, and another subset that act as output nodes. Typically a neural network has to learn a mapping between the input and the output, by iterating through a training set, until the error between the network output and the target output is minimised. Similarly, Artificial Immune Systems (AIS) is another bio-inspired paradigm which can be used for detection and/or identification. AIS is inspired by the mechanisms of the immune systems for recognising and handling threats [24].

When processing chemical spectra a very popular approach that moves further away from domain specific data processing is Artificial Neural Networks (ANNs). In [25] a feed

forward neural network trained with the back-propagation algorithm is used to classify chemicals into chemical classes after applying a set of pre-processing steps to the spectra. In [26], ANNs are used to detect hazardous materials hidden by innocuous materials. The spectra come from a simulator of the gamma prompt neutron activation analysis and the dimension of each spectrum is lowered through binning to 100 spectral bands. After being trained for the recognition of 19 unadulterated materials, the network is presented with samples of C4 hidden among other materials, such as silk and rubber and it manages to identify the presence of the explosive. This is mainly attributed to the characteristic presence of carbon and nitrogen in the C4-containing mixtures, which raises concerns as to how well this approach could generalise for other hazardous materials with less distinctive signatures.

Artificial immune systems (AIS) also find application in chemical identification. For such applications the most appealing property of the immune system is that it maintains a population of constantly evolving detectors (antibodies) that can identify known and unknown foreign agents (antigens). In [27] an immune-genetic approach is used for the identification of spectra (of single products or mixtures), through their binary, binned representations. The system involves evolving a population of specialists (detectors) for each chemical product, so that they match only the compound for which they are evolved and not the normal state of the spectrum, or spectra generated by other chemicals. During runtime a compound (an antigen) will be matched against all specialists (antibodies) and is associated with one or more (if it is a mixture) products, while if a peak of the spectrum remains unassigned, the compound is considered unknown and a new population of specialists is evolved for it. The algorithm is tested on mixtures, whose composition it is able to identify, especially as more specialists are assigned to each product. Because multiple detectors are assigned to every class, this approach can be challenged with the addition of many classes, in terms of both memory and computation requirements. A similar immune-genetic approach is used in [28] combined with a multilayer network strategy for the online classification of chemical spectra. The highest layer is responsible for the constant evolution of specialists (antibodies), which act as prototypes for non-self elements (in general and not for specific agents as in [27]), using principles from the domain of AIS. This level periodically distributes the new specialists to the lowest level and is also responsible for other actions like suitable sensor selection and the combination of decisions and feedback from the lower layers. The detection takes place at the lowest level where the sensor inputs

are matched against the library of antibodies. Results indicate that the detection rates depend on the number of chemical compounds that need to be detected [28] [29], as more generalised antibodies need to be evolved for multiple targets; this poses great challenges in complex detection tasks.

2.3.2 Sensor arrays - electronic noses

A very common tool used in chemical detection and especially in the detection and identification of VOCs is the electronic nose, the combination of a gas sensor array with pattern recognition algorithms. An electronic nose performs odour detection, mimicking the human olfactory system and it is a rapid, inexpensive alternative to instrumental, analytical methods, such as spectrometry techniques, that are often expensive and require trained personnel [30]. Moreover, sensor arrays produce signals of lower dimensions, typically 3-16 features, which corresponds to 3-16 sensors; therefore it is easier to process by pattern recognition algorithms, compared to spectral data [17].

Every sensor in the array senses the concentration of a specific gas in the ambient atmosphere and converts it into an electrical signal. The signals from all the sensors form an output pattern, the fingerprint or signature of a chemical agent. This serves as input to the pattern recognition algorithm after possibly passing through some pre-processing stages such as feature extraction (the use of one or more transformation of the input to produce new salient features) and data normalisation. Typically the sensor array must be exposed to all the target odours, so that a library of signatures can be created. This library serves as the training set of the pattern recognition algorithm.

A general review of pattern recognition algorithms that are used in electronic noses is presented in [30], along with sensor selection principles and strategies. Bicego et al in [31] support that the algorithmic part of the electronic nose is a very important component as it determines the selectivity of the instrument, especially when dealing with device miniaturisation; hence a flexible calibration and recognition tool is needed. To this end they compare the k^{th} Nearest Neighbour algorithm and artificial neural networks (ANN), as well as several dimensionality reduction techniques, that reduce the computational complexity. The best performance is achieved through an ANN trained using a Reactive Tabu Search (RTS) strategy, while for dimensionality reduction, Discriminant Analysis (DA) has been shown to reduce computational complexity (important in miniaturisation), without substantial information loss. An electronic nose is coupled with Support Vector

Machines (SVM) in [32] for the task of odour recognition. Since the SVM is traditionally used to find the optimal hyperplane separating two classes, several SVMs have to be trained to distinguish one class from all the rest. Recognition is then performed using the leave-oneout procedure. While the results compare favourably to two neural network approaches, the proposed system could be challenged as more target classes are added to the problem description. More SVMs would need to be trained (and possibly kept up to date) in such a case, increasing the computational complexity.

On the basis of the requirements for pattern recognition algorithms summarised in the beginning of the section, Shaffer in [17] compares 7 pattern recognition algorithms: Nearest Neighbor(NN), Mahalanobis Linear Discriminant Analysis (MLDA), Bayes Linear Discriminant Analysis (BLDA), SIMCA, Back Propagation trained ANNs (BP-ANN), Probabilistic Neural Networks (PNN) and Learning Vector Quantization (LVQ), on simulated and real data collected with a four to six SAW sensors array. Overall, the best method is found to be the BP-ANN, whose main drawback is that it involves numerous parameters which leads to long training times. Generally the non-linear methods are more suitable, because chemical data is typically non linear. Algorithms that perform comparably well are the LVQ, which only fails in associating a classification with a confidence measure, and the PNN which trains fast, but has high computational requirements for prediction. Continuing this study in [33] the improved PNN (IPNN) is proposed, a combination of PNNs and LVQ. The PNN models the whole training vector space by assigning every training pattern vector to a hidden layer node. The class of every test pattern is then predicted by propagating the test vector through the network and estimating the posterior probability of every class. This makes the PNN slow as it compares a new pattern against the hidden layer, i.e. the entire training set. The IPNN improves on this by using the LVQ algorithm to compress the hidden layer to a set of reference vectors. Further improvements concern the streamlining of the training of the IPNN, and parameter selection through a combination of rules of thumb and automated processes. The IPNN compares favourably with the results of the other pattern recognition methods, on the same datasets. It performs better than PNN, because it greatly reduces memory and computation requirements; the hidden layer is reduced to 15% its size. Also the IPNN performs slightly better than LVQ and it provides a confidence measure, which the LVQ was lacking. Additionally an outlier rejection strategy has been implemented, where an outlier is defined as a sample that the network does not know how to classify. The network

manages to recognise as outliers, samples from classes that it was not trained on, with only 0.3% FPrate. However, the parameters for this strategy are somewhat arbitrary and application specific and have to be set by the user.

The combination of a sensor array and a neural network is common in the literature [34]. In [35] a neural network model, the Self Organising Map (SOM), is combined with fuzzy logic (FSOM) for the classification of 15 chemical agents, i.e. the SOM is trained to learn a set of fuzzy classification rules. The network is trained with 6-dimensional labelled data of all classes, and exhibits very high classification accuracy, when compared to other approaches such as the nearest neighbour classification and an LVQ network. This is mainly due to the way the decision border between output classes is treated in FSOM. However a large volume of correctly labelled training data is required and there is no suggestion for updating or retraining the network in case more agents need to be added to the library or if the signatures of the existing entries change (for example due to environmental reasons). Fuzzy rules and a self organised network are also coupled in the fuzzy ARTMAP algorithm, which is used for the detection of four different explosives in [36] and compared with a support vector machine (SVM) which learns through maximising the margin of a hyperplane that separates the output classes. The feature sets come from an array of 4 fluorescent polymer sensors and consist of the change of the response of a sensor upon exposure to the target analytes. The performance is similar for the two algorithms and generally the task of identifying 4 analytes is successful, but it highly depends on the duration of the exposure of the sensor array to the analytes; at least 5 seconds are needed in the case of noise free data, and at least 10 seconds when Gaussian noise is added. This could be a problem in real-time detection and noisy environments or in case where more analytes need to be distinguished.

2.3.3 Sampling and measurement challenges related to mobile robotics applications

The last two subsections provide an insight into current research in chemical detection, which has been extensive. However, the majority of the systems that have been developed are aimed at static and mostly lab based identification of chemicals. What both the instrumental and e-nose reviewed work have in common is that they deal with identifying or classifying specific chemicals from a dataset that contains samples of various chemicals that have been collected under identical conditions, in a controlled lab environment and

under an explicit data collection/measurement protocol. The measurement protocol often includes three phases: the exposure of the sensor array to a reference gas, then the preparation and delivery of an analyte to the sensors, and then cleaning the sensor with fresh air. This protocol can improve the recognition of chemicals, but it limits the use mainly to offline analysis approaches, where the sample can be taken to the lab. In robotics applications, such approaches are limited; an example of controlled chemical detection in robotics is [20], where essentially, the robot is viewed as a vehicle to bring the instrument to a specific target sample, instead of bringing the sample to the instrument, without considering continuous monitoring or real-time requirements (the detector had to be aimed to a specific target and the measurement lasted 10 minutes). However, real-time, weight and power restrictions, makes this sample handling approach infeasible for application on a mobile robot [16]. In addition to that, the fact that the features that are used for the classification of chemicals, are not extracted from a continuous time-series, but from a sample corresponding to a controlled, well defined measurement, hinders the application of this work in online detection scenarios.

2.3.4 Chemical sensing for mobile robots

Until recently, artificial olfaction as part of robotic applications had not been addressed to a great extent, partly because of technical limitations. However, as sensing technology is being developed, chemical sensing robots become possible, by integrating a robotic system equipped with suitable sensors and software strategies for chemical detection and other related tasks [37]. As Lilienthal et al. note in their review [16], most of the mobile-robot chemical sensing systems deal with the problem of chemical source localisation, which can be decomposed to three subtasks: gas finding, gas source tracking, and gas source declaration. Although the majority of publications deal with gas source tracking, the other two tasks are equally important, not only as part of gas source localisation, but in their own right as well. Gas finding, which is of interest in this work, is needed when the presence of a chemical of interest has to be detected, either to initiate the tracking or in a stand-alone surveillance/monitoring application.

Lilienthal et al. [16] claim that gas finding can be reduced to a suitable exploration strategy and then setting a threshold for the detection of the desired concentration of a target chemical. Ishida et al. [37] notes that the detection of the chemical plume is straightforward, because the chemical sensors of the robot do not respond until they come into

contact with the chemical to be detected/traced. However, the underlying assumptions behind these arguments are: uncluttered, controlled environments, well defined classes of chemicals, and highly sensitive and selective sensors. In reality, sensors are cross selective to multiple agents -as discussed in section 2.2. Some of these agents might be part of the background, or there may be other agents different than the target analyte in the area. Therefore, more sophisticated techniques for detecting chemicals of interest are needed, like the ones presented in the previous section. The detection strategy needs to account for changing environmental conditions, noise, multiple possible targets, unknown chemicals of interest. Factors to be taken into account when designing a robotic chemical detection system include the integration of chemical sensing systems to mobile platforms, ensuring high sensitivity, quick and robust response and the ability to detect a number of different chemicals [16]. Also, for in situ, real-time detection the data coming from the sensors are a time series, and there are some data streaming issues to be considered as well. The main are: handling continuous flow, online processing, limited memory usage, restricted processing time, change detection and minimising energy consumption for embedded applications [38] [39].

Relevant work

Vembu et al. in [40] present an SVM approach for identifying and locating chemicals, which uses domain specific kernels that exploit the temporal dependence of time series. Every sample is expressed as an 8 dimensional timeseries, obtained from a controlled three-phase measurement from 8 MOX sensors. Three groups of SVMs are examined: the first is the baseline that uses no temporal information, but treats the whole timeseries as a single feature vector. The second group, first fits the timeseries in a dynamic model and uses the features of the model as the input of the SVM kernel. The third group is the time series kernels, which use the error made by a model on two samples over different model parameters, as input to the kernels of the SVM. A multiclass SVM is trained using the one-vs-one strategy. The best performance is found with the time series kernels (group3), which are computationally more expensive but have very good performance and it is argued that powerful classifiers can be trained given large amounts of data. In fact the dataset they have used is extensive and has been collected over a three month period (684 measurements). However, the existence of large amounts of data for every chemical that can be of interest in a field application can be challenging. Moreover, the training pro-

cess and parameter selection (hyper-parameters of both the SVM and the feature/kernel algorithm) can be time consuming, as some of the parameters have to be fine-tuned and others determined by cross-validation. If re-training is required, due to some environmental change, or addition of new classes, this could be challenging, as a new large enough training set cannot be collected on the spot and training would take too long to do online. Even though the temporal factor has been considered, every sample is measured in a controlled setup with the minimum time of exposure of the sensor being two minutes. The problem with this, is that it limits the plausibility of continuous monitoring deployment, as this does not correspond to continuous, unstructured measurement situations. Even if a real-time system performs detection by alternating sampling - identification cycles, a 2 minute time series sample is arguably too long for real-time detection.

An attempt to address continuous and real-time applications is presented in [41]: it is argued that even though good results have been achieved with controlled sampling approaches, these are of limited use in cases where continuous monitoring and sampling the environment over extended periods of time is needed (for instance pollution monitoring, or industrial process monitoring); this is a problem to which little attention has been given. In their work a sensor array mounted on the robot, is directly exposed to the environment and analysis is based only on the transient information in the signals instead of the steady state of controlled sensor measurements. The difference between the two is illustrated in figure 2.1. In order to process the continuous signal, s, a segmentation methodology is proposed, where every segment is identified, using the first derivative (ds/dt), as one of three phases: baseline, rise and decay phase (segmentation is also shown in figure 2.1). The baseline phase is used for signal correction, while the rising and decay phases (sub-timeseries) are used as inputs for the classification. The input timeseries / transient phases undergo feature extraction and then they are used to train an SVM and an RVM (Relevance Vector Machine). The RVM is functionally identical to the SVM, but provides probabilistic classification. This allows for the association of a confidence measure with a classification, which is important because of the possible sub-optimality of the segmentation step, but also because it allows for a rejection class to be implemented for samples with no definitive identity. Both the SVM and RVM are evaluated for all combination of feature extraction, baseline correction and data normalisation methods. In general the SVM performs better that the RVM (classification rate of up to 94.3% vs 84.7%). Using the rejection threshold decreases the classification error rate of the RVM

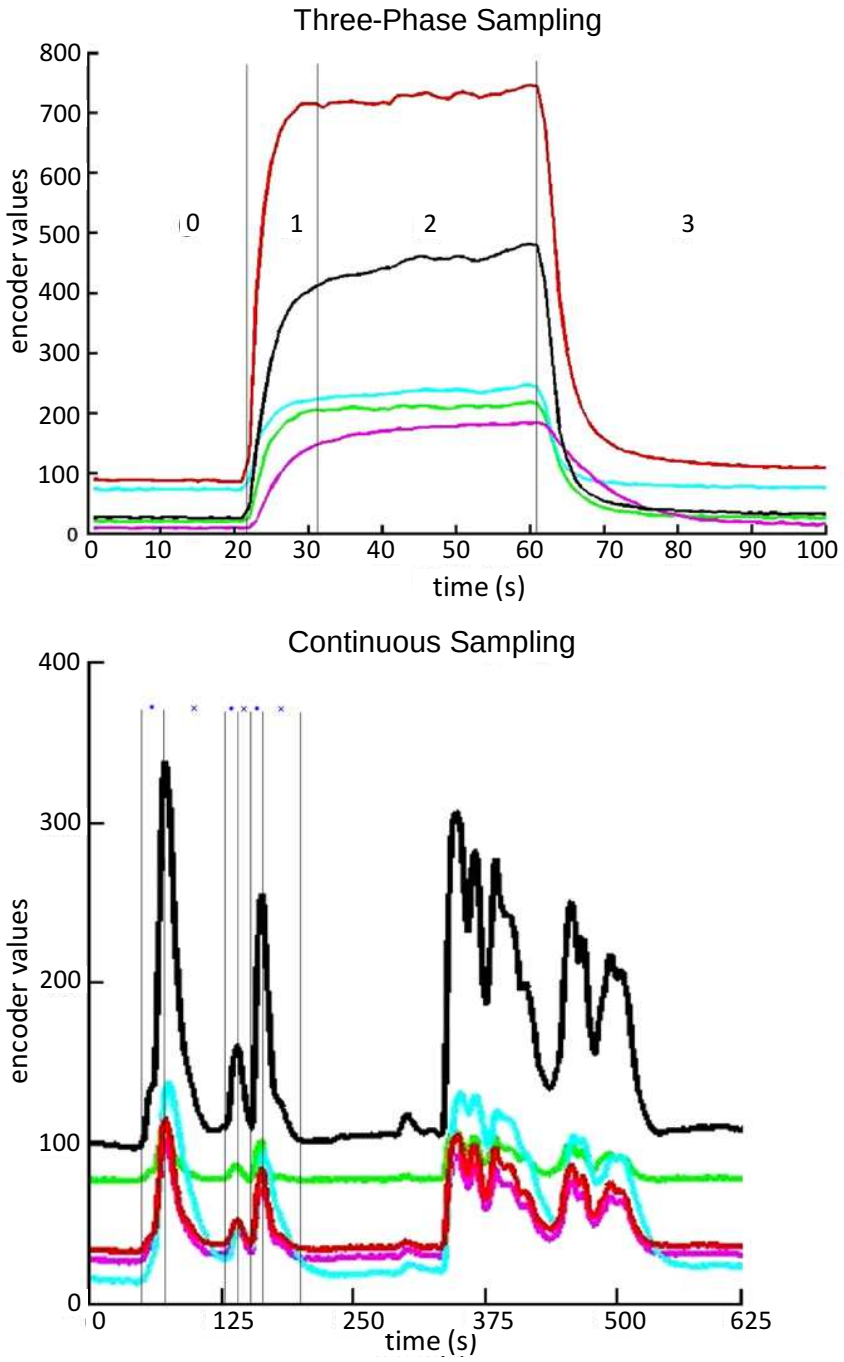


Figure 2.1: Example of signal collected using the three phase sampling process versus continuous sensing, adapted from [41]. In both figures the response from five MOX gas sensors is visible. Upper: example of signal collected using the traditional three phases sampling process: 0 baseline, 1 transient, 2 steady state, 3 recovery. Lower: example of signal collected continuously with the mobile robot in an uncontrolled environment. At the start of the signal the segmentation phase is noted: asterisks correspond to parts of the signal that have been segmented as rise phase, while diagonal crosses indicate the recovery phases.

and increases its performance to SVM levels. The RVM has the added advantage of faster prediction because it uses a sparser representation compared to the SVM. This is a good

step towards continuous monitoring, however, if the whole rise or decay phase, or both, is needed for classification, obtaining the whole segment can take several seconds, which is a drawback for real-time detection.

2.3.5 Limitations

The research presented in this section deals with detecting a number of chemicals, typically from data collected under a lab controlled protocol. A variety of methodologies for the recognition of chemicals has been reviewed. Following the requirements set by Shaffer, they vary on the accuracy, and speed requirement. However, a lot of them face a few of the same challenges. Pattern recognition algorithms which are typically used for chemical detection, are often challenged, when real-world applications are considered, in a few aspects:

- The algorithms are usually limited to a small number of classes/chemicals that can be detected and the addition of a large number of classes can be problematic. In some network approaches, the increased output class space can be challenged both by memory restrictions and complexity of the prediction phase. In SVM approaches, multiple SVMs have to be trained to separate all classes, either on an one-vs-one basis or an one-vs-all basis. Additionally, as more classes are added, more training data is needed. The acquisition of sufficient amounts of data for all classes may be complex and time consuming.

- When a chemical detection system is deployed in an unknown environment there can be chemicals other than those included in the training library, that might be of interest. As only specific chemicals are included in the training, there is the risk that analytes of interest can be misclassified or missed. Additionally, the signatures of known chemicals can change or appear different than the library entries, either because they are masked (intentionally or not) by other compounds, or because of sensor imperfections or changes in the environment (e.g. different temperature, humidity levels, addition of new background chemicals - more on this is discussed on Chapter 3).

- It is hard to expand the library of known compounds and add chemicals online, as the training is time consuming and on occasion complex - not meeting the *simple to train* requirement set by [17].

- A common limitation, which has been widely acknowledged, is that the detection of chemicals needs to be extended to cases where there are other chemicals in the background. In this case a target chemical needs to be distinguished from a multitude of analytes which are considered "normal", or of no particular interest.

Some of these challenges can be at least partly addressed if the problem is treated as an anomaly detection problem. Concepts associated with anomaly detection, the basic approaches for anomaly detection and relevant to chemical detection work will be reviewed in the next subsection.

2.4 Chemical sensing as anomaly detection

Anomaly detection aims at finding observations that are different in relatively homogeneous, large amounts of data. Anomalies are defined inside the context of the normal class, so assuming that the detection algorithm is trained on sufficient "normal" data, it will be able to detect any number of chemicals that cause a sensor response that deviates from the normal chemicals that exist in a specific environment. In a dynamic unknown environment, it is argued that a chemical agent detection system, should primarily detect anomalies and as a second, post-processing step, classify or recognise them.

An anomaly can be defined as an instance in a dataset that significantly deviates from the majority of the data, which can be considered to be normal, "a pattern in data that does not conform to a well defined notion of normal behaviour" [42]. Generically put, in a system an anomaly is any kind of event that is not expected and is inconsistent with the normal and known behaviour of the system. The problem of anomaly detection can be concisely expressed in the following: given a system that can be described by a series of vectors, X_i, at a given instance i, with n features or attributes $S = \{X_i | X_i = \{x_{i,1}, x_{i,2}, ..., x_{i,n}\}\}$, anomaly detection research aims at establishing a consistent method that will detect any instance $X_{an} = \{x_{an,1}, x_{an,2}, ..., x_{an,n}\}$ that cannot be considered a normal or expected expression of the system.

Anomaly detection is also often referred to as novelty detection, outlier detection, deviation detection, exception mining or even more domain specific names as fault detection and misuse detection, which are all considered to be fundamentally similar concepts [43]. Even though the terms novelty detection and outlier detection are often used interchangeably to anomaly detection, these are not always considered to be completely identical

tasks. The main difference is in the post-processing of the anomalies, novel events or outliers; i.e. the strategy concerning how the events or anomalies are to be handled once they are detected. An anomaly can be recognised, classified or retained with an appropriate label. An outlier can be expunged from future processing. Novelty detection usually aims to incorporate the novel patterns into the normal model [42]. In any case, the tasks are very closely related, since the core problem is the detection of an anomaly/novelty/outlier, and they will be studied here under the same context.

Detecting anomalies entails the identification and modelling of a bounded region of normal behaviour and classifying observations that deviate from that model as anomalous. This task can be challenged in a number of ways, some of which are discussed by Chandola et al. in [42]:

- The boundary between normal and anomalous behaviour is rarely precise in realistic situations
- Malicious activity can make an anomaly seem normal

But even if a clear boundary between normal and anomalous behaviour is assumed to exist:

- The normal model is subject to constant change and evolution
- A normal model is hard to construct due to the limited availability of training and validation data
- There are domain specific requirements and limitations
- Noise can affect the classification decisions

There are many different formalisations and definitions of anomaly detection, and related concepts. For instance Chandola et al. classify anomaly detection approaches into 6 general categories: Classification-based, Nearest neighbour-based, Clustering-based, Statistical, Information theoretic and Spectral. A good review of anomaly detection work is also presented by Markou and Singh in [44] [45], where the problem of novelty detection is treated as a requirement for a classification or identification scheme, since a machine learning system can never be trained on all possible target classes and objects that it might encounter on runtime. In their review anomaly detection approaches are classified under two categories: statistical or neural approaches.

In this review a more high level presentation is followed, categorising the anomaly detection approaches into three fundamental categories/types, as outlined in [43]. This taxonomy is also illustrated in figure 2.2.

Type 1 approaches determine the outliers with **no prior knowledge of the data**, typically through unsupervised clustering. The data is organised into one or more clusters and an observation is classified as an outlier if separated from the formed clusters. These techniques usually treat the data as a static distribution and require all the data to be available before processing. However, it is argued that "once the system has a sufficiently large database with good coverage, then it can compare new items with the existing data" [43].

Type 2 approaches **model both normality and abnormality** (typically supervised classification) and require labelled data of both classes, normal and anomalous. Some of the problems with type 2 approaches are that the data is typically considered static and the classifier needs to be retrained if the distribution shifts (unless an incremental classifier is used). Moreover, the training data should cover a broad enough space to allow for good generalisation.

Type 3 approaches typically **model normality only** and are analogous to semi-supervised recognition or detection, as the detector is trained on normal data but needs to recognise anomalous ones. This class of techniques is commonly referred to as novelty detection. It is suitable both for static and dynamic distributions and can be implemented in an incremental way. However, for successful generalisation, the full gamut of normality needs to be available for training.

Several different strategies have been developed for detecting anomalies, some of which will be reviewed in the following subsections. Existing anomaly detection work from all the categories will be reviewed, not restricted to chemical sensing, as the research that treats it explicitly as an anomaly detection (and not identification) problem is not extensive. Instead the focus will be anomaly detection methodologies, and relevant work from chemical detection will be discussed where appropriate.

2.4.1 Anomaly detection evaluation

Before reviewing some of the anomaly detection methodologies, it is worth presenting some of the metrics commonly used to assess anomaly detection algorithms. An anomaly

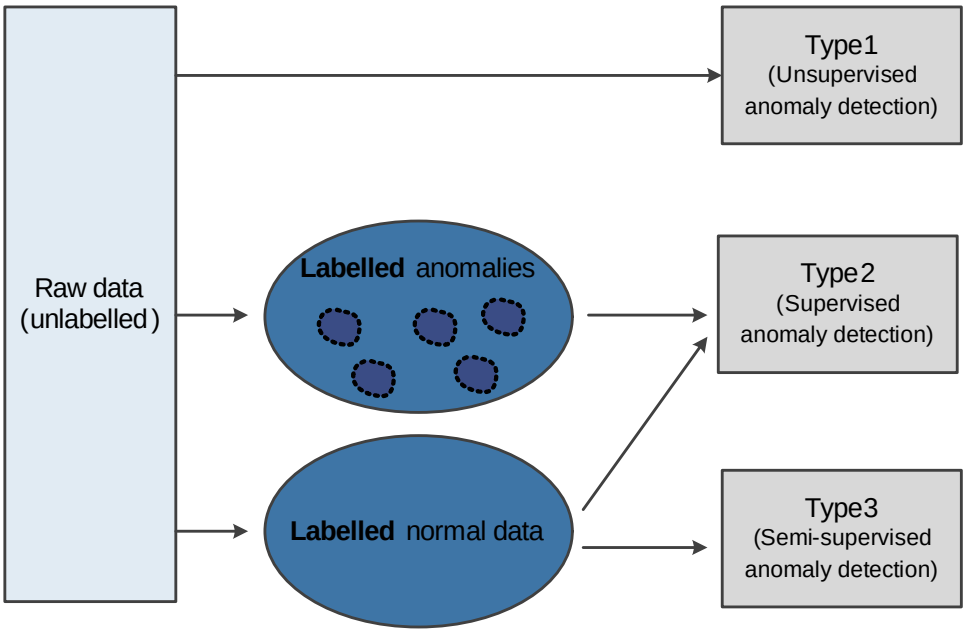


Figure 2.2: Types of anomaly detection systems. Type 1 performs some form of unsupervised clustering on the raw sensor data. Type 2 is supervised training, typically of a classifier that is trained using labelled data from all the anomaly classes, as well as normal training data. Type 3 is semi-supervised, as only normal data is used to train a model of normality and declare samples that deviate from that as anomalies.

detection algorithm is susceptible to two types of errors:

- Type I error: this error occurs when an input vector is declared anomalous, when in fact it reflects normal behaviour. This type of error is commonly referred to as a false positive, false detection, or false alarm.

- Type II error: this error occurs when a input vector that reflects abnormal activity is falsely classified as normal. This type of error is referred to as false negative or absence of alarm.

In detection and classification problems, the confusion matrix is a visual and concise way to represent the correctly detected/classified items, in relation to the misclassified ones. For the purposes of this, anomaly detection can be viewed as binary classification, i.e. the classification of examples into positive (anomaly) or negative (normal). For binary classification the confusion matrix is given in figure 2.3.

The confusion matrix summarises the output of a detection algorithm that classifies a set of data points into positive (anomalous) or negative (normal). In anomaly detection, a false positive (FP) is a normal instance that is falsely detected as an anomaly and a false negative (FN) is an anomaly that has been classified as normal, i.e. it has been missed.

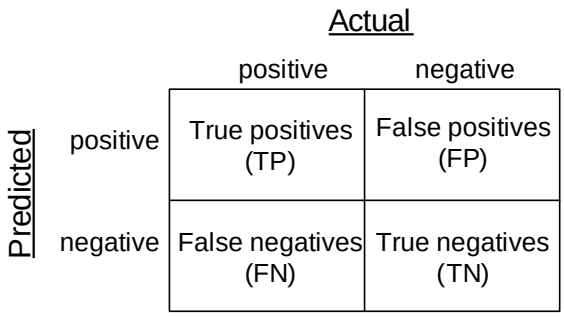


Figure 2.3: Confusion matrix for the binary classification case.

True positives (TP) and true negatives (TN) are data points that have been correctly classified as anomalies, or normal instances respectively. The goal is to maximise the true positives, while false positives and false negatives are kept to minimum. Metrics related to these concepts and typically used are the following:

$$TP\ rate = \frac{TP}{TP + FN} \tag{2.1}$$

$$FP\ rate = \frac{FP}{FP + TP} \tag{2.2}$$

$$FN\ rate = \frac{FN}{FN + TP} \tag{2.3}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.4}$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}, \tag{2.5}$$

where MCC is the Matthews Correlation Coefficient. In anomaly detection problems, the number of negative, i.e. normal, examples in a dataset, or a time series is usually significantly higher than the positive examples and the expected values of FP and FN. A measure that can be more informative in that case is the precision which quantifies how many of the detected anomalies are true anomalies (2.6). The detection rate can also be used, which is the number of detected anomalies over the total number of labelled (positive) examples (2.7).

$$Precision = \frac{TP}{FP + TP} \tag{2.6}$$

$$Detection\ rate = \frac{TP}{TP + FN} \tag{2.7}$$

For the case of classification, e.g. when a detected anomaly is classified in one of N available classes, the classification matrix is an NxN matrix. The element c_{ij} represents

how many data points that actually belonged in class j have been classified into class i. The ideal classifier would produce a confusion matrix with non-zero elements only in the diagonal, c_{ii}. The classification accuracy is the number of correctly classified data points (the sum if the elements in the diagonal) over the total number of labelled data points.

2.4.2 Anomaly detection methods

A lot of diverse approaches have been developed for anomaly detection. Here a few key approaches are reviewed from the three categories outlined earlier (section 2.4), including but not limited to chemical detection applications.

Type 1 - no prior knowledge of the data

This family of approaches does not use labelled information about the data. It classifies and clusters the data, considering examples that are not sufficiently represented, as anomalies.

Using an auto-associative ANN paradigm, Crook et al. [46] choose a Hopfield network to perform online novelty detection on a mobile robot application. This selection is based on the observation that the calculation of the energy of a Hopfield network is very similar to the calculations made in a spiking neural network model, which actually replicates the novelty detection neurons. The Hopfield auto-associative network is a fully connected recurrent neural network that achieves familiarity discrimination (the term here is used similarly to novelty detection) by comparing the energy of the network when presented with a pattern against some threshold. Generally known (familiar) patterns give low energy, while novel patterns cause high energy. The experimental set-up consists of a robot equipped with a camera, which has to learn its environment, in this case a gallery of pictures, through a Hopfield network and detect novel elements that are added to the gallery. Learning and detection phases are alternating and new elements are added at every cycle. The goal is to determine whether the novel pattern can be detected and whether it can then be learnt. The Hopfield network detector is indeed shown to be able to detect novel elements and incorporate them in the learnt model in only one cycle. However, the main problem with this is that it is not as sensitive as other approaches, and in fact the more patterns that are learnt by the Hopfield network, the bigger the change that is necessary in order to considered a pattern novel.

An unsupervised anomaly detection approach using Artificial Immune Systems (AIS)

is presented in [47] to address the problem of not having correctly labelled or purely normal data available. Here an evolutionary artificial immune network algorithm, aiNet, proposed by De Castro [48] is used, which simulates the immune network response to antigenic stimulus. The main idea is to use an evolutionary artificial immune network to compress unlabelled training data to a network of detectors and then perform clustering analysis on this obtained network. The latter entails identifying the clusters on the formed network. New data is classified based on these clusters: if the new point is close enough to the heavily populated, "normal" clusters, it is considered normal, otherwise it is an anomaly. The algorithm is used on an intrusion detection problem (computer network security), which is a very commonly addressed anomaly detection task. The parameters are set experimentally and the algorithm's performance is evaluated with respect to the detection rate and the false positives rate; it is found to outperform existing approaches, as the detectinaon rate is higher for similar false positive rates and vice versa and the detection of unknown threats is also higher. However, the algorithm has a lot of parameters and thresholds that have to be finely tuned or predefined, which makes it difficult to use in different problems or data. Moreover, if the background distribution, i.e. the network behaviour that is considered normal, changes, the network would have to be retrained.

A chemical sensing anomaly detection approach is presented in [49]. The data coming from a gas chromatography ion mobility spectrometer (GC-IMS) have two parameters, the GC retention time and the IMS drift time, which allows them to be cast as an image and be analysed with image processing tools. Statistical signal processing is used to enhance the image and the Reed Xiaoli Detector (RXD) from the domain of hyperspectral imaging is applied to extract unknown targets that are spectrally distinct from their background (Mahalanobis distance is used to this end). However, each sample is collected over a 60 second period, which limits the possibilities for using such an approach in real-time applications.

Type 2 - modelling both the normal and the anomalous class

A two class classifier, the SVM, is commonly used for anomaly detection. A Support Vector Machine (SVM) typically operates in two class classification problems and tries to identify a boundary that separates the two classes. The desired boundary is a hyperplane, defined by a subset of the training data, the support vectors, that maximises the margin between the two classes. In [50] SVMs are used for intrusion detection, in the domain

of computer security, outperforming ANNs. The disadvantage of this approach is that a labelled set of both normal and anomalous behaviour is needed, which is not straightforward to obtain.

The importance of training data is also demonstrated in [51]. In this comparative study on the application of ANNs for anomaly detection, the Multi Layer Perceptron (MLP) ANN, a very popular supervised ANN algorithm that learns an input-output mapping, is tested under the two class classification paradigm. When data both for the normal and the anomalous class is available then the MLP is trained as a non-linear classifier. Barreto et al. [51] find that the performance of the two-class MLP-ANN is sensitive to the frequency of the examples from each class. The best performance is achieved when the two classes are represented equally. However, this contradicts a basic assumption of anomaly detection, that anomalies are rare; examples that correspond to anomalies can be rare or even non-existent [51]. Moreover, both for the SVMs and ANNs, if either the type of the attacks change, or the normal behaviour changes, a new hyperplane separating the normal and abnormal class would need to be generated, requiring a new set of labelled training data.

This problem of training data corresponding to anomalies can be partially addressed by generating artificial negative examples, a negative example in this context being an anomalous example. In [52] ideas from immunology are used in combination with classification algorithms. The immune-inspired Negative Selection algorithm [53] is used for self-non-self discrimination; Starting from a set of binary vectors representing the normal state of a system (self), a set of detectors are created that match the normal vectors as little as possible. These are considered to represent the anomalous state of the system (non-self). Using this algorithm the problem of anomaly detection where only data representing the normal state is available, can be treated as a classification problem, since artificial training examples of the abnormal class have been created. In [52] this strategy for generating negative examples is combined with two classification algorithms, an MLP-ANN and a fuzzy rule classifier evolution algorithm. Although this strategy is effective, it is acknowledged that the real valued representation of the data makes the calculation of the number of detectors that have to be generated more difficult. This neuro-immune approach is compared to a SOM, trained on normal data only, in [54], on time series data. The normal training set is created by extracting 4-dimensional vectors from the time-series using a sliding window. The two approaches are found equally effective in detecting an

anomalous segment introduced in the time-series, using the detection rate and false positive rate as comparison measures. The advantages of the neuro-immune method is the classifier constructed is less sensitive to the detection threshold, in contrast to the SOM that needs fine tuning and that the network is generally less complex than the SOM. However, it is found that both methods benefit from more complex representations, i.e. higher number of neurons. This can be a serious limitation, especially when the dimensionality increases (here 4-dimensional vectors are used), as the search space grows and so do the number of detectors necessary to cover the non-self space.

Multiclass approaches can also be used for anomaly detection, by including a number of anomaly classes as well as the normal class. In a problem of chemical detection, pattern recognition algorithms are compared in [55]. The algorithms are explicitly trained for 3 chemical agents - types of anomalies, so this still suffers form the limitation of the identification approaches. A Bayesian classifier is found to have the best results, against a decision tree and an MLP neural network, with a perfect detection rate, but a very high rate of false positives (24%).

An array of chemical sensors together with laser scanner data and video event analysis is used in [56] to assist in the detection in a surveillance system of a person carrying possibly hazardous material. Video event analysis consists of storing a number of events in a forest of graphs and matching the event to be analysed against this knowledge base of events. This alarm level is fused with the output of an array of QMB chemical sensors and persons of interest are detected and associated with one of three alarm levels (normal, suspicious and critical), using the PMHT-c (probabilistic multiple hypothesis tracking) algorithm. However this is limited to a known base of anomalous events.

Type 3 - modelling the normal class only

Representing fully the anomalous class (or classes) is challenging, since the frequency of anomalous examples is by definition much lower than the examples for the normal class. An alternative is to model only the normal class and denote as anomalies observations that do not conform with this model. Barreto et al in [51] examine the use of ANNs, under the single class classification paradigm: if data that represent only expected (normal) behaviour is available, a representation of the normal behaviour is built and a new pattern must be classified as either normal or anomalous (single-class case). An issue that is addressed in [51] is how standard neural network architectures can be used for

novelty detection. The Self-Organising Map (SOM) [57] is an unsupervised neural network that projects complex non-linear relationships between high-dimensional data onto a lower dimension space. When used for novelty detection the SOM is trained on data representing the normal behaviour of the system. When presented with a new input, the winning neuron, or the winning weight vector is the weight vector that matches the input more closely. The error between this winning weight vector and the input vector is used for detection; if the error is higher than a threshold, then the representation built by the SOM is considered to not describe the input vector, and hence it is considered novel, or an anomaly. The MLP-ANN can also be used in the single class case, in an auto-associative architecture: it is trained to produce an identity mapping of the input to the output, i.e. to learn a generalised representation of the input [58]. When presented with an abnormal pattern the reconstruction of it (using this learnt identity mapping) should be poor, hence the reconstruction error is thresholded and used to detect an anomaly. Variants of these two networks are compared under a unified threshold setting strategy, on a biomedical application (breast cancer data set). The results indicate that it is advantageous to use ANNs under a single-class classification paradigm, trained using only normal data, than using them as 2-class classifiers.

A statistical method is presented in [59] which makes use of prior domain knowledge: the feature space is split into two sets of attributes: the indicator attributes, which are the ones directly indicative of an anomaly and the environmental attributes. In conditional anomaly detection (CAD), the indicator attributes are used to detect an anomaly, but they are conditioned on the environmental attributes values. Qualitatively, an anomaly is an observation whose "indicator values are not in keeping with its environmental attributes" [59]. The model, f_{CAD}, consists of a Gaussian Mixture model (GMM) for the indicator attributes, a GMM for the environmental attributes and a mapping function between the two. Maximum likelihood estimation coupled with the expectation maximization (EM) methodology is used to determine the parameters of the models. A data record is considered an anomaly if the probability density at this data point is less than the median probability density at all of the test data points. The GMM-CAD full algorithm (one EM algorithm estimates the parameters of the two GMMs and another EM these of the mapping function) is found superior to other variants of the approach, as well as to a 5th-NN and a LOF (Local Outlier Factor) algorithm, especially for data with high dimensionality. This is an interesting approach with good results, however it includes complex

models, which are computationally expensive, especially if they need to be recalculated in an online application. The algorithm also relies on a successful partitioning into environmental and indicator attributes, which requires domain knowledge and evaluation of the data semantics, and cannot be applied in all classes of problems.

A statistical anomaly detector for contaminants on a water distribution system is presented in [60]. This is a parametric approach, as it assumes that the chemical sensory data coming from a multi-sensor instrument come from a normal distribution. The properties measured include pH, conductivity, chlorine residual. The normal model/baseline quality is constructed over an extended period of time, plus additional measurements of 100 minutes prior to the introduction of a contaminant. The limit of detection is set to three standard deviations; anything that falls outside this limit represents an anomaly. Although it is proved that, for the specific contaminants used for the experiments, even if the distribution is not normal, it is safe to state that an instance outside the three sigma region represents an anomaly, such an assumption cannot easily be generalised. Also, including recent measurements in the model means updating the model continuously, which can add memory and computational burden to the system.

2.4.3 Limitations

It is well known that no one strategy is globally the best for all anomaly detection tasks [42] [43]; the selection of a suitable approach for a problem depends on a number of factors which include but are not limited to: the nature and dimensionality of the dataset, the availability of labelled training data, the distribution model and whether valid assumptions can be made about it, the speed and computational complexity of training and testing the detector, scalability requirements and incremental learning capabilities.

Earlier sections (sections 2.3, 2.3.4) have outlined certain requirements for chemical detection systems for mobile robots. The most important of these are high accuracy, speed, online detection, simple to train, low memory and computation requirements. Network-based approaches become increasingly complex as the number of dimensions increases, which has implications both for their training (slower and requiring more data) and their prediction time, as propagating an input vector through a complex network can be costly. Dimensionality can also affect auto-associative algorithms, which rely on a compressed representation of the input space. In chemical sensing this limits these approaches to small sensor arrays only. There are cases where spectrometers are preferable, as “the

number of identifiable patterns from a sensor array is limited by the number of different gas sensors" [61], or for detecting specific class of chemicals like high-vapor pressure explosives [62]. Finally, especially in applications involving unknown, dynamic environments, it is important that the detection algorithms do not rely on an exhaustive representation of the full gamut of normal and abnormal inputs, like the two-class/multi-class approaches and that assumptions about the input data distributions (parametric models) are avoided.

Some of these limitations are addressed by the Receptor Density Algorithm (RDA) [4], a bio-inspired, non parametric anomaly detection algorithm, that has been shown to be a fast and accurate method to detect anomalies in high dimensional, mass spectrometry data. The RDA has the added advantage of extracting clean noise free signature of the anomalies, which can then used for identification, if appropriate. Details on the principles behind the algorithm and its existing application on chemical detection problems is reviewed in the following section.

2.5 RDA - immune inspired anomaly detection

The Receptor Density Algorithm (RDA) is a bio-inspired algorithm that aims at discovering anomalies embedded in noisy data and extracting clean, noise-free signatures which can be used for recognition of these anomalies. This algorithm is of great interest, as it has proven effective in the task of chemical detection in spectrometry data.

2.5.1 Biological principles

The RDA is inspired by the T Lymphocytes (T cells) of the immune system and their discriminating capabilities of peptide bound Major Histocompatibility Complex (pMHC) molecules on Antigen Presenting Cells (APCs), through their T Cell Receptors (TCRs) [63]. Of the pMHC expressed on an APC, 99.9-99.99% is abundant self-pMHC and the other 0.01-0.1% is non-self pMHC. The discrimination capabilities of the TCR between self and non-slef pMHC stem from the complex information processing pathways and signalling mechanisms. The interested reader is referred to [64] and [4] for a detailed analysis of the mechanisms and processes that have been modelled. This section focuses on the extracted computational model and its applications.

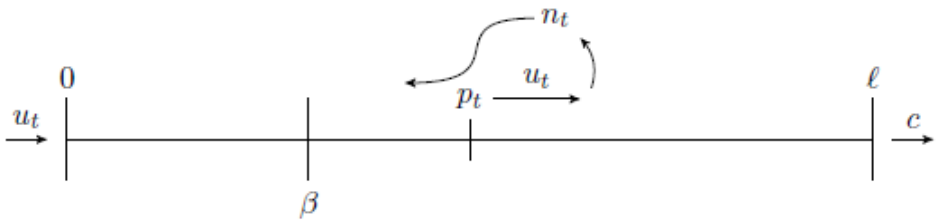


Figure 2.4: A single receptor, from [64]. A receptor receives input u_t which advances the position of the receptor p_t. When the p_t exceed the negative feedback threshold, β, negative feedback n_t is generated, which suppresses the position. If the position exceeds the detection threshold l, then a classification signal of 1 is generated, which signifies an anomaly.

2.5.2 The computational model - RDA

The RDA abstracts from the T-cell signalling mechanisms and simplifies the biological processes. The function of the algorithm is based on the notion of the receptor, whose diagrammatic representation is given in figure 2.4. Each receptor at time t is described by its position p_t, the negative feedback n_t and the thresholds β and l. The receptor receives input u_t, and advances its position p_t. If the position exceeds the β threshold, $p_t \geq \beta$, negative feedback is generated, which regulates the position and reverses its progression. Both the position and the negative feedback decay in time. If the input is strong enough to push the position over l, then a classification signal $c = 1$ is produced, and an anomaly is said to be detected. The equations summarizing these dynamics are equations 2.8 and 2.9.

$$p_{i,t} = bp_{i,t-1} + K_i(S_t) - \alpha n_{i,t-1} \tag{2.8}$$

$$n_{i,t} = dn_{i,t-1} + gH(p_{i,t-1} - \beta) \tag{2.9}$$

Given a multidimensional input at time t, S_t, a receptor is assigned to each position, or feature, i. The first equation 2.8 determines the position of the receptor at time t, $p_{i,t}$, in respect to the previous position $p_{i,t-1}$, the input S_t and the negative feedback $n_{i,t-1}$; b is the position decay rate, and $K_i(\cdot)$ is a kernel function that accounts for the interactions between neighbouring receptors, and serves smoothing purposes. The kernel is defined in equation 2.10, with $K_i(\cdot)$ a kernel function [1], h the kernel width and g_b a

[1] an example of a kernel function that can be used is the standard normal kernel $K(x) = (\sqrt{2\pi})^{-1} exp(-x^2/2)$

scaling parameter; α is the negative feedback efficacy.

$$K_i(S_t) = \frac{g_b}{hn} \sum_{j=0}^{w-1} s_j K(\frac{i-j}{h}) \tag{2.10}$$

In equation 2.9 d is the negative feedback decay rate and g the negative feedback growth rate; $H(\cdot)$ is the Heavyside step function: $H(x) = 1$ when $x > 0$ and $H(x) = 0$ otherwise, i.e. negative feedback is generated only when $p_{i,t-1}$ has progressed beyond β.

2.5.3 The RDA for chemical detection

The RDA is an anomaly detection algorithm that can find wide application, since it does not require any domain knowledge, with the exception perhaps of setting the parameters. In chemical agent detection, the input to the algorithm can be the readings of one or more sensors. If a sensor array is used, every sensor (which typically outputs a single value reading) will be associated with one receptor, as illustrated in figure 2.5. Similarly, if a spectrometer is used, then a receptor will be assigned to every channel/point of the spectrum. The RDA receives data continuously and declares an anomaly when a receptor produces an anomaly classification signal, i.e. $p_{i,t} > l$.

Time of anomaly

The time or duration of an anomaly is defined as the time interval $t_a = t_e - t_s$, where t_s is the first timestep when a receptor exceeds l, and t_e the first timestep when every receptor's position falls below l.

Signature of anomaly

$$\sigma(A) = (\sigma(A_0), \sigma(A_1), ..., \sigma(A_{w-1})) \tag{2.11}$$

$$\sigma(A_i) = \frac{1}{t_e - t_s} \sum_{t=t_e}^{t_s-1} (p_t - \beta H(p_t - \beta)) \tag{2.12}$$

The next step is to extract the signature of the detected anomaly. The signature of an anomaly $\sigma(A)$ is the distance above β of every receptor, averaged over the duration of the anomaly (equations 2.11, 2.12); w is the total number of receptors, aka the dimensionality of the input and $H(.)$ the Heavyside step function ($H(x) = 1$ if $x >= 0$ and $H(x) = 0$ if $x < 0$). The process of detection and signature extraction in a spectrum of 200 channels is depicted in figures 2.6.

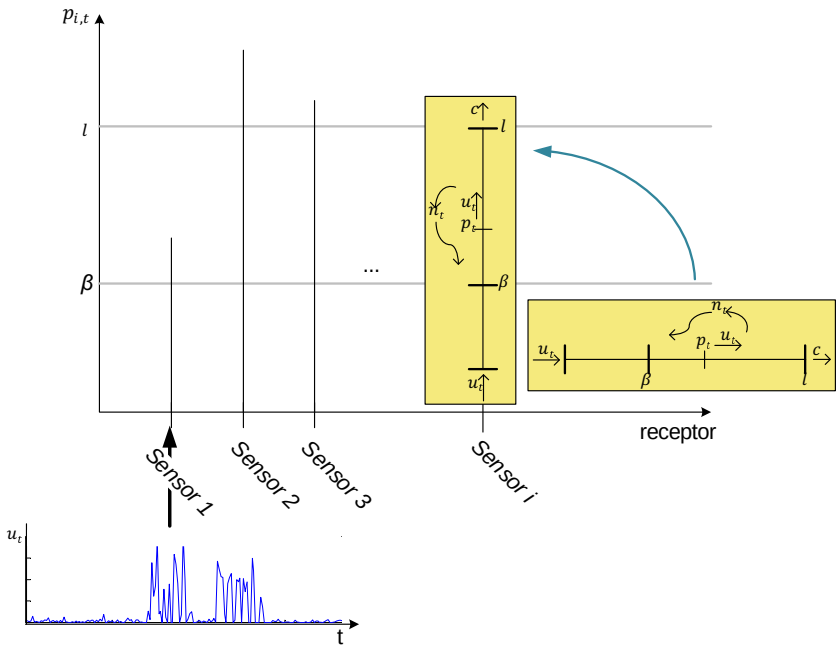


Figure 2.5: RDA, inverted multiple receptors. Given a multidimensional input, a receptor is assigned to each position, or feature. When sensor arrays are used, a receptor corresponds to one chemical sensor. Every sensor outputs a uni-dimensional timeseries, which is the input u_t for the corresponding receptor. An inverted visualisation is employed here for clarity reasons. Similarly, if a spectrometer is used, then every channel would correspond to one receptor.

Chemical identification

If provided with training data, or a library of known signatures, the RDA performs identification of the anomalies detected. A simple way to do that is through a matching function, like the one used in [64] [5].

$$\mu(A,B) = \frac{1}{\parallel \sigma(A) \parallel \parallel \sigma(B) \parallel} \sum_{i=0}^{w-1} \sigma_{A_i}\sigma_{B_i}, where \parallel \sigma(A) \parallel = (\sum_{i=0}^{w} \sigma_{A_i}^2)^{1/2} \tag{2.13}$$

The match between two anomalies A and B is given in equation 2.13, with σ_{A_i} defined in equation 2.12. This is compared to a threshold and if it is sufficient high A and B are identified as the same anomaly.

The operation of the RDA has two levels. The first is the detection of anomalies and the second level, which can be viewed as post-processing, is identification of the detected anomalies, by matching the extracted signature against possibly available library entries. A summary of the two levels of RDA output is presented in figure 2.7.

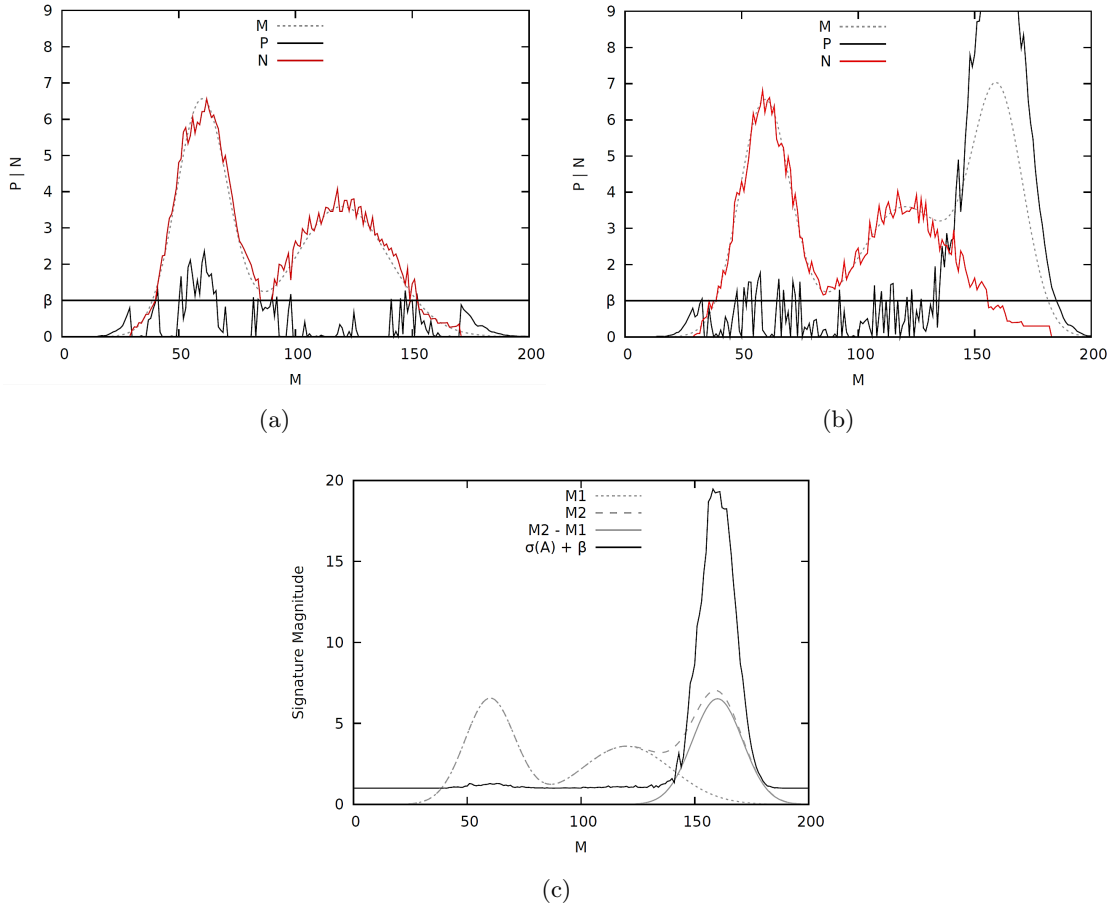


Figure 2.6: Anomaly detection and signature generation with the RDA, from [5]. This figure illustrates the response of the RDA to a model sensor M with 200 spectral channels. Every channel (on the x-axis) is assigned to one receptor. For all the channels/receptors the input u_t is depicted in dotted line, the position p_t is noted with a black line and the negative feedback n_t with a red line. Figure 2.6a is a still of a normal spectrum with no anomalies, at time t_0. Figure 2.6b is a still of the spectrum at time t_1, when an anomaly is present; note a new peak centred at the 160th channel. This anomaly is detected because of the large increase of p, which exceeds the detection threshold l (top of the graph). The signature of this anomaly, $\sigma(A)$ is given in the last part, 2.6c, averaged over the whole duration of the anomaly.

Application

The RDA has been applied on chemical data coming from a mass-spectrometer, winning the 2009 ICARIS competition with an anomaly detection rate of 86.5% and a false positive rate of 3.2%. The algorithm was also tested on a robot-mounted experiment and it was successful in detecting Deep-heat in a noisy environment, using data from an ion mobility spectrometer (IMS) [5]. One of the challenges of the RDA is the multiple parameters that have to be carefully tuned. Guidelines for parameter setting are given in [64], but still the exact values have to be determined through trial and error. This challenge was addressed

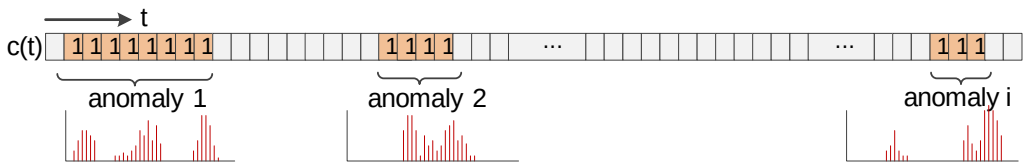


Figure 2.7: A minimal representation of the output of RDA. At every timestep (with every new reading), the RDA outputs a classification signal. The start of an anomaly is the first timepoint where the position of any receptor exceeds l, resulting in a classification signal of 1. The end of an anomaly is the first timepoint where every receptor has fallen below the detection threshold. Therefore, one anomaly takes place over a number of consecutive timepoints (duration). The first - anomaly detection - level of the RDA is the binary, classification signal output, indicating the presence or not of anomalies. For the second level - identification -, a signature is extracted for every anomaly and this signature can be used to recognise the anomaly.

in [65] using a genetic algorithm (GA) to evolve a set of parameters for the mass spectrometry problem. The detection performance improved with the evolved parameters, while a parallel implementation of the GA was proposed for faster parameter optimisation. Apart of the parameter setting, an important disadvantage of the algorithm is that it is static. There is no mechanism to keep the algorithm up to date with significant changes in the environment (i.e. in the background signature). Similarly in the case of sensor drift, a change in the response of the sensors when exposed to the same chemicals, it is possible that the signature matching part of the algorithm is going to fail, as the signatures of the same substance may appear different.

Nonetheless, the RDA has important advantages over other reviewed approaches. Firstly, it detects anomalies in a timeseries, without using whole segments to extract features but in an online manner; a decision is made with the acquisition of every new reading - point in the timeseries. Secondly, it is a very lightweight, fast algorithm, as it does not use complex representations or models of the input space and it only needs to update two equations (2.8, 2.9) at every timestep. These two features of the algorithm make it suitable for real-time detection of chemicals. The training of the RDA, which is essentially parameter setting (through evolution or manual tuning), requires a training set with anomalies, to make sure that the RDA is able to detect small anomalies but still not produce a lot of false positives. However, in contrast to typical multi-class approaches, as reviewed in 2.4.2, the anomaly class does not need full or equal representation of all the classes; only a timeseries with occasional anomalies is needed for tuning/evolving the parameters. If anomalies are to be identified, as a post processing step to detection, then

training data involving labelled anomalous classes is needed in order to create the library. The training process itself is relatively simple, as a small number of parameters need to be determined.

2.6 Summary and discussion

In this chapter a review of the domain of chemical agent detection has been presented, identifying key challenges and discussing some possible strategies that can be used to handle these challenges. It has been argued that the problem of chemical agent detection can be viewed primarily as an anomaly detection problem, and as an identification problem at a second level. By distinguishing between these two tasks, the system can deal with any unknown and unexpected anomalous chemical, without needing previous experience on that particular compound.

The work reviewed in this chapter shows that the problem is usually treated more as a chemical identification problem; The main trends in sensing technology and chemical identification, for a variety of possible sensors and analytical instruments (spectrometers) have been discussed in sections 2.2 and 2.3 respectively. Considering chemical sensing robotics applications, a lot of the chemical detection systems are challenged, as they do not address real-time, continuous applications: the chemical data is not treated as a timeseries, but as distinct samples, each corresponding to several second or minutes of measurements. When time-series are processed they are often segmented (again in multi-second segments) in order to extract feature vectors.

Another limitation comes from the fact that the majority of chemical detection research is aimed at static and mostly lab based detection: identifying or classifying specific chemicals from a dataset that contains samples of various chemicals that have been collected under identical conditions, in a controlled lab environment. Considering deployment in unknown dynamic environments, detecting only specific chemicals can be a problem, as the exact description / library entries of chemicals of interest might not be available when navigating an unknown environment. Additionally, a "known" chemical might not be matched to its library entry, if it appears different, due to noise, different environmental background or sensor imperfections.

This is why it is useful to treat the problem of chemical detection in dynamic, unknown environments primarily as an anomaly detection problem. An anomaly is defined as an observation that deviates from the normal expression of a system. This allows for

a chemical of interest to be defined in the context of a specific model of normality. In robot-mounted chemical detection, when navigating a specific environment, a chemical of interest would be a chemical compound that is quantitatively different than the normal chemical background of this specific environment. The basic concepts and principles of anomaly detection, as well as relevant work, have been reviewed in section 2.4. In section 2.5, the RDA, an immune inspired algorithm for anomaly detection is presented. The RDA has been previously successfully used for the detection of chemicals of interest (anomalies) in mass spectrometry data. Additionally, it extracts clean, noise-free signatures of the anomalies, which then can be used for their identification, provided that there is a library of signatures of known compounds.

The RDA is chosen as the anomaly detection unit for the work of this thesis, not only because it has shown high accuracy with low FPrate in previous work, but also because it has several advantages that make it suitable for use in a real-time robot-mounted detection scenario. It can detect anomalies in a timeseries, making a decision with the acquisition of every new reading (online) and it is lightweight and fast, which makes it suitable for online, real-time detection. The training of the RDA, which is essentially parameter setting (through evolution or manual tuning) is relatively simple and it does not require an exhaustive set of all possible anomalies, especially if the identification is not considered, and because the RDA has a small number of parameters to be estimated or optimised.

The last feature is especially important when the problem definition is expected to change in some way. This can happen either with the occurrence of new, previously unseen anomalies (or presented differently because of a drifting signal), or with a change in the normal background. This has been touched upon in this chapter, but it has not been discussed in detail. In a real-world application of robot mounted detection in dynamic changing environments, however, this is expected to happen. For instance when the robot moves to a different environment, the notion of what is normal changes and the performance of the anomaly detection algorithm is expected to deteriorate. Using the RDA as the chemical detection algorithm, the problem of deteriorating performance under changing environments and how it can be addressed will be investigated in the next chapters.

Chapter 3

Adaptivity

The work reviewed in the previous chapter, irrespectively of whether chemical detection is viewed as an anomaly detection or classification problem, is mostly static. A system is trained for a specific environment: the one in which the sensor measurements are collected. The algorithms are expected to perform chemical detection, assuming that the environment, the targets and the general sensing conditions remain reasonably the same through the system's lifetime. However, this is not a realistic assumption for a lot of chemical detection tasks, especially for tasks that are not lab based. The RDA [4], which is a powerful algorithm used for chemical detection, can cope with some change, as the signatures it produces are to some extent free of background noise. It can also tolerate noise through the negative feedback mechanism, which does not allow small background fluctuations to immediately raise an alarm. Nonetheless, in cases of significant changes associated with dynamic environments, the algorithm does not have a mechanism to adapt itself to the new conditions.

A typical chemical detection set-up would involve one or more sensing instruments, operating in an environment, continuously sensing, detecting and classifying anomalies when present. Excluding the rare case of a highly controlled and stable environment, in most real world applications there will be factors that constantly change, like the temperature, the humidity and the pressure. Additionally, new chemicals can gradually appear without necessarily being anomalous. In the absence of anomalies the environment in which a chemical detection system operates, is the normal background. It is reminded to the reader that a typical anomaly detection system tries to model the background distribution, the "normal case", using some technique, parametric, non parametric or heuristic. Anomalies are then detected based on this learned model of normality. When environmental factors

change, however, the background distribution, i.e. the model of normality, changes. This change is particularly important when a long term application or a mobile platform are considered. In these cases, new elements can be added to the background over time, or as a sensor platform moves through different locales.

Adaptation in the context of chemical sensing is also often associated with a degradation in the sensing instruments. The model of the environment (normality) and the anomalies are created based on the readings of the sensors used. Apart from the imperfections of the instruments, which can be a source of variability, there is a great challenge posed by the phenomenon of sensor drift. Sensor drift is the change in the sensor's response over time when it is exposed to the same chemical, a quantitative change in a characteristic that should be remaining constant [66]. This means that the whole model that has been created will change, and it may not be usable in the long term.

The problem of adapting to changing environments or conditions that degrade the performance of a system, is a prominent topic of research in the machine learning community [67]. In machine learning problems, the change in either the classification rules or the data generating distributions is called concept drift [68]. A system that is expected to operate under such conditions must have the ability to detect concept drift and/or adapt. The detection of concept drift and the adaptation of a system in order to mitigate the effects of concept drift will be the focus of this chapter.

3.1 Structure of this chapter

This chapter is structured as follows: in section 3.2 the problem of concept drift, relevant definitions, challenges and strategies for dealing with concept drift are discussed. In section 3.3 a popular model for dealing with concept drift, classifier ensembles, is presented along with relevant applications in adaptation. In section 3.4 relevant work on adaptive chemical detection, using single or multiple classifier systems is reviewed. Finally, in section 3.5 a summary of adaptivity issues discussed is presented.

3.2 The problem of concept drift

So far, the notion of change either of the data itself or the interpretation of the data has been qualitatively discussed. In machine learning, concept drift refers to a change in the class definitions (concepts) over time; an environment from which drifting class data is

obtained is often referred to as a non-stationary environment[1] [69]. Models built on old data are no longer consistent with the new data and regular updating of the models is necessary [68]. Often the term context is used interchangeably with concept, as a set of examples where the generating function is stationary, and concept drift is the transition between different contexts. This transition can be sudden, when something triggers an instantaneous change, or gradual, when the transition happens over a period of time. The latter is usually associated with long term applications and incremental change.

Tsymbal in [68] define real concept drift as a change in the target concept and virtual concept drift as a change in the underlying distribution. The application input data consists of a series of samples $X_t = \{x_{1,t}, x_{2,t}, x_{k,t}\}$, where k is the number of features. Each sample is associated with an output y_t (a label or a prediction). When a detector or classifier is trained, the mapping between the input and the output $X \rightarrow y$ is learnt. Under this notation, real concept drift is a change in the distribution of the output y given the input X. Virtual concept drift is a change in the distribution of the incoming data, X, itself [70]. Although definitions vary, Tsymbal [68] supports that virtual concept drift, a change in the data generating distribution, also affects the decision boundary, which means that in both cases the learnt mapping between the input and the output must be adapted to reflect the current state of the system. This is graphically shown in the two-dimension, two-class case in figure 3.1. Finally, Widmer et al. [9] hold that concept drift can be caused by changes in *hidden contexts*. A lot of applications are associated with a number of hidden contexts, which are not directly measured, but they affect the problem in some way (for instance seasonal effects). A change in these can affect to varying degrees the target concepts, producing the effect that is generally known as concept drift. For instance, in chemical detection the temperature or humidity, which may or may not be directly measured, could be considered as hidden contexts. They do not directly affect the notion of anomalies, but they affect the measurement process, thus changing the chemical sensing problem.

Learning non-stationary environments has received much less attention, even though it reflects a lot of real applications where data distributions are inherently non-stationary, such as spam, fraud or climate change detection [69].

[1] An environment does not necessarily have a literal geographical existence. It is considered as an abstract model which generates the data of the application.

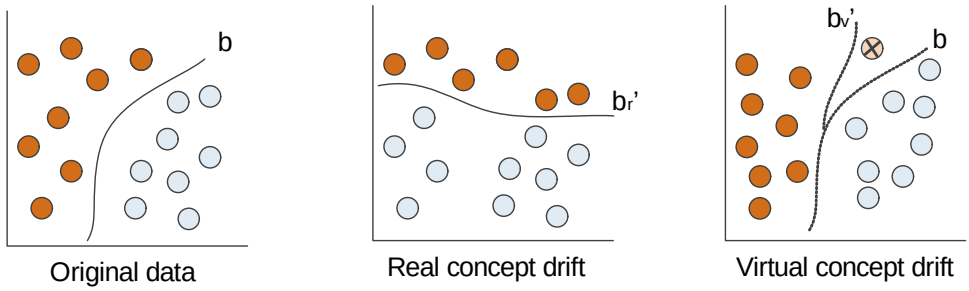


Figure 3.1: Real and virtual concept drift, adapted from [70]. A simplified two-dimensional case is presented, with two classes that are separated by a decision boundary, b. Real Concept drift is a change in the decision boundary (br'), because of some change in the context of the problem, without the class data necessarily changing. In virtual concept drift, the decision boundary theoretically does not change, but the distribution of the data inside the classes drifts. However, it can be argued that when the distribution inside the classes changes, new decision boundaries can be found that separate the classes better (for example bv'). This in turn can lead to different decisions made on new observations.

An ideal concept drift handling system should meet three requirements [68]:

1. It must adapt in a timely manner to concept drift.
2. It must be able to distinguish between concept drift and noise.
3. It should be able to recognise and handle recurring contexts (usually associated with cyclic phenomena).

There are two main strategies for handling concept drift [67]. The first is to adapt the learner in frequent time intervals independent of whether a change has occurred. The second strategy is to adapt the learner only when a change in concept has been detected. Gama et al. in [70] refer to adapting without explicitly detecting change as **blind adaptation** and adapting after detecting change as **informed adaptation**.

3.2.1 Blind Adaptation - updating the learner regularly

In the case of blind adaptation, the model is periodically retrained or updated, irrespective of whether concept drift has occurred. The goal of this class of approaches is to keep the prediction, detection or classification models up to date, by incorporating every new observation or batch of observations [70]. This is adopted in [71], where regular retraining is used to keep an anomaly detector up to date. The anomaly detector consists of modelling the normal data using an Support Vector Regression (SVR) model. It is argued that the notion of an anomaly should be defined in the context of the representation of

the current knowledge; therefore the normal model should be constantly updated with the acquisition of new data. Hence, at every timestep the SVR model is adapted by being rebuilt using all the data that has been acquired so far. However, rebuilding the model at every timestep can be very costly, especially when the size of the training data increases. To solve this problem a "forgetting" version of the algorithm is proposed which uses a sliding time window of fixed size, L; at every instance the model is rebuilt based on the last L data points. Modelling only the latest data, not only keeps the cost of calculating the model stable, but it is also claimed to be more suitable for anomaly detection, as the most recent state of the system can be represented more accurately. For real-time applications however, remodelling the system can still prove time consuming, even for windows of controlled size.

In [72] an adaptive online classification algorithm for data streams is presented. The idea of building up-to-date models under the assumption that the recent examples are the most relevant to the current concept, is employed here as well. In particular, adaptation is achieved by exponentially weighting the incoming examples, so that the most recent data is weighted more heavily, and then using them to update a perceptron-based anomaly detector. The proposed λ-perceptron algorithm is suitable for online use, because it updates the parameters with every new data piece, without requiring to re-process the entire data acquired so far. This again is an example of online learning; when a misclassification happens it becomes known straight away and this is the info used to update the model (a gradient of the error is used to update the weights).

Updating constantly has the advantage of maintaining a model up to date without the added effort of identifying the time and type of changes, or devising an explicit detection and adaptation mechanism. Nontheless, it has important limitations. The first is that, depending on the model, substantial effort may be needed to update the model continuously, which can be a problem in time-constrained applications. The effectiveness of the approach depends highly on the selection of the amount of data used for updating, i.e. the window size, as the learning algorithm is required both to perform well on stable phases (where the concept does not change) and adapt quickly to changes [67]. In periods of stability, updating constantly, can also prove harmful. Pavlidis et al. [72] find that λ-perceptron algorithm, as well as other methods that always update their parameters, tend to overfit the data when there are extended periods of only one concept. Examining medical image data, when there are extended periods in the video without any tumour

pixels, these algorithms overfit, resulting in a high rate of misclassification when frequency of tumourous appearances increases.

3.2.2 Informed adaptation - detecting concept drift

In order to avoid introducing unnecessary complexity and degrading the performance, by updating continuously at stable phases, the learner has to update only when it is necessary. Therefore, it becomes important to detect when there is concept drift and then update the system and adapt it to the new concept. In contrast to blind adaptation, informed adaptation is reactive, as it initiates an adaptation process only when a trigger has been flagged. The triggers that can be used to detect such drifts include performance measures like the accuracy of the current learner, the properties of the current model, or the properties of the data [73]. A high level view of this concept drift taxonomy is presented in figure 3.2 and analysed in the remainder of this section.

Monitoring the performance/error of the current model

This class of approaches can be used in cases where classification ground truth exists: when the algorithm makes a detection or classification for a new data point, the true label of this sample becomes immediately available. Under this assumption classification errors can be computed, as more data is processed. This paradigm is not suitable for all applications. It is commonly used for online learning, or updating predictive models online, to react to concept drift.

To deal with concept drift in an online learning scenario, Widmer and Kubat propose the FLORA algorithm family (FLOating Rough Approximation) [9]. The learner at every timestep is retrained or updated based on a sliding window containing the most recent batch of examples. The basic problem with setting the window size is: a narrow window does not accommodate enough examples to train a good model for the stable phases, while a wide window will not allow the learner to react quickly enough to concept drift. The basic concept of the FLORA algorithms is to adjust the window size, depending on the accuracy and stability of the current model and the suspicion of concept drift. A sudden dip in the accuracy is the trigger used for declaring suspected concept drift. The response is to decrease the window by 20%. Otherwise, it is increased or decreased by one, depending on whether the learning has converged. Further versions of the algorithm address recurring contexts by implementing a mechanism for context storage and recall, and noise

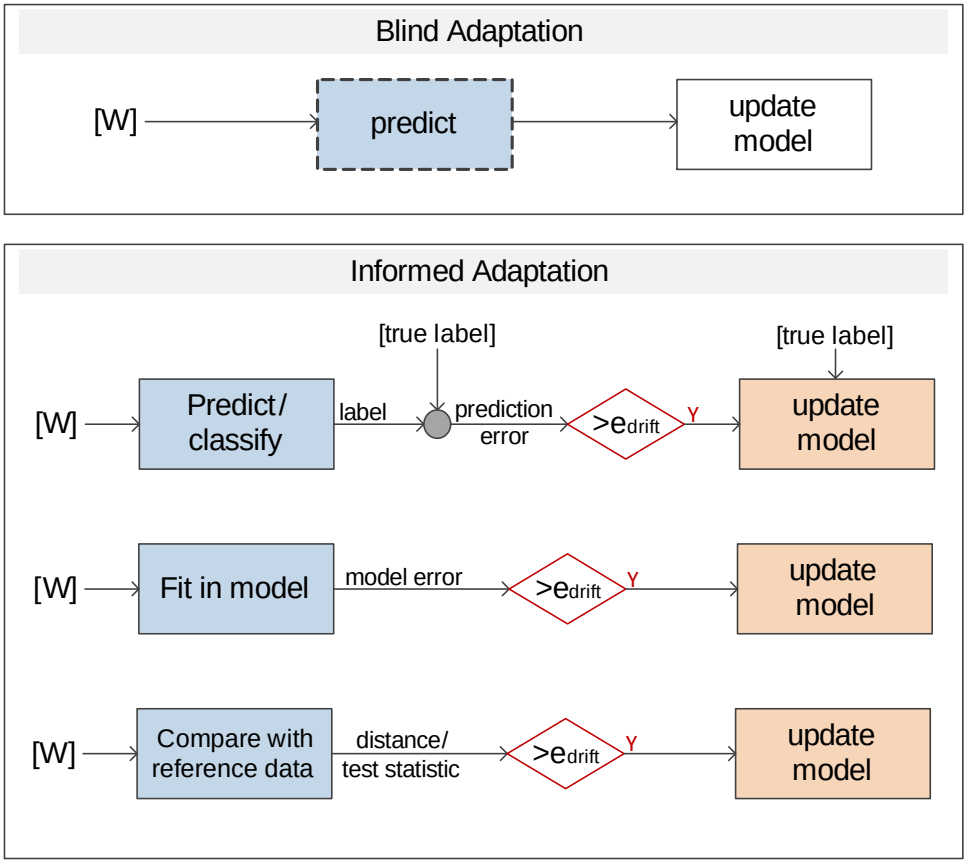


Figure 3.2: Adaptation with (Informed) or without (Blind) the detection of concept drift. As a new batch of data [W] (one or more observations) becomes available it is used to update the model. In blind adaptation, a model of the input space is continuously updated, using a pre-set quantity of recent data. Informed methods are based on measuring a property and using this to guide adaptation. This is typically used either to adapt on demand and/or to adjust the number of samples used to rebuild the model. The first type measures the prediction error of the current model, assuming that a label becomes available after prediction. The second type builds a model of the input space and measures how well the new data [W] is represented by the current model. The third type monitors the input data directly and its similarity to older input data that act as reference. In every case adaptation is triggered if the measured property exceeds a threshold, e_{drift}, that signifies concept drift, possibly with statistical significance.

tolerance by associating the accuracy estimates with statistical confidence.

To determine which data should be used for retraining when the concept changes, Gama et al. propose the drift detection method (DDM) [67] that uses a pair of thresholds declaring a warning and a drift level. The error rate of the learner's decision is monitored and used as a trigger. When the error exceeds the warning threshold a warning is produced and if it rises further and exceeds the drift threshold, then concept drift is declared. The latest examples, starting from the one that triggered a warning, until the detection of concept drift, are stored in a temporary memory and they are used to retrain the classifier.

Tested with three different classifiers, this method is found to significantly improve the final error rate of all the classifiers, compared to the case of not retraining in response to concept drift.

Based on this framework, the Early Drift Detection Method (EDDM) is proposed in [74]. Instead of using the error rate (the number of classification errors), here the distance between classification errors is monitored. Following [67] they use two thresholds, a warning and a drift threshold. The data gathered between the activation of the two thresholds is used for retraining the model. EDDM is found to outperform DDM in cases of noisy data, while an added advantage is that adaptation is also triggered when overfitting occurs. The analysis presented also shows that DDM may struggle in case of very slow, gradual drift. If the error rate stays between warning and drift level for too long, a large amount of data is retained for retraining the model, resulting in high memory requirements. EDDM deals indirectly with this issue, because it detects many concept drifts and retrains the model on smaller batches. The same two threshold framework is followed by Nishida and Yamauchi in [75]. The difference is that here a statistic is calculated to detect significant decreases in the recent accuracy. The proposed statistic, that is equivalent with the chi-square test with Yates's continuity correction, is monitored following the two threshold detection and retraining framework. It compares favourably to DDM and EDDM for sudden drift, while for gradual changes the overall error rate is comparable to EDDM.

In the previous cases, the property monitored is expected to remain stable if concept drift does not happen. However, there are applications where different types of factors can affect the monitored metric. One of these is presented in [76], which deals with adapting an Evolutionary Algorithm (EA) under a changing fitness landscape. The fitness values are used for concept drift detection. During evolution, the fitness of the population gradually improves as the EA converges; however, when the fitness landscape changes, the fitness of the population is assumed to change in an unexpected way. Forming two samples from the fitness values of the population of two consecutive generations, S_t and S_{t+1}, a statistical test is applied to determine if S_t and S_{t+1} come from the same distribution. Three non-parametric two-sample tests are compared: the Kolmogorov-Smirnov, the Wilcoxon-Mann-Whitney and the Jensen-Shannon distance. The best is found to be the Wilcoxon-Mann-Whitney test and the worst is the Kolmogorov-Smirnov, which has difficulties in distinguishing between natural alterations in fitness distributions and alter-

ations by changes in the fitness landscape.

The main limitation of this class of approaches, is the assumption that ground truth is available immediately, or even soon after the classification of a sample. However in many real applications the true labels will not become available immediately, but with unpredictable delay and they may be unreliable, biased or costly to obtain [70]. Also, retraining from scratch on a new batch of data can be time consuming and may not be suitable for real-time tasks.

Monitoring properties of the model

A common strategy for a class of network-based learning models, is to monitor the performance of the model in the form of the coverage of the input space, instead of using explicit performance information. These networks are typically created in an unsupervised manner. They do not require explicit training with a labelled data set, but they self-organise as new data appear, so that their nodes/weights encode a set of prototypes for the input space. The trigger for updating is often the case when a new input point is not sufficiently represented by the existing network.

The Radial Basis Function (RBF) network is an example of networks that use this strategy of on-line adaptation [77]. The network is structured so that each node in its hidden layer represents a fuzzy subspace. When a new example arrives, concept drift is declared if this example doesn't belong to any of the existing subspaces currently represented by the nodes. The response is the addition of a new subspace in the form of a new node and the recalculation of the weights based on a moving time-window containing only the latest inputs. To regulate the size of the network, nodes that have been inactive for a long time (no data point has been assigned to this fuzzy subspace) are deleted from the network. Even without concept drift, the weights are slightly updated so that they match the latest input better. This scheme allows adaptation to two types of changes: the addition of a new node can track changes in the operational region - abrupt concept drift. The regular updates of the weights track changes in the dynamics of the system - slow, gradual concept drift.

Similarly, [78] presents a network resembling a SOM. The network consists of nodes, associated with weight vectors, and edges that connect the nodes. The weight vectors represent prototype vectors that describe the input space, and the edges represent the proximity of those vectors. When the network is presented with an input, the node that

matches best the input is considered the winner. The winner node and its neighbours are then trained a little so that they represent the latest input better. If, however, the winning node is not similar enough to the input, concept drift is declared and a new node will be added and connected to the current two best matching nodes of the network. To stop the network from growing uncontrollably, a mechanism for deleting nodes is used, based on their age. Marsland et al. note that this system can be either used to learn dynamic distributions with changing concepts, or as a novelty detection filter. The implication of this is that it only follows a changing distribution and does not perform a task in changing conditions. In anomaly detection for instance, this paradigm has to have a way of differentiating between a change in the normal model or an anomaly.

To address this, a network that performs anomaly detection in changing conditions, due to temporal or spatial variations, is the artificial immune network proposed in [79]. The network maintains a population of antibodies which represents a model of the normal state of the system. When an antigen, i.e. a new data point, arrives, it is matched against all the antibodies. The antibody with the highest affinity (similarity) is activated and will be cloned and mutated, in order to cover the input space around the latest example a little better. Antibodies that have not been recently activated gradually die off. If no antibody is found that matches the antigen, an alarm will be raised, as it may be an anomalous antigen. This data point is also added to the antibody repertoire. Since it is assumed that normal is common and anomalous is rare, if the antigen was an expression of the possibly changed normal state, the corresponding antibody will be re-stimulated and will live in the network. Based on the same assumption, if it was an anomaly, the corresponding antibody will gradually disappear from the antibody pool and if the same antigen (anomaly) is encountered, an alarm will be raised again. This approach accounts for gradual shift in the normal behaviour, through the mechanism of mutation and for abrupt shifts through the addition of new antibodies, although initially an alarm is raised. However, a number of parameters have to be tuned to ensure that the system has a reasonably good memory of normality, but can "forget" anomalies quickly in relation to the frequency of their appearance.

Monitoring properties of the data

This family of approaches deals explicitly with the detection of concept drift in incoming data, without reference to a particular pattern recognition algorithm. The incoming data

is treated as a data stream, which has the element of temporal evolution. The method most commonly adopted is to enforce moving time windows over the input data; then the problem of detecting change is reduced to determining if the data inside the two windows have been generated by different distributions [80]. Testing whether two samples are statistically significantly different is called *statistical hypothesis testing*.

Kifer et al. [80] present a study on non-parametric statistical hypothesis testing to detect changes in data streams. The strategy they follow is to keep the reference window as the first window of the current context and with every new datum becoming available slide a detection window so as to contain the latest samples. These two windows are compared and if they are found significantly different, then a new context is declared and the reference window is reset accordingly. Non-parametric tests are considered, which make no assumptions about the underlying distribution. Two univariate statistics are proposed, the $\phi(A)$ and the $\Xi(A)$, called relativized discrepancy, and they are tested against the Wilcoxon and the Kolmogorov-Smirnov tests on data with simulated drift. In most of the tested scenarios, with the exception of discrete distributions, the ϕ and Ξ statistics outperform the other two. However, since they are univariate, they cannot be applied in data with multiple features or high dimensionality.

A sliding overlapping windows scheme is employed in [81] to detect concept drift, by monitoring a stream of one or multiple input features. Two consecutive fixed size windows slide by one at every timestep, to contain the most recent data. For the univariate case (one feature) the Kolmogorov-Smirnov and the Mann-Whitney tests are used and for the multivariate case (multiple features) the Hotelling T^2 test is selected, to statistically compare the samples contained in the two windows. A key issue identified is the selection of window size which depends both on the type of change but also on the feature space and the statistical test used.

In [82] a methodology for detecting change in data streams using a dynamic window size is presented. The ADWIN (ADaptive WINdowing) algorithm automatically increases the window size when no change is apparent and decreases it when data changes. In particular the algorithm keeps a window of size W, which is increased at every timestep with the most recent observation. Then all partitionings of the window into two (the old and the recent) are tested for similarity of the observed averages of the data in the subwindows. If the averages are found significantly dissimilar, the old sub-window is dropped. Obviously as long as no change is detected, the size of W increases linearly with time. To

address the resulting high memory and time requirements, Bifet et al. propose the improved ADWIN2, which uses a variation of exponential histograms to deal simultaneously with multiple subwindows, with no memory overhead. Compared against a fixed window strategy and a fixed window with flushing as in [80], the ADWIN2 shows competitive performance, having the added advantage of not tuning the window size. Coupled with a Naive Bayes predictor, it is compared against Gama's two threshold method, DDM [67]. As the number of samples increases, the percentage of changes detected by DDM decreases, while ADWIN2 is not affected. However, for the detected changes, the average time until detection is smaller for DDM. Generally the main advantage of the ADWIN2 is that it adapts to a variety of problems because the window size can be adjusted to the scale of the change in the different datasets.

The advantage of using this class of approaches to detect drift is that they are not limited to a specific type of model, but can be used in conjunction to any pattern recognition algorithm. Moreover, they do not require any type of feedback or additional training data, as there is no need to calculate or estimate the error of the algorithm. Their disadvantages are their sensitivity to the window size, and the fact that comparing consecutive windows can introduce unwanted time delays; this is an important challenge in real-time applications. Moreover, usually these approaches do not account for differentiating between more than one process inflicting changes in the incoming data. For instance, in anomaly detection, which is the application of interest in this work, the stream of data can change either due to an anomaly or concept drift.

3.3 Ensembles for adaptation

A paradigm for dealing with concept drift that has drawn a lot of attention recently is *ensemble learning* [83]. Ensembles are multiple classifier/detector systems that have been widely used to increase the accuracy and generalisation of classifier systems [10]. Even though ensembles are often discussed in the context of classification, the principles and methodologies can be directly transferred to other relevant tasks such as anomaly detection. Under this premise, in the following discussion there will not be separate mention to ensembles of classifiers and ensembles of detectors. In this section, the basic concepts behind ensemble learning will be briefly introduced, and their application in the context of adaptation to concept drift will be discussed.

3.3.1 Ensemble learning

An ensemble is a set of independent learning machines, called base learners or base classifiers, whose decisions are combined in order to improve the overall performance of the system. Dietterich in [10] shows that an ensemble can reduce the total probability of error compared to the case where only one learner is used. The condition for that to happen is that the base learners are **accurate** and **diverse**. A base learner is accurate if it performs better than random, i.e. $p(error) < 0.5$, and diverse if its errors are relatively uncorrelated to the other classifiers's errors. Dietterich shows how the probability of error decreases in an ensemble of L classifiers, whose decision is given by a majority vote on the individual decisions of the base classifiers: if each classifier, l, has error rate $p < 0.5$ and the errors of the individual classifiers are independent, then the probability that the ensemble decision will be wrong, P_{error}, is given by the area under the binomial distribution where more than $L/2$ base learner decisions are wrong (equation 3.1). For example, in an ensemble with 21 classifiers, each having an error rate of 0.3, the probability that more of half of them will be wrong, therefore the probability of error of the ensemble, falls to 0.026.

$$P_{error} = \sum_{i=[L/2]}^{L} \binom{L}{i} p^i (1-p)^L - i \tag{3.1}$$

Reasons for using ensembles

Stemming from the previous analysis, one of the most important reasons for using ensembles is that if each base learner has a reasonable accuracy, then statistically the probability of error for the whole ensemble is reduced significantly, provided that the errors of the base learners are somewhat independent. There are more reasons, though, for using ensembles. While each base learner might be the best in a subspace of the problem, no individual learning algorithm can be globally the best in a problem domain [84]. Ensembles try to combine the local good behaviour of the base learners in order to achieve enhanced accuracy and reliability of the overall system. Moreover, learning and combining multiple models can lead to better representation of more complex hypothesis, which cannot be learned by a single learner. There are also practical reasons for using ensembles, like too much or too little training data [85]. If a process produces large volumes of data, it might be practical to train multiple classifiers on subsets of the data. On the other hand, the lack of data can be addressed by using resampling techniques to draw overlapping random subsets of the available data and train the base learners on these. Finally, different base

learners can be trained on different data sources, for instance in the case of multi-sensor systems [86].

Methods for constructing ensembles

In an ensemble, the base learners have to be generated so that they are different from each other, thus promoting diversity [85]. Combining the relevant reviews in [10] [84], six different strategies for constructing ensembles are:

Mixtures of experts Every base learner is trained only for a region of the input space. Then, depending on the available input data, a supervisor learning machine selects the most appropriate base learner.

Manipulating the training examples The learning algorithm runs on a different subset of the training data for every base learner. The two most popular ways for creating different subsets of the training data are bagging and boosting [87]. In bagging, new training sets are created by uniformly sampling the training data, while in boosting, sampling is done using a weighted distribution: for each base learner the training examples that were most misclassified by the previous learner are favoured.

Manipulating the input features A subset of the input features is fed into each base learner. The selection of the subsets of features can be done either by hand, using domain knowledge, by feature extraction algorithms, or stochastically by randomly sampling the feature space.

Manipulating the output targets This class of methods, also called output decomposition, creates subsets of the original classes (assuming a multi-class classification problem) or, in more general terms, reduces the output to a binary one. The division of the output classes is different for each classifier.

Injecting randomness Injecting randomness can lead to a better exploration of the model space. Depending on the learner used, different starting point can be selected for a local search algorithm, or random initial weights for a neural network.

Test and select methods These methods select the best combination from a pool of possible base learners, so as to optimise the ensemble performance. One way to achieve that is through greedy search, where a new base learner is only added to the

ensemble if it improves the overall performance [84]. In principle any optimisation technique can be used to select the best team of base learners.

Combining the individual outputs

After each learner generates an output, the individual outputs are combined into the ensemble output. The selection of the combination strategy depends among other factors on the problem at hand, any available a priori knowledge and the type of the output [84]. In the case of discrete output (e.g. a class label) a straightforward way to combine the individual output labels is majority voting. This can be refined by weighting each learner. The weights can reflect the estimated performance of the classifier, its estimated contribution to the ensemble output, or some other metric. In the case of continuous output, averaging and weighted averaging are the corresponding strategies. Instead of using the outputs directly to induce the ensemble's decision, they can be used as inputs to a second-level learning machine that is trained to combine the individual decisions. A recent comparison study between traditional combination schemes and optimised by evolution schemes is presented in [88].

3.3.2 Ensembles for concept drift

Most of the current research on multiple classifier systems assumes that the classification problem is fixed. However, "everything that exists changes with time and so will the classification problem" [83]. Ensemble methods are accurate, flexible and sometimes more efficient than single classifiers and recently their application in changing environments has been explored [83]. The problem examined is that of data streams with concept drift. The two main characteristics of such applications are the huge amount of data and the drifting concepts: Wang et al. [89] argue that maintaining a single up-to-date classifier for infinite data streams with concept drift poses the following serious challenges. Firstly, it is difficult to decide which examples represent the outdated concept; forgetting examples at a constant rate risks either not picking up transients in the data or not having a stable enough model. Secondly, a slight drift can trigger disproportionately large changes in the model and compromise the learning efficiency. Finally, substantial implementation efforts are required to adapt the single classifier to drifting concepts.

An ensemble could possibly deal both with the high data volume and drifting concepts, by using subsets of the available data points to train the base classifiers [90]. Since there

are multiple experts in the ensemble, the total learning efficiency is not compromised as much if the base learners are different, because different expertise will be retained in the multiple models. Finally, as far as speed is concerned, although Kuncheva [83] notes that it might not be time efficient to run and update an ensemble instead of a single model, it has been found that a simple ensemble might be easier to use than a single adaptive decision tree for mining changing data streams [89]. Moreover, if the base learners predict and update independently, the possibility of parallelisation should be considered, which could greatly loosen the time constraints.

In order to take advantage of the benefits of ensemble learning in applications involving concept drift, the ensembles have to adapt. An ensemble can adapt by updating the base learners using recent data, by creating new learners, pruning out old members, dynamically recombining the base learners, or combinations of the above [83]. Most of the current work uses ensembles to deal with the online learning - prediction paradigm. Under this paradigm, the current model/knowledge is used to classify a new input data point or a new chunk of data, or to predict the next value in a data stream. Immediately, or soon afterwards, it is assumed that the ground truth, i.e. the true label(s), become available. This information is used to evaluate the accuracy of the predictions of both the ensemble and the base learners. Similarly to many concept drift methods, the error of the model is used for updating. It either guides a regular adaptation, or it indicates concept drift and triggers adaptation in response. The following discussion presents the state of the art in these two types of ensembles for concept drift.

3.3.2.1 Update the ensemble regularly

In order to track concept drift and changing environments, this class of methods regularly update the ensemble every time training data becomes available. A basic approach outlined in [90] is to use the latest labelled data to train and add new experts and remove base learners that perform poorly. The performance of a base learner in this context is assessed based mainly on its accuracy on a subset of points whose classification by the ensemble is ambiguous. The proposed Streaming Ensemble Algorithm (SEA) [90] receives the data in batches and trains a new base classifier only on the latest batch. The new classifier is added to the ensemble, until a maximum size is reached. When that happens, the new classifiers are added only in replacement to a lower quality classifier. To promote diversity, the quality is determined based on how well the base classifiers classify points

on which the ensemble is undecided: learners are not punished heavily on points that are either easy or impossible to classify. The SEA is tested on synthetic data with concept drift, against a single, incrementally trained decision tree. The results show that the SEA recovers faster from concept drift because, unlike the single incremental classifier, it does not retain outdated knowledge. However, the size of the data block is found to be a critical parameter; bigger blocks lead to higher accuracy in stable phases, and smaller blocks lead to faster adaptation to change.

The Accuracy Weighted Ensemble (AWE) proposed by Wang et al. [89] creates new learners in a similar way, with the difference that a weight is assigned to each classifier (which reflects its quality) and it is used both in a weighted majority vote and to decide if a new classifier will be added or removed. The weight of every classifier here is determined as the difference between the mean square error of the classifier on the latest training data and that of a hypothetical random classifier. Again, the ensemble has a fixed size, so the new classifier is added only if there is a worst classifier in the current ensemble to be replaced. The AWE is found to outperform a single classifier build only on the most recent concept, when a reasonable number of base learners have been added (up to 6 depending on the dataset). Wang et al. also determine that the size of the data chunk can be a critical factor, as large chunks lead to high error rates, because concept drift can exist inside the chunk. Small chunks can also lead to errors, as the base classifiers are not supported by enough training data.

In order to follow different types of drift, in [91] the latest batch of data is used both to train a new classifier (which is added to the ensemble with a heavy weight) and to update incrementally all the base learners. The creation and heavy weighting of new classifiers on the latest data, allows the rapid adaptation to sudden drifts. The slow incremental updating of all the classifiers allows good accuracy in slow gradual drifts, but also in periods of stability. A similar combination is used in [92] where all the classifiers are adapted online with every training example that becomes available. If the weight of a classifier, which is calculated using ideas from reinforcement learning, falls below a threshold, it is cleared and retrained on a small batch of training data stored for this purpose.

One important quality of adaptive ensembles, that single classifiers do not have, is their ability to retain past knowledge and expertise. The Learn++.NSE algorithm proposed in [69], weights the members of the ensemble not only based on the accuracy on the latest data block, but also taking into account the error history of each classifier. Moreover,

Learn++.NSE does remove old learners from the ensemble. Assuming that there are no memory constraints, all the classifiers are retained and used if relevant to the current concept. When a base learner is not relevant to the current concept, it is weighted low and has effectively negligible contribution to the weighted majority decision. The algorithm is found to accommodate a wide variety of drift scenarios, while a weight analysis shows that the use of existing knowledge is very efficient, because early classifiers are reactivated when they are needed the most and temporarily disabled when they are irrelevant. Even though not removing classifiers can lead to good performance, this comes with high time and memory requirements. Brezinski et al. [91] have examined using a buffer for old classifiers to address recurring contexts, but a memory and time analysis has indicated that this is costly and should be used only when cyclic phenomena are highly expected.

3.3.2.2 Update when needed

The second class of approaches does not create new classifiers/learners all the time, but only when needed. This can be accomplished by combining an ensemble with an explicit concept drift detection method, for instance combining online bagging [93] with the ADWIN algorithm [82] in order to determine when the concept has changed and when a new learner must be added [94]. However, the most common approach is to monitor the global output. A very popular algorithm in this category is the Dynamic Weighted Majority (DWM) algorithm [95]. The DWM uses a dynamically maintained ensemble of learners to achieve quick convergence of the system to the new concept. In a classification example, the decisions of the base classifiers are weighted and the class with the highest accumulated weight is the global estimate. When a new training point becomes available (labelled example), the local and global decisions are evaluated. The DWM follows the current concept by penalising the weight of base learners that misclassify the example and removing them when their weight reaches a lower threshold. A new learner is added every time the global prediction is wrong. DWM is found to accurately track drifting concepts and quickly converge to the new concept after the change. Kolter and Maloof extend their work in [96] to form a more general framework, Add.Exp, which addresses the continuous case (for regression tasks) as well, and modifies the weighting mechanism so that the newly added learner does not dominate the ensemble. This way Add.Exp is more robust that DWM; it doesn't allow for new individuals to dominate, as noise can be incorporated into the knowledge of the system. Finally, in [97] the Early Dynamic Weighted Majority

(ERDWM) is proposed, which adds to DWM the increase of the classifier weights, if its decision is correct when the global decision is incorrect, and an upper limit in the ensemble size. The ERDWM is tested against DWM and a single classifier with an EDDM adaptation method [74] (reviewed in section 3.2.2), on synthetic data with concept drift. The ERDWM exhibits similar accuracy to the DWM but, because of the fixed ensemble size, it has lower memory and time requirements, especially right after the change in concept. It is argued that the single EDDM classifier is better in terms of memory and execution time and hence more suitable for resource constrained environment. However, both DWM and ERDWM give more reliable and accurate results even in noisy environments, as they do not forget the initial concepts and take into account previous learning and experience.

A drift-triggered adaptation is combined with regular updating in a hybrid approach, ACE2 [98]. The basic concept of ACE is that it maintains both an online classifier and batch classifiers that are added when concept drift is detected (monitoring the prediction accuracy over a window), or when a maximum number of new examples is observed (regularly). Whenever a batch learner is added the online learner is reset. This way the online learner closely tracks the most recent concept, while previous knowledge is stored in the batch learners. The decision of the ensemble is based on weighted majority voting, where only classifiers that make confident predictions are used. Classifiers are removed taking into account the age and recent accuracy. The ACE2 is tested on synthetic data with drift and is found able to respond to all types of drift: sudden, gradual and recurring. Their study shows that ACE2 outperfroms Add.Exp for gradual drift because the latter does not change the learners quickly enough in the case of gradual drift. AWE is also outperformed by ACE2 in the case of sudden drift because very small data chunks are needed in AWE for quick adaptation, which, as discussed earlier, leads to degradation in performance. Moreover, the conservative pruning mechanism allow ACE2 to handle recurring drifts.

3.3.2.3 Diversity

The fact that diversity is a very important factor in ensembles, has been established in studies for the fixed classification problem [99]. Recently, a study has been done on the impact of diversity in ensembles for the case of concept drift [100]. The main finding of this study is that different levels of diversity are needed before and after drift. When there is no drift, low diversity ensembles lead to better convergence when learning a stable concept. However, high diversity ensembles help to quickly recover from a sudden increase

in the error that is caused by the change in concept. In other words, high diversity is useful to reduce the initial increase in error, but afterwards, low diversity should be adopted to aid the convergence to the new concept. Based on this study, Minku et al. propose the DDD Diversity for Dealing with Drifts - framework [101]. The DDD employs different levels of ensemble diversity before and after concept drift to improve the accuracy and obtain better generalisation on the new concept. This is also a hybrid approach that maintains a set of ensembles regularly updated using online bagging [93] with different diversity levels; the diversity level is controlled by manipulating a parameter of the online bagging. Concept drift can be detected using a standard drift detection method, and the response of the system is to switch to higher diversity ensembles. When a new stable phase is reached, the system switches to lower diversity ensembles. This scheme leads to increased accuracy compared to the DWM both in cases of stability and in the time right after drift. Additionally, it has better accuracy than EDDM mainly when the drifts have low severity or low speed. It also has good robustness to false alarms as it maintains the old ensembles. One disadvantage, however, is that, as the DDD maintains four ensembles, it has relatively high time and memory requirements.

3.4 Concept drift in chemical agent detection

In chemical agent detection, concept drift can be expressed in many ways. It is reminded to the reader that, according to [68], real concept drift is defined as a change in the target concept, and virtual concept drift as a change in the underlying distribution. Chemical agent detection is subject to both types of concept drift, as it often has to deal with non-stationary data and environments. Real and virtual concept drift translate in changes in the targets to be detected and changes in the chemical background respectively. Real concept drift can be attributed to either new, previously unseen targets, or to the same targets, but with different signatures, due to variations including sensor drift. The sensor drift effect is a dynamical process caused by physical changes either in the sensors or in the chemical background [32]. Moreover, a chemical agent or mixture may not give a well-defined signature because of the memory effect [32]; a measurement at time t is influenced by the sensor's measurement at time $t-k$, i.e. it is influenced by the last k timesteps. The chemical background itself can change over time gradually due to long time deployment, or due to moving between different locales where the normal background involves a different mixture of chemical compounds. It should be noted here that concept drift is not to be

mistaken for sensor drift, which is a phenomenon relating principally to the morphology of the sensor. Concept drift is the general problem in machine learning, while sensor drift can be considered as a form or cause of concept drift.

All the above can be considered as hidden contexts. They are not directly measured, but can cause the degradation of the performance of a chemical detection algorithm. Discussing the problem of sensor drift, Sisk et al. argue that constantly retraining the detection algorithm, using a set of calibrants, is an effective and robust solution [102], but is time-intensive and the system needs to be taken offline for retraining. Separating and rejecting drift effects from real responses assumes the existence of calibration data containing a significant and known amount of drift [103] [104]. As concept drift (either in form of sensor drift or changing environment) is an unexpected phenomenon, this class of methods cannot be used in real-time applications. Instead, in the following analysis, attention is drawn to adaptive models, which aim to adapt a pattern recognition algorithm by taking into account changes in the input feature space, caused by drift effects. Most of the research on adaptive chemical detection focuses on detecting and dealing with the effects of sensor drift or changing environmental conditions (which sometimes are interdependent). However, for the purpose of this review they are grouped under the general machine learning problem of concept drift and adaptation.

3.4.1 Adaptive chemical agent detection systems - single models

To address changing environmental conditions, the fuzzy ARTMAP network is used in [105]. The fuzzy ARTMAP is a supervised Adaptive Resonance Theory (ART) network that is self-organising and self-stabilising and it is used as the pattern recognition module of an electronic nose. The way the network self-organises resembles the strategies used in [77] [78] [79]: every time a mismatch is detected, the network stores the novel pattern in additional weights. A slow recode strategy is also used to change existing weights towards the actual input pattern. In this way the network is able to learn about new events without "losing" its acquired knowledge. The ability to slow recode over learned categories is argued to help the network adjust to phenomena like long-term sensor drift. The algorithm is tested on a real dataset for cattle diagnosis. The measurements using an electronic nose were taken on the field over a two-week period and they are likely to reflect environmental changes over the data collection time. The results indicate adaptivity and suitability for non-stationary environments, as the algorithm outperforms the MLP with

a 79% correct classification rate. However, it is not conclusively proven if this is due to adaptivity or noise tolerance, partly due to the lack of controlled concept drift events.

A method that deals with sensor drift and can be coupled with any classifier is presented in [106]. A pattern recognition algorithm is trained on a training dataset, collected under stable conditions, and the centroids of the elements belonging to each class are calculated. At runtime, each new sample is classified after applying a drift correction factor, *cf*. This factor is periodically updated, using the recent data and classifications, to account for the current drift: every W samples (fixed sliding window), an evolutionary process is invoked to find the optimal *cf* that minimises the cumulative distances of the samples in W to the centroid of the classes to which they were assigned. The assumption in this approach is that the change is gradual enough, or the classes are separable enough, that no misclassifications will happen because of the drift in this window. For artificial datasets the correction process improves the results for 4 classifiers tested, when a window size of 50 is used. However, for the real data, a window of 100 is needed to tackle the additional complexity. Although there is still some improvement over the static classifiers, the non-homogeneity and partial overlapping of the classes (there are more misclassifications), hinders the performance of this method. Additionally, the big window increases the effort needed for the computation of the correction matrix, which limits real-time applications.

In [107] Perera et al. investigate leakage detection using an electronic nose, under the presence of sensor drift. The proposed Recursive Dynamic Principle Components Analysis (RDPCA) uses a fixed-size moving window to rebuild at every timestep a variance decomposition of the data based on the latest data. The dynamic model takes into account both feature covariances (sensor interdependency) and the covariance between a given feature and its past readouts (temporal evolution of sensors). For detection, the residuals of the incoming sample are used. The parameters of the RDPCA are optimised via ROC plots [1], which requires sufficient labelled data. The performance of the detector under heavy concept drift is tested on simulated data, based on models of the real sensor data, to which simulated sensor drift has been linearly added. This RDPCA algorithm is found able to detect clearly oil leakage events under heavy sensor drift. Static PCA [109] on the other hand is found unable to discriminate between anomalies (leakage events) and

[1] A Receiver Operating Characteristic (ROC) curve quantifies and visualises the detection performance as the rate of false positives against true positives, when a given parameter or detection threshold is modified [108]

the drift. One problem of RDPCA is that the algorithm needs careful tuning, and a large volume of labelled data is required for this. Moreover, this is a blind adaptation paradigm and the proposed model is not trivial to rebuild at every instance, which can limit its real-time applicability, especially if higher dimensional data is considered.

In a task of gas classification using a sensor array under the presence of sensor drift, two regularly updated models are proposed in [6]. The first is a self organised classifier, which classifies a gas based on proximity to a set of templates. The gas templates are updated with every sample, to track the drifting signal. This method fails to track the classes if they are not sampled frequently. The second approach proposed, uses system identifications theory to model the sensor array based on the readings of all the sensors. A model is constructed for each sensor-gas pair. Every gas model makes a prediction for the responses of all the sensors at the present timestep; the current data point is classified as the gas that has the overall lowest prediction error across the models of all the sensors in the array. This method tracks the classes more effectively, achieving a long term 85% classification accuracy. However, long pauses in the measurements cause misclassifications, as the dynamics between the sensors of the array can change during the pause.

Dependency on the frequency of the classes is a common problem of adaptive classifiers. To solve this, Martinelli et al. propose the A^2INET [110]. A^2INET is based on the immune network algorithm AINET [111], an unsupervised algorithm that models the measurement dataset as a set of templates (antibodies). A^2INET creates an AINET model for every class and a new sample is assigned to the class that contains the highest affinity antibody to the new datum. To follow sensor drift for the specific class and match the new datum better, the class templates undergo hypermutation. To compensate for classes that were not updated, a global centroid normalisation is proposed. All the antibodies (of all the classes) are forced to follow the same variation as the centroid of the last updated class. The parameter that determines how fast the centroids will move is critical, as it needs to be large enough to follow the drift, but not too large as to cause the antibodies to move faster than drift. Tested on simulated data, it is shown that if the presentation of the classes is balanced, the centroid normalisation does not have an effect. However, in the case of class absence for a prolonged time, using the centroid normalisation offers an improvement of up to 30% in the synthetic data. The main drawback of this approach is that it depends heavily on the value of the normalisation parameter. To overcome this, they suggest that an intermediate training step could be assumed, to tune

this parameter on data that have some drift. However, the acquisition of such data can be tricky, as drift is a phenomenon that can appear unexpectedly.

3.4.2 Adaptive ensemble in sensing applications

The majority of the research in using ensembles for sensing applications, addresses a static scenario and uses an ensemble for multi-source data classification applications [112] [113]. The training data can be manipulated through bagging, boosting, or assigning different sources to the base learners. The latter has been used in e-nose application, where every sensor is assigned to a base learner [114]. Alternatively, in the case of high dimensional hyperspectral data, input feature manipulation has been used for creating an ensemble, where each classifier is assigned a feature subset [115]. An approach that deals with sensor drift is presented by Vergara et al. in [116]. In that work ensembles are used to classify data from a 16 sensor e-nose, collected over a 36 month period, in which sensor drift is captured. This data is divided into batches and base classifiers are sequentially created, trained on these batches, following the paradigm of regularly updating the ensemble (section 3.3.2.1). The main limitation of this, which they acknowledge, is that it is a supervised, offline approach, where labels are assumed to become available instantly after the prediction. In a real-time scenario, the labels might not become available immediately; the bigger the delay between the current batch and obtaining some labels, the more the performance would deteriorate.

3.5 Summary and discussion

In this chapter, the problem of concept drift and adaptation has been discussed. Concept drift is the phenomenon of change either in the target concepts or in the processes that generate the data of the application. In every case, a system which has been trained on labelled training data drawn from fixed distributions, experiences a degradation in performance when these distributions change. This problem is of particular interest in autonomous real-time chemical detection. Concept drift can occur because of changes in the environmental conditions (e.g. temperature or humidity), moving to different locations (in the case of a chemical sensing mobile robot), or sensor drift, i.e. the degradation of the sensors' performance over time. The purpose of this work is not to analyse or model the reason that caused this change. In chemical sensing in particular, it has been argued

that separating and rejecting drift effects from real responses can be impractical because it assumes the existence of calibration data containing a significant and known amount of drift [103] [104]. Instead, the aim of adaptive systems is to make the necessary changes to their models, so that they avoid or recover from the degradation in performance in the face of concept drift.

There are two main strategies for handling concept drift. The first, blind adaptation, updates the model (classification or anomaly detection model) regularly, without knowledge of whether there has been concept drift. However, in order to avoid introducing unnecessary complexity and degrading the performance by updating continuously at stable phases, the learner has to update only when it is necessary: informed adaptation is reactive, as it initiates an adaptation process only when a trigger that indicates that concept drift has happened has been flagged. The triggers that can be used to detect such drifts include measures like: the accuracy of the current learner, the properties of the current model, or the properties of the data. The first can be used in cases where classification ground truth exists, i.e. the true label of a new data point becomes immediately available after prediction or classification; misclassifications can be used then as an indication of concept drift. This paradigm is not suitable for all applications, as in many real applications the true labels will not become available immediately, but with uncontrollable delay and they may be unreliable, biased or costly to obtain [70]. An alternative is to monitor the performance of the model. This can be applied in specific network-based learning models, which encode in their nodes/weights, a set of prototypes for the input space. The trigger for updating is often the case when a new input point is not sufficiently represented by the existing network. Monitoring the data deals explicitly with the detection of concept drift in incoming data. The method most commonly adopted is to enforce moving time windows over the input data stream; then the problem of detecting change is reduced to determining if the data inside the two windows have been generated by different distributions [80]. The advantage of using this class of approaches to detect drift is that they are not limited to a specific type of model, but can be used in conjunction to any pattern recognition algorithm. Moreover, they do not require any type of feedback or additional training data, as there is no need to calculate or estimate the error of the algorithm. Their disadvantage is that they are sensitive to the window size and comparing consecutive windows can introduce unwanted time delays.

In recent years attention has been drawn to ensemble learning for adaptation in appli-

cations with concept drift. It has been argued that in data stream applications, with high volumes of data and drifting concepts, single adaptive classifiers face several challenges: it is often challenging to adapt the classifier under large volumes of data, and selecting subsets of data carries the risk of deciding which data is relevant to the recent concept. Additionally, the reaction of the classifier can be disproportionate to the amount of drift. An ensemble could possibly deal both with the high data volume and the drifting concepts, by using subsets of the available data points to train the base classifiers [90]. Additionally, since there are multiple experts in the ensemble, the total learning efficiency is not compromised as much if a subset of the base learners do not capture the new concept perfectly. Most recent work on ensembles for concept drift is promising: ensembles have been found to outperform single incremental classifiers. The latter use all the data and this makes them slow to react. Moreover, they are more robust, as they retain past information and they can recover more easily from false positives. It has also been found that switching to a highly diversified ensemble can provide a fast response to concept drift [101]. The main limitation of the existing work is that typically it is assumed that ground truth is regularly acquired, an assumption which, as discussed earlier, does not hold in autonomous real-time applications.

Most existing adaptive chemical detection approaches try to maintain up-to-date models of the sensors and the classes, which can help in tracking slow drift. However, a lot of these models are costly to continuously update. They often rely on balanced representation of all classes, or they attempt to mitigate the update of one class/sensor to all the sensor/class models. This as well can be challenged, as it has been found that different sensors drift in different directions. Moreover, the existing approaches deal almost exclusively with sensor arrays and are likely to be challenged in high dimensions associated with spectrometry data.

This chapter serves as a review of the domain of adaptation and the most used methods to address concept drift. The purpose of the work presented in this thesis is to develop a chemical detection system that will adapt autonomously and in a timely manner. Informed adaptation ensures that resources are not dedicated to adapting the model without needing to do so, and monitoring the input data can be used in parallel with the RDA for detecting drift. This is the problem analysed in Chapter 4. Adapting in response to concept drift is also hindered by the lack of feedback (effectively the lack of training data). The possibility of retraining the RDA on the most recent data, either to replace

the current RDA, or as a new base learner of an ensemble is limited. Drawing inspiration from the fact that switching to a highly diversified ensemble can provide a fast response to concept drift, an ensemble approach that uses a diverse ensemble of existing RDAs to respond to concept drift, is presented in Chapter 5.

Chapter 4

Detecting Concept Drift Using Statistical Hypothesis Testing

The focus of this chapter is the detection of concept drift in chemical data, as part of the development of an adaptive chemical detection system. An autonomous system, that can be used without constant user supervision, not only has to have adaptation mechanisms in place, but also the capability to detect the need to adapt. This adaptive system is developed for use by a robot equipped with chemical sensors, navigating changing environments and detecting chemicals of interest.

The main computational unit in the adaptive chemical detection system is the RDA [4], a detailed description of which is given in section 2.5. The RDA is an immune-inspired algorithm, which aims at discovering anomalies embedded in noisy data and extracting clean, noise-free signatures which can be used for recognition of these anomalies. It has been used in [4] [5] as a tool for anomaly detection and generation of clean signatures that are used for chemical identification.

The RDA uses negative feedback to suppress the input and ensure that only true anomalies are detected, instead of spikes in the measurements or noise in the data (equations 2.8, 2.9). This allows for some tolerance, but it is not sufficient when the background changes significantly, e.g. when the robot moves to a new environment where there is a constant presence of a new chemical. This type of change, constitutes a change in the model of normality (see figure 4.1) and changes the context of the problem. This change in the background data generating model is referred to as *concept drift*.

The detection of change is a very important step towards creating an adaptive algorithm. Concept drift is essentially a "permanent" change that has to be learnt and its

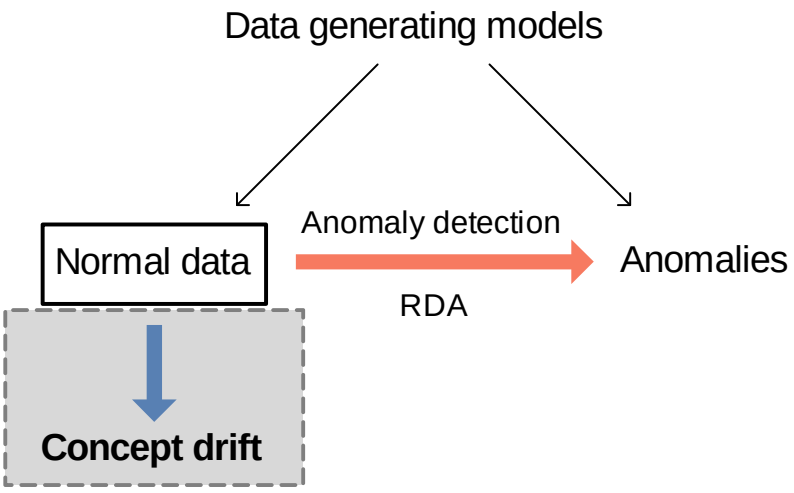


Figure 4.1: Concept drift in the normal data model. Concept drift in machine learning is a change in the data generating distributions [9]. The case of interest here is a change in the normal background, for instance when the robot moves to a new environment. The occurrence of concept drift and its subsequent detection is independent of the occurrence and detection of anomalies.

detection is the cue that will trigger the adaptation of the system. As discussed in the previous chapter it is important that the system adapts when a real change has happened. Systems that adapt blindly, irrespectively of whether there has been concept drift, risk introducing unnecessary complexity, which in principle is not ideal in a resource-constraint application. So, assuming that the cost of constantly reoptimising/adapting the algorithm is higher than the cost of detecting change, it is argued that the system should only adapt when concept drift is detected. The triggers that can be used to detect concept drift include performance measures like the accuracy of the current learner, the properties of the current model, or the properties of the data [73]. Monitoring the data directly to detect concept drift has the advantage of not requiring external feedback and can be implemented in parallel with the RDA.

Following from that, the structure of the proposed system is presented in figure 4.2. The main module, which is responsible for the detection of chemicals of interest, is the anomaly detection module, which uses the RDA. In the training phase, the RDA is trained using labelled data, appropriate for the specific application. Training of the RDA is equivalent to setting its parameter set to suitable values (parameter tuning). This can either be done manually, using properties of the single receptor to ensure appropriate behaviour [63], or it can be automated, using an optimisation algorithm; in [65] a Genetic Algorithm has been used to find the parameter set that optimises the performance of the RDA.

After suitable parameters have been found for the RDA, i.e. for the anomaly detection module, the system starts its normal operation (system runtime phase). The trained RDA is used to detect anomalies in real-time, on unknown data collected with chemical sensors. When there is a change in the background environment, the RDA is likely to fail in accurately detecting anomalies. The proposed solution is to implement a concept drift module, which operates on the same sensor data and its goal is to detect such a change in the environment and trigger an adaptation process of the RDA to this new environment. Statistical hypothesis testing, which operates directly on the raw sensor data and looks for significant differences between past and present data, will be used to detect concept drift. This chapter, hence, focuses on the implementation of a concept drift detection module, using statistical hypothesis testing, while the adaptation module will be presented in Chapter 5.

This chapter will address the detection of concept drift in chemical data, under simulated concept drift. The data used are associated with the 2009-2010 ICARIS competition [117] and have been collected, under laboratory conditions using a mass-spectrometer.

4.1 Structure of this chapter

A review of statistical hypothesis testing for follows in section 4.2. Section 4.3 presents the methodology of applying statistical tests in mass spectrometry data to detect concept drift, and of enhancing the data with artificial concept drift. The experimental framework, in which the concept drift detection methods have been tested, is presented in section 4.4. Finally, the results of experiments of concept drift detection in the presence or not of anomalies are presented and analysed in section 4.5, followed by a summary and discussions in section 4.6.

4.2 Statistical hypothesis testing for concept drift detection

Chemical sensing deals with multivariate time series data. As chemical detection is treated as an anomaly detection problem, the signal/input corresponding to a specific anomaly-free environment will be referred to as the background signal, or background environment. The goal of this work is to detect changes in this background signal (concept drift), without any assumptions about the generating distribution. Statistical hypothesis testing typically

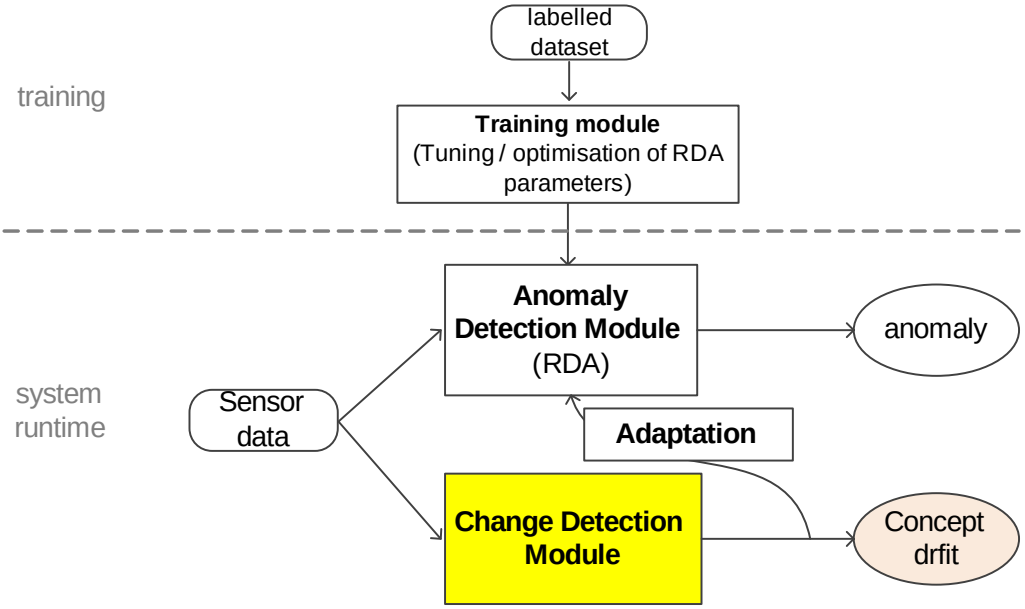


Figure 4.2: Proposed system and concept drift detection module. The proposed system can be decomposed to three modules, the anomaly detection module, the concept drift detection module and the adaptation module. The anomaly detection module is realised via the RDA and detects anomalies in the incoming sensor data. This module has to be trained before use, on suitable training data. The concept drift detection module uses the same incoming sensor data to detect changes in the background that could affect the performance of the RDA and triggers an adaptation module in response to the detected change. The adaptation module is responsible for re-training or re-calibrating the anomaly detection module to avoid the expected decrease in performance.

compares recent data to a historical reference sample using an appropriate statistical test; if a statistically significant difference is found between the two, then concept drift is detected [80].

When the underlying distribution of the data cannot be assumed to be normal, non-parametric statistical tests are used (in contrast to parametric tests which are used to compare normal distributions). A class of non parametric tests uses some form of ranking of the samples in order to determine the similarity of the distributions generating the data. Such tests are the Mann−Whitney, the Smirnov and the Wald−Walfowitz test. These cannot be implemented directly on multidimensional data, such as data that come from a mass−spectrometer or a sensor array. Instead a methodology has been proposed by Friedman and Rafsky [118] to adapt the Smirnov and Wald−Wolfowitz test to multidimensional data, using a minimal spanning tree constructed by the data. This methodology is additionally extended here to the Mann−Whitney test, as well. Apart from adapted statistical tests, there are other methods that are designed specifically for multi−dimensional

data. The method of this category that has been used here is the Maximum Mean Discrepancy method [119], which uses a kernel function to transform the data to a lower dimension.

The work most relevant to that presented in this thesis is [120], where a number of non-parametric, multivariate statistical tests is compared on available datasets. Statistical tests are compared in this chapter as well, to establish whether it is a suitable method for the detection of concept drift in the chemical sensing domain. However, this investigation is extended to combining statistical hypothesis testing with the existing anomaly detection system, and examining whether it is possible to distinguish between concept drift and anomalies.

4.3 Materials and methods

In order to detect whether there has been a change in the background distribution of the incoming sensor data, statistical tests can be used, to compare recent samples to some reference sample. Letting the most recent sample be X and the reference sample be Y, statistical tests aim to determine if there has been a significant change between the two samples. The null hypothesis that is tested, H_0, is that X and Y are generated from the same distribution.

4.3.1 Non−parametric statistical tests

Non-parametric statistical tests have the advantage of not making assumptions about the underlying distribution of the samples in order to determine if two samples come from the same distribution. In the case of the data coming from the chemical sensor, a known underlying distribution cannot be assumed; therefore, non-parametric statistical tests are a reasonable choice for the comparison of the two samples X and Y. To state the problem formally, in the one dimensional space it is assumed that the two samples to be compared are $X = \{x_1, x_2, ..., x_m\}$ with m observations and $Y = \{y_1, y_2, ..., y_n\}$ with n observations, where $x_1, x_2, ..., x_m, y_1, y_2, ..., y_n \in R$. The pooled sample is defined as $X \bigcup Y = \{x_1, x_2, ..., x_m, y_1, y_2, ..., y_n\}$ with $N = m + n$ observations.

The three tests selected from this category to use in the context of detecting concept drift, are the Smirnov test, the Wald-Wolfowitz (runs) test and the Mann-Whitney test [121]. Details on these statistical tests are given in the following subsections.

Smirnov test

The Smirnov (or KolmogorovSmirnov, or KS, test) two sample test is a non-parametric test that determines if two samples, X and Y belong to the same population or distribution, and it is based on quantifying the distance between the empirical distribution functions of the two samples. The test starts by ranking the pooled sample, XUY, in ascending order. Then, for each data point, $i, 1 < i < N$, in the pooled sample the quantity d_i is calculated:

$$d_i = \frac{r_i}{m} - \frac{s_i}{n} \tag{4.1}$$

where r_i is the number of X observations with a rank less than or equal to i. Similarly, s_i is the number of Y observations with a rank less than or equal to i. The test statistic is $D = max\{d_i\}$ and can be used directly, with the null hypothesis being rejected for large values of D. Alternatively, the statistic has been tabulated and critical values are available.

Wald−Wolfowitz (Runs) test

This test is lower in power than the Smirnov test, but it is very straight forward to compute [121]. The Wald-Wolfowitz test also starts by sorting the pooled sample. Instead of ranks, this time the data points are assigned labels, + or −, depending on whether they originate from sample X or Y respectively. A run is defined as a group of consecutive same labels in the sorted pooled sample. The quantity of interest, which acts as the test statistic, is the number of runs in the sorted pooled sample, R (see example in figure 4.3). This can be used directly to qualitatively reject the null hypothesis for low values of R, or the statistic W can be calculated according to formula 4.2 for which the critical values are available.

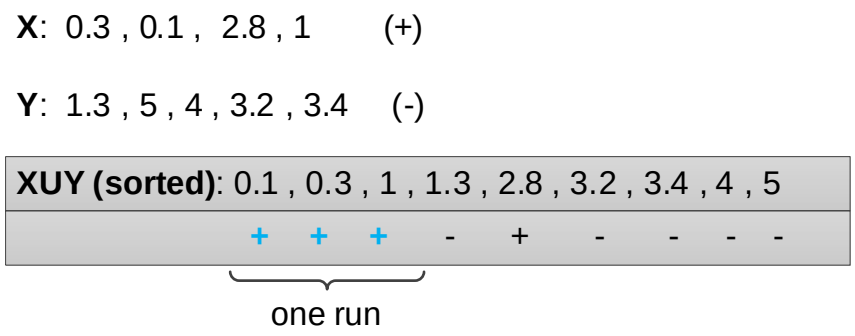


Figure 4.3: The Wald-Wolfowitz (runs) test. In this example the number of runs R is 4.

$$W = \frac{R - \frac{2mn}{N} - 1}{(\frac{2mn(2mn-N)}{N^2(N-1)})^{1/2}} \tag{4.2}$$

Mann−Whitney test

The Mann-Whitney U test (or Wilcoxon rank-sum test) also ranks the pooled sample, ensuring that if same ranks occur the mean rank is assigned to all candidates. For each sample X and Y, the sum of the ranks of its members is calculated, R_x and R_y respectively. The test statistic is $R_{Mann} = m(N+1) - R_x$, if $R_x < R_y$ or $R_{Mann} = n(N+1) - R_y$, if $R_y < R_x$. Again, the null hypothesis can be rejected for low values of R_{Mann}. Alternatively the U statistic can be used, as

$$U = min\{mn - R_x + m(m+1)/2, mn - R_y + n(n+1)/2\}. \tag{4.3}$$

For large samples the normal approximation and the z statistic can be used [122]

$$z = \frac{U - m_U}{\sigma_U}, \tag{4.4}$$

$$m_U = mn/2, \tag{4.5}$$

$$\sigma_U = \sqrt{\frac{mn(N+1)}{12}}. \tag{4.6}$$

4.3.2 Adapting statistical tests to higher dimensions

All these tests are typically defined for univariate samples and rely on sorting the samples in some way. In the multivariate case, however, i.e. when the samples X and Y comprise of multi-dimensional observations (such as the input of a multiple channel spectrometer), sorting the observations is not straight forward. There are a few variations of adapting the statistics to higher dimensions. For the Smirnov test, for instance, the cumulative distribution functions of the two samples can be compared with all possible orderings, and the largest of the set of resulting K-S statistics can be used [123] [124]. However, such approaches are associated with long computation times [124]. Instead, the strategy proposed by Friedman and Rafsky [118], which makes use of a minimal spanning tree (MST) in order to meaningfully rank or sort multidimensional data, is adopted. This is simpler both conceptually and in terms of implementation and can be applied to a multitude of univariate statistical tests. Originally in [118] the method is proposed for

the Smirnov and the Wald-Wolfowitz test, but in this work it is also used to extend the Mann-Whitney test to high dimensions.

Minimal Spanning Tree

From graph theory, a tree is a graph consisting of nodes and edges (connections between pairs of nodes) without cycles, which has a unique path connecting any two nodes. A weighted tree is a tree with weighted edges. In the case of numerical data, these weights (or lengths of the edges) correspond to some form of distance metric between the two nodes that are connected. The minimal spanning tree, is the tree with minimum weight, i.e. the tree with the minimum total length.

The creation of the minimal spanning tree starts from a random node which is connected to its nearest neighbour, according to the chosen distance metric. There are many distance metrics that can be used, such as the Euclidean distance, the Mahalanobis distance or the correlation. In this work, as the data is continuous, the Euclidean distance is used. The construction of the MST proceeds iteratively through the node of the tree that is nearest to any of the still unconnected nodes. The two nodes are connected and the process continues until all the nodes are connected, forming the tree. An example of constructing an MST in the two dimensional space is illustrated in figure 4.4.

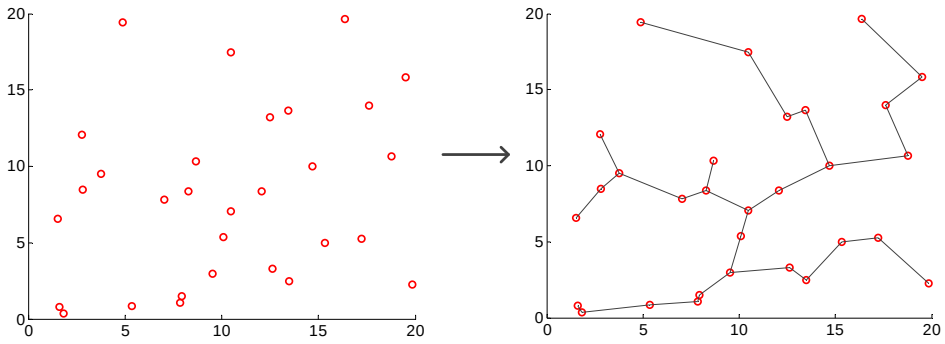


Figure 4.4: Construction of Minimal Spanning Tree in the two-dimensional space. The process starts with a connection between a random node and its nearest neighbour. Then iteratively the nearest neighbour to any node already belonging to the tree, is connected to it.

Height Directed Preorder Traversal and ranking of the nodes

Once the MST has been created, it can be used to rank the nodes, using a height directed pre-order traversal. Firstly the tree is rooted at its centre. This can be done by examining

the eccentricity of the nodes. The eccentricity of a node is the number of nodes included in the longest path starting from this node and ending at any other node in the tree. The degree of a node (in a rooted tree) is the size, in number of nodes, of the subtree originating at that node. The centre can be defined as the node with the minimum eccentricity. After the tree is rooted the nodes can be assigned ranks through a height directed pre-order traversal of the tree: traversal begins at the root which is assigned a rank of 1. The rest of the nodes are assigned ranks as they are visited. The traversal of the tree is done in a top-down manner and the principle is that when a node has more than one descendants, the one with the smaller degree is visited first. An example of this is illustrated in figure 4.5.

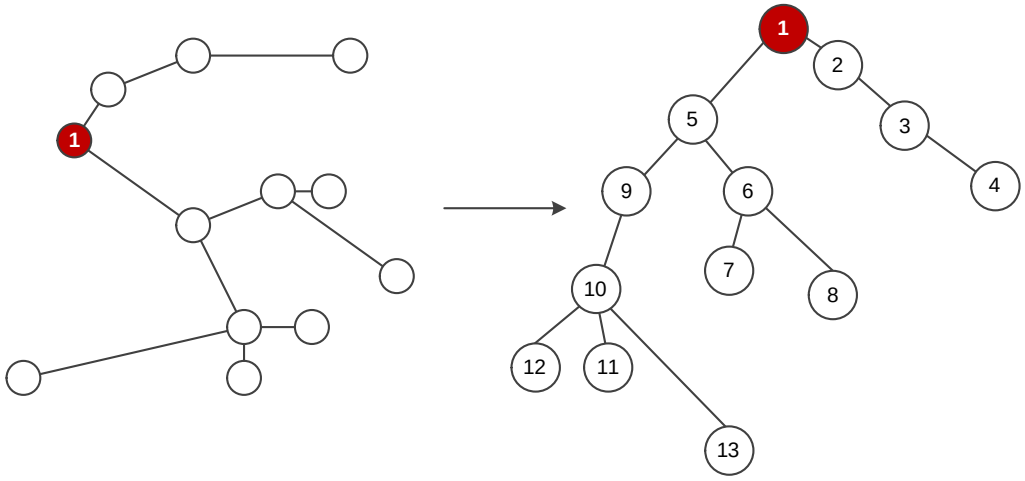


Figure 4.5: Height directed pre-order traversal. The number inside each node is its rank. Starting from the root, i.e. the center of the tree, at every level the node with the least descendants is visited (and ranked) first.

Adapted non-parametric tests

Once the nodes, i.e. the data points in the pooled sample, are ranked following the minimal spanning tree method, the Smirnov and Mann-Whitney statistics can be directly calculated as in the univariate case. For the Wald−Wolfowitz test the nodes are again assigned labels, depending on which sample they originated from. The quantity R, which was the number of runs, can now be defined as the number of edges that connect nodes with different labels (see figure 4.6). The lower the R statistic is, i.e. the lower the number of disjoint trees that would result from removing those edges, the more confidently the null hypothesis can be rejected. This makes the Wald-Wolfowitz test comparatively easier

to compute, as the height-directed traversal of the tree is not needed.

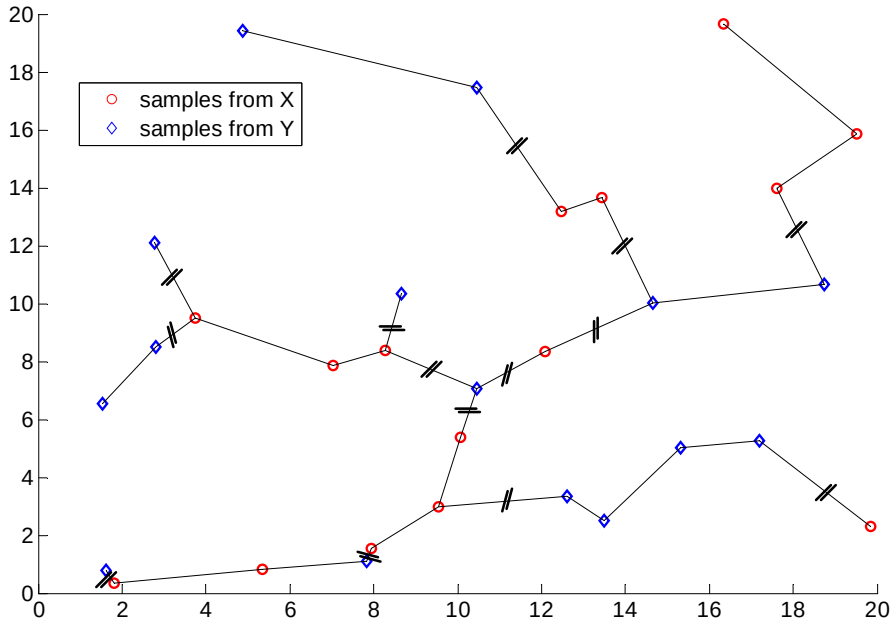


Figure 4.6: MST adapted Wald-Wolfowitz test for two dimensions (units are indicative). The number of runs, R, is equivalent to the number of branches that separate points belonging to X from points belonging to Y. In this example R=14.

4.3.3 Multivariate statistical tests - Maximum Mean Discrepancy

Instead of adapting univariate tests, there are statistical tests that are aimed specifically at multidimensional data. In [119], Gretton et al. propose a non-parametric framework for comparing two multivariate distributions, which uses a kernel function to transform the data into a lower dimension. The problem again entails having two samples X and Y, coming from distributions p and q respectively; the similarity between p and q is tested. If f is selected from a class of functions $F : X- > R$, as a "well behaved (e.g. smooth) function which is large on the points drawn from p and small (as negative as possible) on the points drawn from q" [119], then the test statistic, the Maximum Mean Discrepancy (MMD) can be calculated:

$$MMD[F, p, q] = sup_{f \in F}(E_{x \backsim p}[f(x)] - E_{y \backsim q}[f(y)])^1 \tag{4.7}$$

[1]Supremum of a subset S of a set T is the least element of T that is greater than, or equal to all the elements of S

The Maximum Mean Discrepancy can be defined as "the difference between the mean function values on the two samples" [119]. When the value of MMD is large, p and q are considered to be different, while MMD approaches zero when $p = q$.

Gretton et al. have calculated and used empirical estimates for the MMD statistic, by setting F to the unit ball in a reproducing kernel Hilbert space. Two estimates are calculated. The unbiased estimate, MMD_u, is given in formulas 4.8, 4.9 and the biased estimate, MMD_b, in formula 4.10, where $k(.)$ is a kernel function.

$$MMD_u^2[F, X, Y] = \frac{1}{m(m-1)} \sum_{i \neq j}^{m} h(z_i, z_j), \tag{4.8}$$

$$h(z_i, z_j) = k(x_i, x_j) + k(y_i, y_j) - k(x_i, y_j) - k(y_i, x_j) \tag{4.9}$$

$$MMD_b[F, X, Y] = (\frac{1}{m^2} \sum_{i,j=1}^{m} k(x_i, x_j) - \frac{2}{mn} \sum_{i,j=1}^{m,n} k(x_i, y_j) + \frac{1}{n^2} \sum_{i,j=1}^{n} k(y_i, y_j)) \tag{4.10}$$

In order to use these estimates in statistical hypothesis testing, critical values both for MMD_b and MMD_u are calculated in [119]. In the case where the unbiased estimate is used, the null hypothesis ($p = q$) can be accepted at a significance level α, when $MMD_u^2 < (4K/\sqrt{m})\sqrt{log(\alpha^{-1})}$. Similarly, in the biased case, the null hypothesis can be accepted when $MMD_b < \sqrt{2K/m}(1 + \sqrt{2log(\alpha^{-1})})$. In both cases, K is the upper bound of the kernel function: $0 <= k(x, y) <= K$ for every (x, y).

4.3.4 Statistical hypothesis testing using sliding time windows

As mentioned earlier, concept drift detection is the detection of a change in the distribution or model that generates the background data (see figure 4.1). When dealing with a time series, like the data coming from a chemical instrument, a common approach for concept drift detection is windowing [74] [82] [120].

At time t_0 two consecutive time windows are examined(as seen in figure 4.7), one with the n_1 most recent examples, W_1, and one with the previous n_2 examples, W_2. The statistical test is applied in order to determine if the data in the two windows come from the same distribution. In this work, detecting concept drift in data coming from a chemical instrument, in particular a spectrometer, is of interest. The sensor data, i.e. the measurements of the instrument, are denoted $\overline{u}_t$, and at every time point the measurement corresponds to an l channel spectrum $\overline{u}_t = \{u_{t,1}u_{t,2}...u_{t,l}\}$. If X corresponds to the data in window W_1 and Y corresponds to the data in window W_2, then $X = \{\overline{u}_{t_0-n_1+1}, ..., \overline{u}_{t_0}\}$

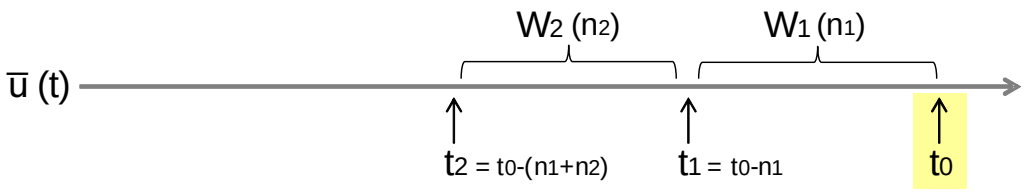


Figure 4.7: Sliding time windows. The two samples used in the statistical tests come from two consecutive sliding time windows. If X corresponds to the data in window W_1 and Y corresponds to the data in window W_2, the null hypothesis is that X and Y are generated by the same distribution. Rejecting the null hypothesis, means that the two windows are generated by dissimilar distributions. From this it can be inferred that concept drifted at time t_1.

and $Y = \{\overline{u}_{t_0-(n_1+n_2)+1}, ..., \overline{u}_{t_0-n_1}\}$ and the null hypothesis is that X and Y are generated by the same distribution. If the null hypothesis, H_0, is rejected then the generating distributions of the data in the two windows are found dissimilar. This effectively means that it is possible to detect if a change happened n_1 timesteps ago. When H_0 is rejected for a pair of windows, then concept drift is detected at time $t_1 = t_0 - n_1$.

One very important factor in this method is the size of the windows. Firstly, the size of the windows is related to the type of drift that needs to be detected; for instance, for slow drift, the windows have to be large enough, so as to capture the difference. Moreover, as this will be applied in combination with anomaly detection, the window size needs to allow the system to differentiate between an anomaly and a concept drift. The final concern is time and computational cost. The system should be able to react online and near real-time. As with this setup concept drift is not being detected at time t_0, but at time $t_0 - n_1$, the size of the window, n_1, has to be small enough so that this detection is not irrelevant by the time t_0. Moreover, the smaller the time window is, the quicker the test statistic will be calculated. On the other hand, a very small time window contains a lot of variance due to instrument noise, and it becomes difficult to capture the variance of the data that is caused by a concept drift. In that sense, there is a trade−off between the speed of detection and the accuracy.

4.3.5 Data and artificial concept drift

The data used to evaluate the effectiveness of the five different presented statistical tests (the Smirnov, Wald-Wolfowitz, Mann-Whitney, MMD_b and MMD_u) is the mass spectrometry data associated with the DSTL ICARIS 2009/2010 competition [117]. This is a timeseries of more than 200,000 timesteps (around 18 hours of data), each one corre-

sponding to a 270 channel spectrum. This data was collected under laboratory conditions using a highly sensitive time-of-flight proton-transfer mass spectrometer (PTRMS) with a data-rate of 3Hz. During the data collection, different substances are introduced to the sensor at different distances and intensities. These constitute the anomalies in the data. A total of seven different substances are introduced to the sensor a total of 245 times. Each of these labelled anomalies is loosely categorised as *weak*, *medium* and *strong*, depending on the exposure of the sensor to a substance at this particular instance. The detection of these anomalies has already been addressed using the RDA [5].

The focus of this chapter however is the detection of concept drift. The existing data does not contain concept drift, i.e. a change in the environment in which the data is collected. Such change would correspond to a robot continuously collecting data and performing anomaly detection while moving through changing environments. This is simulated, by adding artificial concept drift to the ICARIS mass-spectrometry data. As illustrated in figure 4.8, the situation that is being simulated is that of moving from a stable environment A, to a stable environment B, the two environments having different background spectral signatures. A background environment signature in this context refers to a steady state spectrum when only the normal, for this environment, background chemicals are present. It is assumed that the transition from one environment to the other is gradual, i.e. there is a transition period in which the normal background changes. Concept drift is this transition. The transition is simulated with the linear addition over a specific duration (the duration of the transition) of the signature B to environment A. This addition of the new background signature will be referred to in the following discussion, as an artificial event.

Enhancing the dataset with artificial events

Artificial concept drift, or an artificial event, $\mathbf{E}$, is denoted as the addition over a specific transition period, t_s to t_e, of a new background signature, of specific intensity, I_{max} to the existing data. An artificial event, $\mathbf{E}$, is based on a signature $\mathbf{S} \in R^l$, which is the pattern of an assumed background chemical element over an l channel spectrum, scaled to $[0, 1]$: $\mathbf{S} = \{s_1, s_2, ..., s_l\}, s_i \in [0, 1]$, for an illustration see figure 4.9. The concept drift, i.e. this event, is introduced to the background signal in a linear way in time and it is assumed that it is added to the previous spectrum.

An event $\mathbf{E}$ starts at t_s and ends at t_e, meaning that the signature is starting to

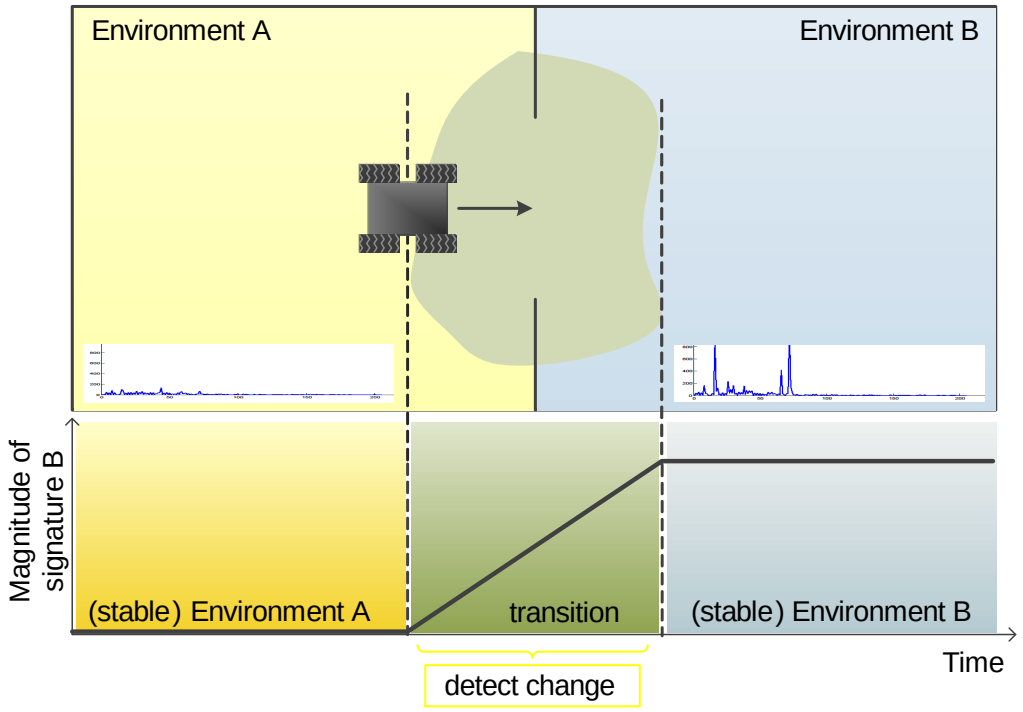


Figure 4.8: A robot moving between environments with different backgrounds. The transition between the two environments and subsequently the two different backgrounds is assumed to be gradual. A linear transition over a specified duration from background spectrum A to background spectrum B is assumed.

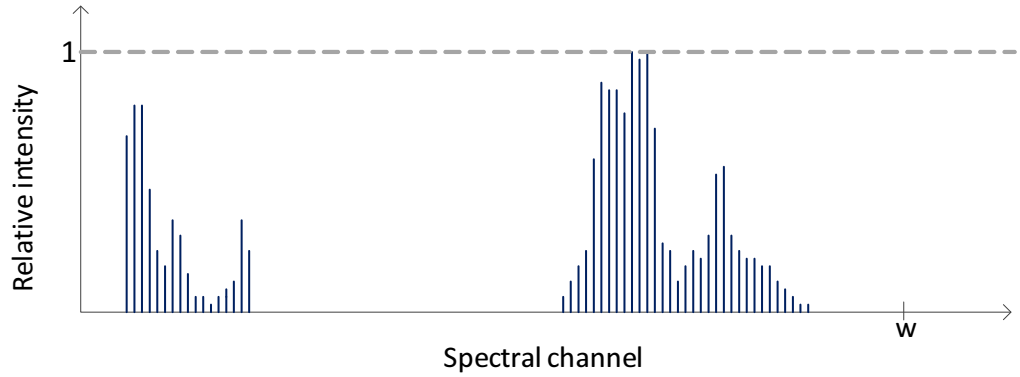


Figure 4.9: The signature of a possible new chemical background, scaled to $[0, 1]$.

appear linearly (in the time domain) at time t_s and at t_e and on it remains at its maximum intensity. The maximum intensity, or simply intensity, of the event, I_{max}, is defined as the intensity of the highest peak of the added background signature. For an event of intensity I_{max}, the scaled signature, $\mathbf{S}$, is multiplied with I_{max}, so that every channel has an intensity proportional to its relative intensity in the signature, when the highest peak has intensity I_{max}. The artificial event is described, at every timestep from t_s to t_e, by the following equation:

$$\mathbf{E}(t) = \mathbf{S} I_{max} \frac{t - t_s}{t_e - t_s}, \mathbf{E}(\mathbf{t}) \in R^l \tag{4.11}$$

This event is added linearly to the existing spectral data. If the existing spectral data, before the addition of any events, is D then D', the data after the addition of an event, is calculated as follows:

$$\mathbf{D}'(t) = \begin{cases} \mathbf{D}(t), & \text{if } t < t_s. \\ \mathbf{D}(t) + \mathbf{E}(t), & \text{if } t_s \leq t < t_e. \\ \mathbf{D}(t) + \mathbf{E}(t_e), & \text{if } t \geq t_e. \end{cases} \tag{4.12}$$

The process of adding an artificial event to the existing mass spectrometry data is shown graphically in figure 4.10.

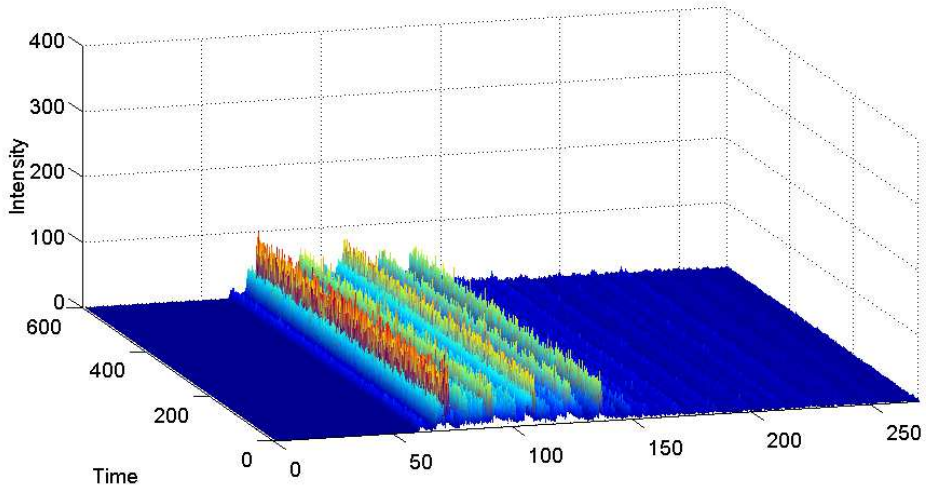
Signature generation

The events that have been artificially added to the data are based on different, random signatures. All the experiments have been repeated on 18 different signatures, to ensure that the observed behaviour is not caused by signatures of specific shapes and involving specific spectral channels. Six patterns have been used that cover different parts and proportions of the spectrum as seen in figure 4.11. Each of these patterns gives rise to three signatures, created by assigning random values only to the spectral channels indicated by the corresponding pattern (figure 4.12). All the signtures are scaled in the range $[0, 1]$.

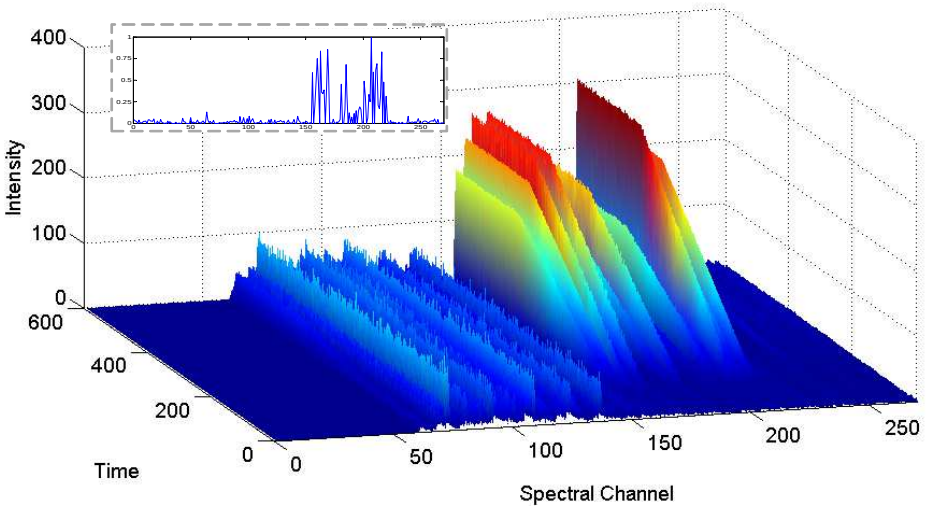
4.4 Experiments

A series of experiments were performed, without **(A)** or with **(B)** anomalies present, using the statistical hypothesis testing approach, to investigate:

(Q.A1) The effectiveness of different methods when the window size is varied.
(Q.A2) The effectiveness and the differences of the methods in different types of concept drift (gradual - abrupt).
(Q.A3) The role of the window size on computational time.
(Q.B1),**(Q.B2)** The performance of concept drift detection when anomalies are present



(a) Data before artificial concept drift



(b) Data after artificial concept drift

Figure 4.10: Example of linear concept drift addition. Illustrated here is a portion of the mass spectrometry data; 600 timesteps over all 270 channels. The artificial event start and end times are $t_s = 100$, $t_e = 300$, while its maximum intensity is $I_{max} = 300$. On the top (a) is the original data from the ICARIS dataset. On the bottom (b) is the data after the linear addition of the event. The artificial event is based on the signature depicted on the top left of (b).

in the data and the effect of tuning the detection thresholds

(Q.B3) The RDA in the presence of concept drift.

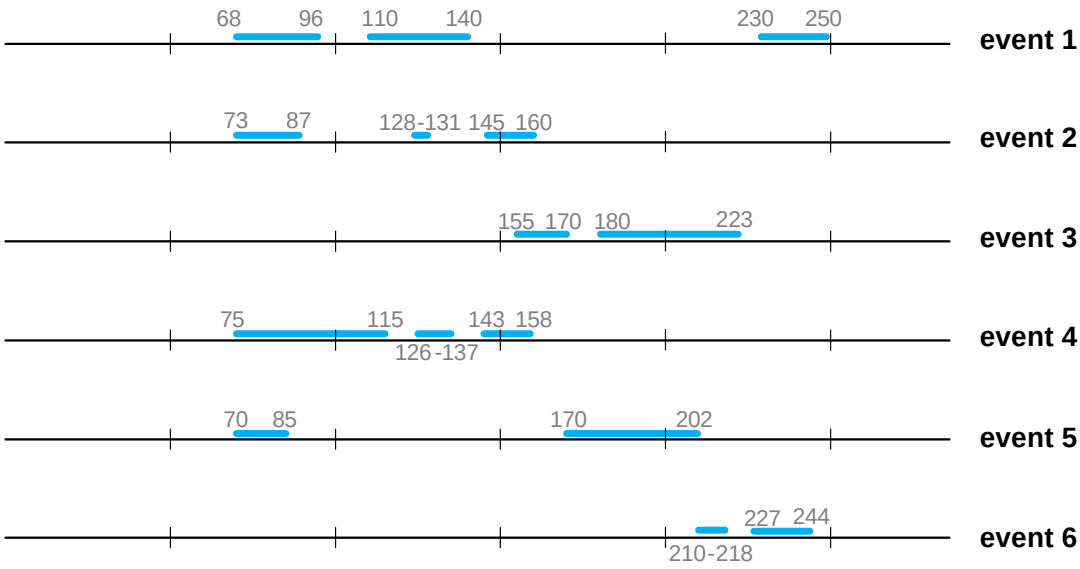


Figure 4.11: Patterns used for random signatures. The noted areas are the spectral channels, over the whole spectrum (270 channels) that will be part of the signature.

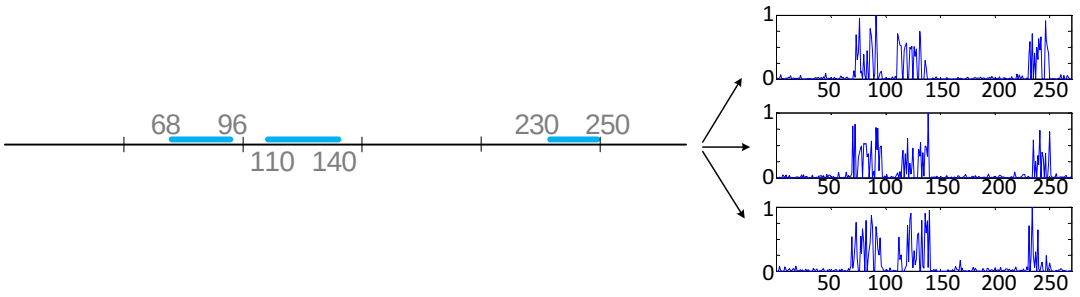


Figure 4.12: Signatures created from pattern. Every pattern gives rise to three signatures. Every signature is created by assigning values drawn from a uniform distribution to the channels involved in the pattern.

4.4.1 Drift detection thresholds

To compare the methods fairly, the detection thresholds have been set according to a significance level of $\alpha = 0.05$. For the MMD_b and MMD_u a Gaussian kernel is used: $k(x_i, x_j) = (1/\sqrt{2\pi})exp(-|x_i - x_j|^2/2\sigma^2)$, with σ set to the median of the pooled sample $(X \bigcup Y)$, according to guidelines set in [119]. This means that the upper bound, K is $1/\sqrt{2\pi}$. Therefore, the detection thresholds, $(4K/\sqrt{m})\sqrt{log(\alpha^{-1})}$ and $\sqrt{2K/m}(1 + \sqrt{2log(\alpha^{-1})})$, for MMD_u and MMD_b respectively can be calculated for this K and $a = 0.05$. For the Mann-Whitney test, the Matlab statistics toolbox is used, and the threshold set through that. For the Wald−Wolfowitz and Smirnov test (for those, the lower the statistic the more confident the rejection of the null hypothesis), the thresholds are qualitatively set to 5% of the statistic's range, to correspond to the 0.05 significance

level. That is $N/20$ for the Wald−Wolfowitz test, where N is the size of the pooled sample and 0.9 for the Smirnov test.

4.4.2 A normal dataset

Anomalies are present in the ICARIS dataset. For the first phase of the experiments, the detection of concept drift only is investigated, i.e. with no anomalies present. For this purpose, a "normal" dataset has been constructed, from anomaly free portions of the ICARIS data. For the second phase of the experiments the concept drift detection methods are tested on the full dataset, to determine how the two algorithms (the anomaly detection - RDA - and the concept drift detection) interact. In both cases (data without or with anomalies) the data set has been enhanced with artificial events that represent concept drift.

Artificial data and after-drift environment

The following discussion provides some context by analysing the ICARIS dataset and justifying why a specific range of intensities for artificial concept drift is selected. In the available mass spectrometry data, anomalies can range from 110 to 10353 in intensity (this is the maximum intensity reached over the duration of this anomaly or detection). In the label files that accompany the ICARIS data, every anomaly has been associated with a strength level (low, medium or high) according to the exposure of the mass spectrometer to the particular chemical substance that constitutes this anomaly. The range of intensities that correspond to each of these levels are depicted in figure (4.13). Running the RDA on this dataset produces a number of false positives; using the RDA original parameter set, as determined in [5], there are seven false detections which are also included in figure 4.13.

It is assumed that concept drift is a background signature that can cause the RDA to produce a detection, i.e the intensity considered for the concept drift has to be in a sub-range of the anomalies intensities range. The high intensity anomalies usually correspond to very high concentrations of a substance. When the system moves to a new environment, it is assumed that the background will be different and will have a background signature in an intensity range that cannot be ignored by the RDA (i.e. not lower than 100). On the other hand, a new stable environment would mean a steady dispersed background, so very high intensity background signatures would not agree with this premise, as they are more

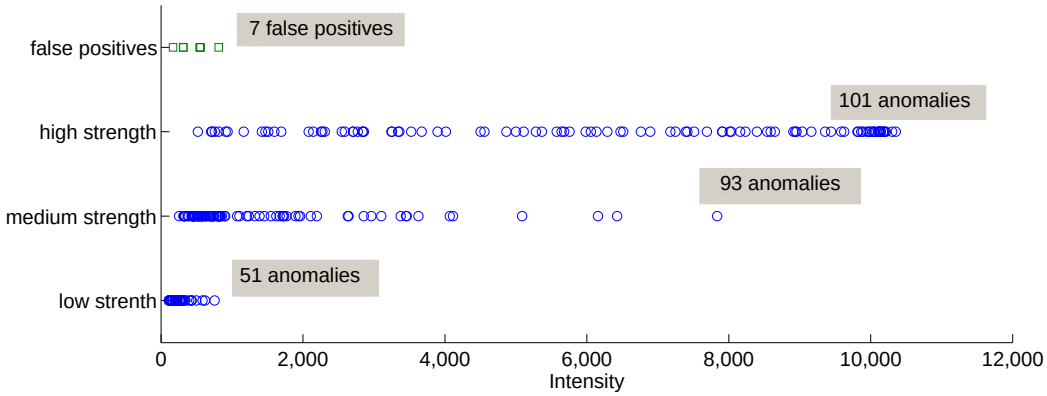


Figure 4.13: Intensities of labelled anomalies in relation to their strength as labelled. Also depicted are the intensities of the false positives that were produced by the RDA in its original tuned implementation.

likely to be associated with a strong localised source of a specific substance, rather than an even background. Therefore, a lower intensity would be more appropriate to simulate a steady background environment. A medium-low intensity region is also a very interesting region, because the RDA is challenged at these intensities. At this range, an incoming anomaly might be missed, or some non-anomalous event (such as opening a door in the lab) might be tagged as an anomaly. The high intensity anomalies are easily detected. The optimisation of the RDA typically focuses on making the algorithm sensitive enough to detect the low intensity anomalies, without producing many false positives. Taking those considerations into account, artificial events of concept drift have been examined, that lay in the 300-1000 region of intensities. The transition period, i.e. how fast the new background signature reached its maximum intensity is also of interest in this chapter. In the experiments carried out this is varied to determine the limits of the concept drift detection methods tested.

4.4.3 Evaluation

For the first part of the experiments, the performance of each method is to be determined and compared when the window size or the speed of the event are varied. The concept drift detection performance is measured as the accuracy of the detection. The accuracy is defined as the number of windows where the drift event is detected, over the total duration of the artificial event (t_s to t_e) in number of windows. The false positive rate, which is the number of windows with falsely detected drift over the total duration of the dataset in windows, is also monitored. A typical detection output of a test and the calculation of

accuracy and false positive rate is demonstrated in figure 4.14.

For the second part, a binned version of the same graphs is used, where the existence of a bar denotes the detection of concept drift. In this phase the performance of two modules of the system are simultaneously monitored: the anomaly detection module, which is the RDA, and the concept drift detection module, which is the statistical hypothesis testing methods. To quantify this, a double truth table is used, to show the number of windows (over the whole dataset) where each of the two algorithms makes a detection and where the detection is attributed (the detection can be caused by an artificial event, an anomaly, or noise in the data).

The results that will be presented are averaged over 18 events created from the signature library discussed in the previous sections.

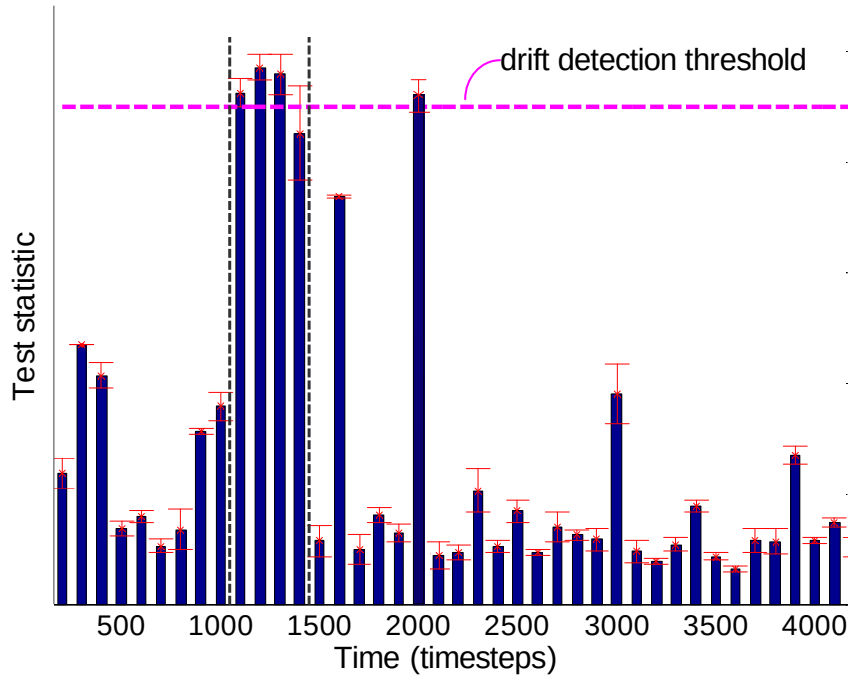


Figure 4.14: Evaluation of the different statistical methods. Every bar in the graph is the test statistic (it applies to any of the tests) for the respective time window. Time is shown in timesteps (of the data file). It is reminded that the sampling frequency is 3Hz, therefore every timestep is 0.33 sec. The accuracy is the number of windows where the drift event is detected, over the total duration of the artificial event in number of windows, noted by the grey vertical lines. The false positive rate (FPrate) is the number of windows with falsely detected drift, over the total duration of the dataset in windows. In this example the accuracy is 0.75 (3/4 windows) and the FPrate is 0.025 (1/40).

4.5 Results

4.5.1 Data with concept drift only (no anomalies present)

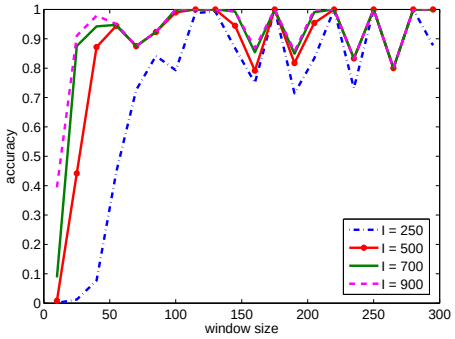
The first phase consists of examining the efficiency of concept drift detection through statistical hypothesis testing, on the artificially enhanced data, coming from the mass spectrometer. No anomalies are included in the dataset for this phase.

Comparison of methods for varying window size [Q.A1]

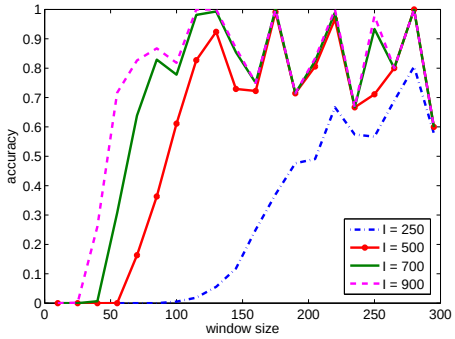
The first set of experiments focuses on the role of the window size. For every experiment, the duration of the drift event is kept constant at $t_e - t_s = 1000$, and the intensity is set to 4 different values ($I_{max} = \{250, 500, 700, 900\}$). For every intensity level, the window size is varied from 10 to 300 timesteps. The results for each method separately are depicted in figures 4.15a−4.15e.

For four out of five methods (with the exception in some cases of the Mann-Whitney test) similar trends are observed: for events with high intensity ($I = 900$ or $I = 700$) there is a local maximum in accuracy achieved for a window size ranging from 100 to 130. This window size is big enough to contain enough change due to the artificial event, but small enough (in comparison to the duration of the drift which is 1000) to precisely detect it. Events with lower intensities ($I = 250$ or $I = 500$) are harder to detect and high accuracy can be achieved for larger windows. For all the intensities tested, using very small window sizes does not effectively capture the change in consecutive windows; very low accuracy, less than 50% is generally achieved for windows smaller than 50 timesteps. The exception to this is the Mann-Whitney test, which is more sensitive and detects concept drift even for small windows. However, this comes at the expense of the false positives rate, which is significantly higher for this method than for the others.

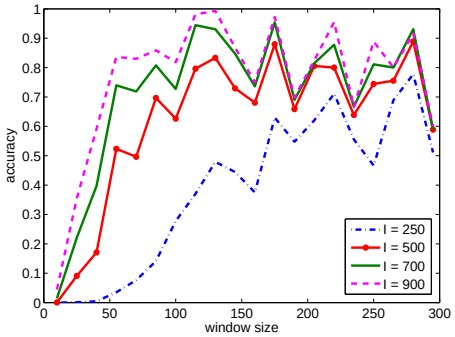
One interesting observation is the instability in the accuracy, shown by a rugged effect, particularly for windows larger than 150 timesteps. The fact that this behaviour is consistent for all the methods and intensities indicates that it is not an artefact of a particular method. It is believed that this effect is due to the placement of the windows. The artificial event always starts at timestep 1000, but as the window size is varied, the start of the first (or last) window where concept drift is present, in relation to the beginning (or end) of the artificial event, varies. If for example the first window where concept drift is present contains timesteps 795-1060 (as is the case for a window of 265), then only 60 timesteps



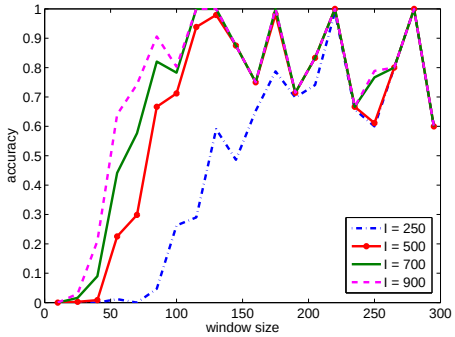
(a) $\mathrm{MMD_b}$



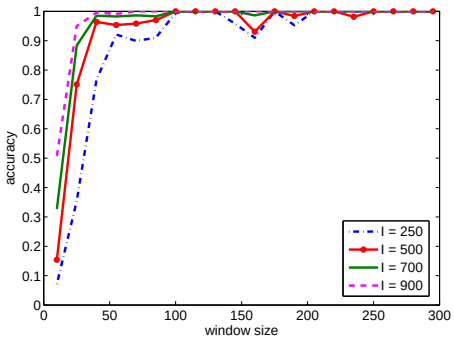
(b) $_{\mathrm{u}}$



(c) Smirnov



(d) Wald−Wolfowitz

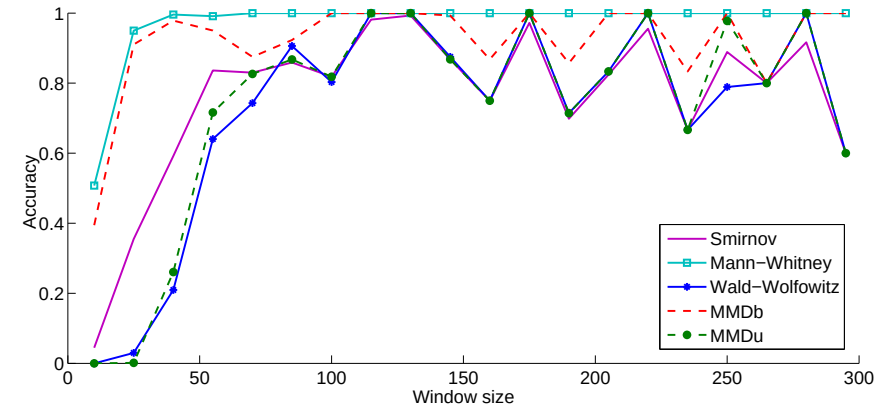


(e) Mann−Whitney

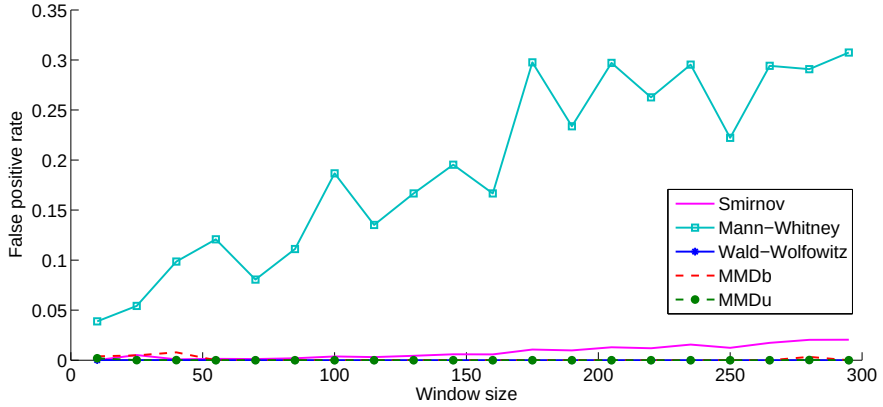
Figure 4.15: Accuracy of all methods over the window size, for different intensity levels. For every method (subfigure) the accuracy for $I_{max} = 250, 500, 700, 900$ is presented. The duration of concept drift is 1000 timesteps. A general observation is that the accuracy (percentage of windows where drift is detected) is higher for larger windows (larger than 100 timesteps). Also larger windows are needed for the accurate detection of very gradual concept drift (low intensity of 250 which corresponds also to low speed of 0.25).

(1000 to 1060), i.e. 22% of the window actually contains data where a new background signature has started being added. This is not enough change for the statistical methods to detect a significant difference between this window and its preceding window, which

contains only normal/pre concept drift data. A similar problem can be observed when only a small portion of the last window containing concept drift actually contains the end of the artificial event. As a result, the accuracy is decreased, as concept drift is missed in one or two windows.



(a)



(b)

Figure 4.16: Accuracy (a) and false positives rate (b) of all methods over the window size, for $I = 700$. The results from the experiments that vary the window size are presented comparatively for all methods for only one intensity level. The duration of concept drift is 1000. The highest accuracy over all window sizes (excluding the Mann-Whitney test) is achieved for MMD_b, while the other methods perform comparatively. The Mann-Whitney test is exluded because of the very high observed false positive rate.

In figures 4.16a and 4.16b the accuracy and false positives of all the methods are illustrated for concept drift of intensity $I_{max} = 700$. The data plotted can be inferred from figures 4.15 but is presented here for a more clear comparison of the methods for a given intensity. For the rest of the intensities the results follow the same trends; for the interested reader, the figures for all the intensities ($I_{max} = [250, 500, 700, 900]$) are

presented in appendix A.1.

Although the Mann−Whitney test outperforms the other methods, the number of false positives it produces is orders of magnitude higher than the false positives rate of the other methods: for a window of 100 timesteps the Mann-Whitney method has an FP rate of 18.7%, while the FP rates of the other methods range from 0 to 0.3%. Since the FP rate of the other four methods is very low, under 2%, their performance is determined mainly in terms of accuracy. Excluding the Mann-Whitney method (because of its high FP rate), the MMD_b results in the best performance: high accuracy, of 97.8% is achieved for windows as small as 40 timesteps. Additionally, the accuracy of the MMD_b is higher than the other methods across the whole range of the tested window sizes. For windows larger than 50 timesteps the remaining methods converge and achieve accuracies around 80%. As explained earlier subsequent decreases are due to the window placement effect.

Comparison of methods for varying event speed [Q.A2]

The next experiments have been carried out to determine the effectiveness of the different methods, when the speed of an event is varied. The speed of an event, $I_{max}/(t_e - t_s)$, determines how gradually or abruptly an event is introduced into the normal data. In the first set of experiments the duration of the event, $t_e - t_s$, is kept constant at 300 and the intensity is varied; by increasing the intensity, when the duration of the event is constant, the speed is increased. The window size for these experiments is set to 100. Figure 4.17 shows that, the more abrupt an event is the easier it is to detect. This is true for all the methods and it is to be expected, because the change is greater from one window to the next, hence it is easier to detect. The kernel based MMD_b method and the Mann-Whitney test (again at the expense of high false positive rate) achieve a perfect accuracy (100%) at speed 0.7, which corresponds to gradual drift and an intensity of 210. The other three methods perform relatively poorly for speeds smaller than 1.5 ($I_{max} = 450$), but converge to accuracies close to 80% for more abrupt events.

Figures 4.18a−4.18e support the previously reported results, but they also showcase the variance in the results. For this event of medium speed (1.3), the Mann-Whitney test and the MMD_b method have the best results in terms of accuracy, but the Mann−Whitney test, apart from the false positives, has a very high variance. In contrast to that, the MMD_u, the Wald−Wolfowitz and the Smirnov test, have low variance, but they struggle in detecting the beginning and the end of the drift; hence the reported low accuracy. These

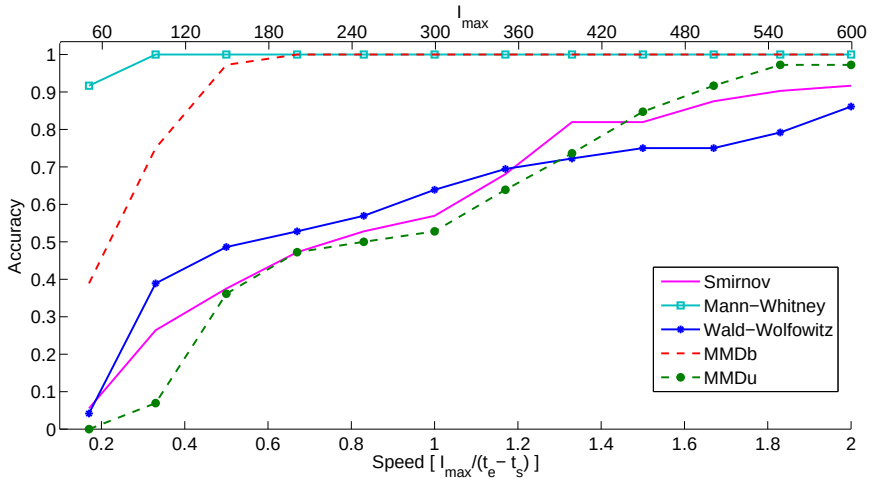


Figure 4.17: Accuracy over the speed of an event (varying the intensity of the event, I_{max}).

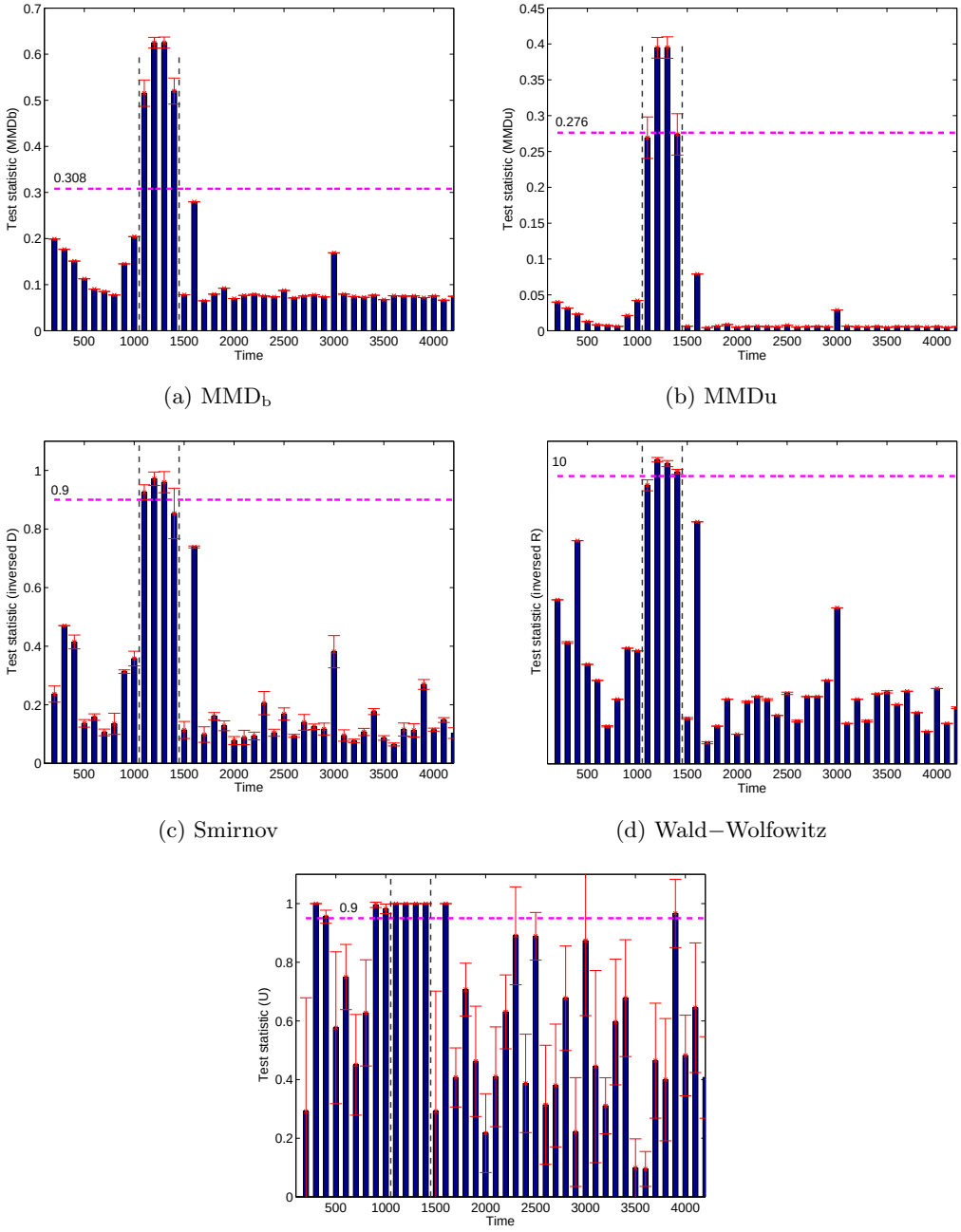
observations apply to the range of speeds tested. The corresponding graphs for the rest of the speeds/intensities are included in appendix A.2.

The speed can also be varied by altering the duration of the event, while keeping its intensity constant. This has been done in order to confirm the previous conclusions irrespectively of the intensity. As illustrated in figure 4.19, the trend of increasing accuracy with increasing speed is present here as well. However, the decrease of accuracy for very high speeds (higher than 2) should be noted. In this set of experiments, high speed corresponds to a small duration, in particular the last point corresponds to 50 timesteps. This duration is smaller than the window size; if only 50% of one window contains change compared to the next window, the difference between the two can be difficult to detect.

Effect of window size on computation time [Q.A3]

The window size effectively determines the size of the samples to be compared, therefore controlling the computational time required for processing each pair of consecutive time windows. This can be seen in figure 4.20, where the processing time per window size is depicted[2]. This figure shows that the processing time needed at the end of each time window heavily depends on the size of the windows, especially for the kernel based methods, MMD_b and MMD_u. The MMD estimates are more computationally intensive than the other three methods that rely on the creation and traversal of the minimal spanning tree, that is why they are affected more by the sample size. For the other three methods the

[2]The exact times reported depend on the system specifications. These experiments were run on an Intel Core i5-2400 CPU @ 3.10Hz, 8GB RAM, running 64-bit Windows 7



(a) MMD_b (b) MMDu

(c) Smirnov (d) Wald−Wolfowitz

(e) Mann−Whitney

Figure 4.18: Detection of concept drift for all methods. The event, of intensity $I_{max} = 400$, in this example is introduced at timestep 1000 and lasts for 300 timesteps, as indicated by the vertical grey lines. The horizontal line is the detection threshold for each method. The bars correspond to the time windows and each bar represents the value of the test statistic as calculated at the end of the window. When the test statistic exceeds the threshold, concept drift is detected for this window. Note that for the Smirnov and Wald−Wolfowitz methods, the results have been inverted for illustration purposes (for these methods drift is detected for low values of the test statistic).

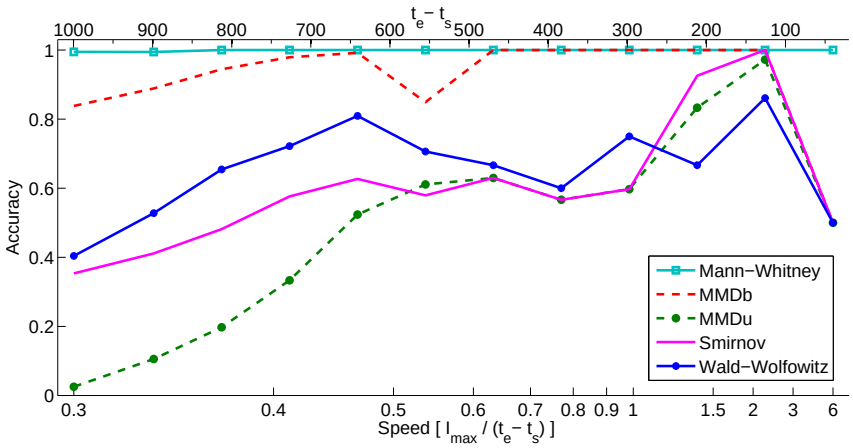


Figure 4.19: Accuracy over the speed of an event (varying the duration of the event, $t_e - t_s$). The intensity of the event is kept constant at $I_{max} = 300$

computational cost is associated with the creation of the MST and the traversal of the tree in order to rank the nodes. The Mann-Whitney and the Smirnov test have the same cost (in sec/window), while the Wald-Wolfowitz is slightly faster, as the number of runs can be calculated without the need to traverse the tree.

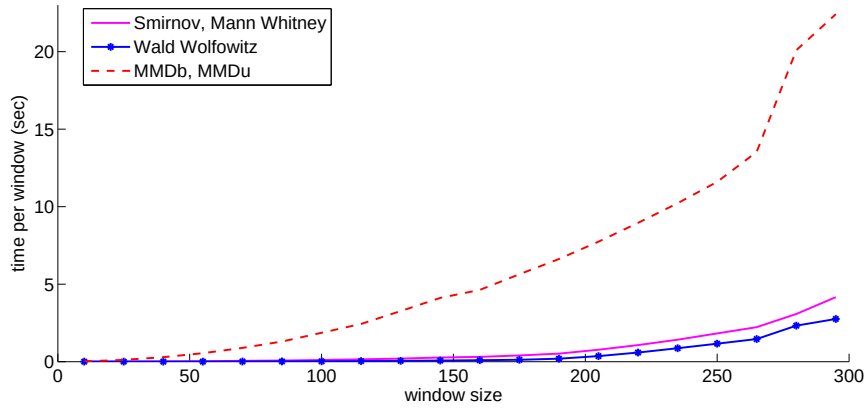


Figure 4.20: Processing time needed per window. Here, the time is given in seconds, but that clearly depends on the processing and memory specifications of the system. The trend, however, should be the same in any case.

4.5.2 Detecting concept drift and anomalies simultaneously

In the previous section the effectiveness of the methods under investigation for events of varying characteristics and for varying window sizes was determined. In those experiments no anomalies are present, but only concept drift. For this second phase of the experiments the original dataset is used, which contains anomalies. This dataset is also enhanced with

artificial concept drift. In this case, an event is periodically added and removed from the data (see figure 4.21). The duration of the artificial drifts is 1000 timesteps, while the intensity is varied from 300 to 900. The more challenging, relatively gradual concept drift is examined here; the speed of the event ($I_{max}/(t_e - t_s)$) varies from 0.3 to 0.9 (refer to figure 4.17). The window size is set to 110, as this results in generally good performance for both methods. Here, results are reported for the two most distinct and interesting methods: the highly accurate MMD_b and the very fast Wald-Wolfowitz method.

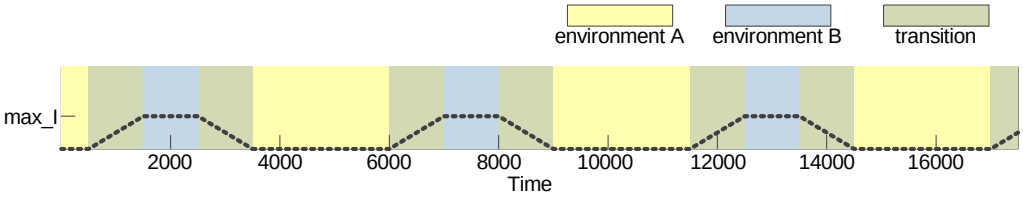
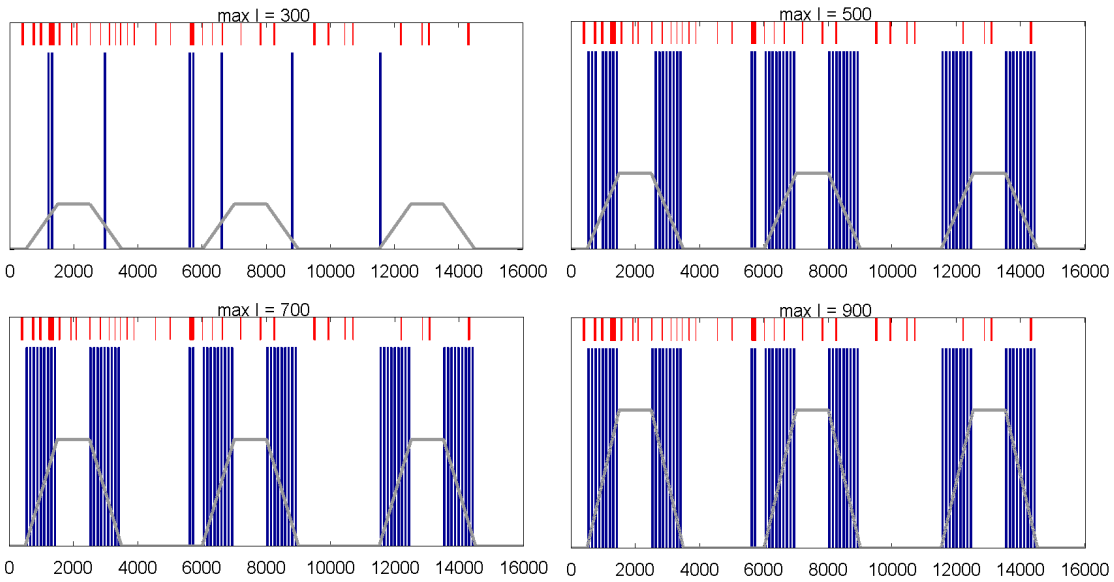


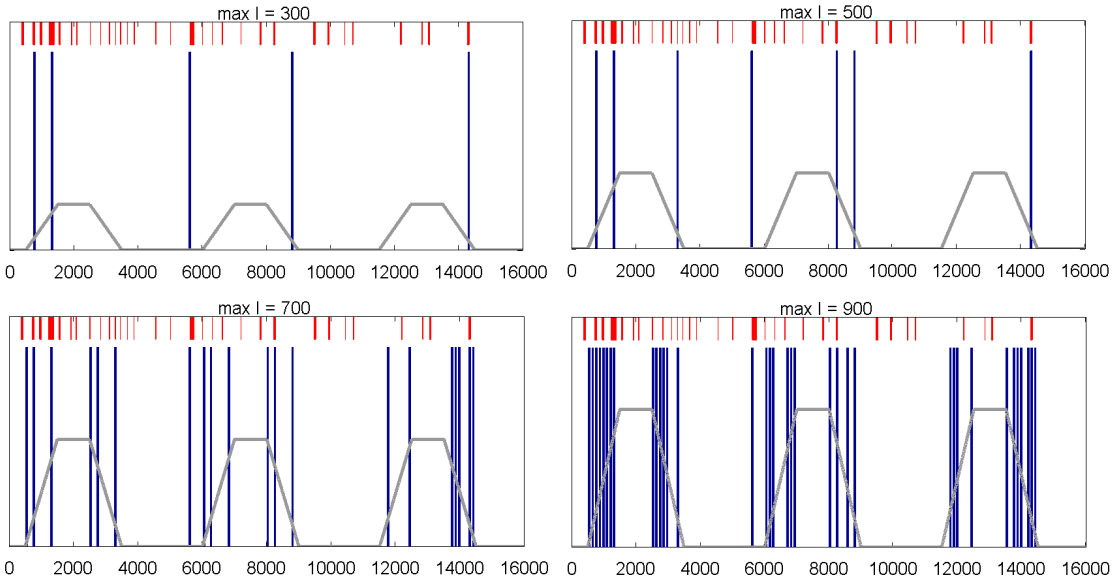
Figure 4.21: Time profile of concept drift in data with anomalies. An artificial event is periodically added and removed from the data. This corresponds to a robot moving between two environments A and B with different backgrounds. Environment A has the background corresponding to the original data (no added signature), while for the transition to environment B an event based on a background signature is added or removed as required. The depicted pattern of adding the artificial events is applied on the entire dataset (all seven file comprising the ICARIS mass spectrometry dataset).

Concept drift detection in the presence of anomalies [Q.B1]

Figures 4.22a and 4.22b show a portion of the data, which is exemplar to how the two methods work. The existence of a blue (bottom) bar means that concept drift is detected in this window, while the grey dashed line shows the actual rise and fall of the artificial events (concept drifts). The anomalies present in the data are marked by the red (top) bars. Ideally, the concept drift detection method should only detect drift in the windows corresponding to the increase or decrease of the added artificial event's intensity (marked by the grey line). The results however, especially for very low intensity/ very gradual drifts do not indicate a perfect performance. Both methods fail for very low and gradual drift ($I = 300$) and both achieve very high accuracy for intense drift ($I = 900$). In the in-between intensities, the MMD_b outperforms the Wald-Wolfowitz, as it achieves very high accuracies for intensities of 500 and higher (speed 0.5). The presence of anomalies has very little impact on the concept drift detection results (notice one false positive around $t = 5700$ coinciding with an anomaly for both methods), which means that both methods detect concept drift and not anomalies.



(a)



(b)

Figure 4.22: MMD_b (a) and Wald-Wolfowitz (b) performance in the presence of anomalies. A portion of the data is presented here (16000 timesteps) No units are displayed here on the y axis, because the results are binned. The presence of a bar indicate the existence of an anomaly (top, red bar), or a detection of concept drift (bottom, blue bar) in the corresponding window.

To investigate in more detail how the presence of anomalies affects the concept drift detection, and similarly how the presence of concept drift affects the anomaly detection (RDA), a double truth table is constructed, showing the actual events in the data (windows containing concept drift only, anomalies only, both or nothing), and how they are

detected. The tables presented here show the number of windows in the entire dataset that correspond to every case. If both the anomaly detection and the concept drift detection, work perfectly, then the ideal truth table would be table 4.1, which only has elements in its diagonal, i.e. correct classifications. However, if the performance of the RDA is not perfect, in one window there can be multiple RDA outcomes; for instance, some anomalies might be correctly detected and some missed. That is why in a few cases, the same window can be counted in multiple cells. In the case of false positives of the RDA, i.e. detections of the RDA which do not correspond to an actual anomaly, these numbers are reported in brackets, as for one window there can be multiple false detections by the RDA.

True \ Detected	Cdrift	Anomaly	Both	Normal
Cdrift	308	0	0	0
Anomaly	0	119	0	0
Both	0	0	92	0
Normal	0	0	0	372

Table 4.1: The ideal double truth table, if the concept drift detection and the anomaly detection work perfectly. Each entry is the total number of windows (over the entire dataset) that correspond to the particular case.

	MMD_b(I=300, not tuned)			
True \ Detected	Cdrift	Anomaly	Both	Normal
Cdrift	28.78	3.67	1.61	274.00
Anomaly	1.00	100.39	2.00	16.61
Both	0.56	76.94	4.61	10.83
Normal	10.00	0.17	0.50	362.00

(a)

	Wald-Wolfowitz(I=300, not tuned)			
True \ Detected	Cdrift	Anomaly	Both	Normal
Cdrift	8.11	3.78	1.50	294.67
Anomaly	0.00	101.39	1.00	17.61
Both	0.00	76.78	4.78	11.39
Normal	3.00	0.67	0.00	369.00

(b)

Table 4.2: The MMD_b (a) and Wald-Wolfowitz (b) methods for an event of intensity $I = 300$, added on data with anomalies. Every time window can contain concept drift, anomaly, both or nothing. Each entry is the total number of windows (over the entire dataset) that correspond to the particular combination of the concept drift and RDA detections, given the true event, averaged over 18 runs. The thresholds for concept drift detection correspond to a 5% significance level (see section 4.4.1).

Table (4.2) - again averaged over 18 different signatures - shows the concept drift /

anomaly detection results for the low intensity, $I = 300$. The tables support the findings shown in figures 4.22a and 4.22b: the concept drift is mainly classified as normal (highlighted in table 4.2, cell $[1, 4]$), i.e. it is not detected. In the case of the MMD_b, concept drift is detected on 30.39, out of 308, windows on average. This is a sum of the cases where only concept drift is detected (28.78) and the cases where there is a detection of both concept drift and anomalies (1.61). In the latter case, this is a correct detection for the statistical tests, but a false positive for the RDA. Similarly, in the case of the Wald-Wolfowitz test, concept drift is detected on only 9.61 windows on average. The detection of concept drift is poor; however, there are very few false positives, while anomalies are not falsely detected as concept drift.

	MMD_b(I=300, tuned)			
True \ Detected	Cdrift	Anomaly	Both	Normal
Cdrift	273.17	0.17	5.11	29.61
Anomaly	3.00	80.22	22.17	14.61
Both	10.17	8.83	72.72	1.22
Normal	45.33	0.17	0.50	326.67

(a)

	Wald-Wolfowitz(I=300, tuned)			
Detected \ True	Cdrift	Anomaly	Both	Normal
Cdrift	273.11	0.00	5.28	29.67
Anomaly	2.00	89.83	12.56	15.61
Both	10.06	9.67	71.89	1.33
Normal	22.06	0.17	0.50	349.94

(b)

Table 4.3: The MMD_b (a) and Wald-Wolfowitz (b) methods for an event of intensity $I = 300$, added on data with anomalies. Every time window can contain concept drift, anomaly, both or nothing. Each entry is the total number of windows (over the entire dataset) that correspond to the particular combination of the concept drift and RDA detections, given the true event, averaged over 18 runs. The thresholds are tuned to 0.17 for MMD_b and 55 for the Wald-Wolfowitz tests.

Tuning the thresholds to improve performance [Q.B2]

In these experiments, the detection thresholds are set to those corresponding to an $a = 0.05$ significance level, 0.308 for the MMD_b and 11 for the Wald-Wolfowitz method. However, if the thresholds are tuned, the results can be significantly improved, especially for the Wald-Wolfowitz method. When the thresholds are tuned by hand, the improvement for both methods can be seen in table 4.3. The cases where windows containing concept drift

are correctly detected is increased, at the expense, however, of some false positives. As seen in the highlighted cells, in this case the Wald-Wolfowitz test outperforms the MMD_b, as it produces almost half the false positives: 22.06 versus 45.33 instances of normal windows, classified as concept drift and 12.56 versus 22.17 windows actually containing only anomalies classified as both concept drift and anomaly.

For higher intensities of concept drift, tuning the thresholds has a positive effect mainly on the Wald-Wolfowitz method, while it is not necessary for the MMD_b method, which achieves high accuracies even for medium intensitites/speeds. Table 4.4 compares the non-tuned MMD_b and the tuned Wald-Wolfowitz methods. Note the high number of concept drift windows detected as concept drift or both concept drift and anomaly (a sum of 283.61 windows from cells $[1,1]$ and $[1,3]$) and the low number of false positive windows for concept drift detection. These are the normal windows, classified as concept drift or both anomaly and concept drift (cells $[4,1]$ and $[4,3]$ respectively), or the instances of windows with anomalies classified as concept drift or both anomaly and concept drift (cells $[2,1]$ and $[2,3]$ respectively). The Wald-Wolfowitz performs comparably, or even slightly better, with 291.33 windows with concept drift detected (cells $[1,1]$ and $[1,3]$), with a comparable number of false positives.

	MMD_b(I=900, not tuned)			
True \ Detected	Cdrift	Anomaly	Both	Normal
Cdrift	233.22	18.72 (76.56)	50.39 (155.44)	19.39
Anomaly	1.00	102.56	2.00	14.44
Both	9.78	11.83	70.22	1.06
Normal	6.44	140.72 (684.22)	4.56 (12.56)	260.72

(a)

	Wald-Wolfowitz(I=900, tuned)			
True \ Detected	Cdrift	Anomaly	Both	Normal
Cdrift	239.11	16.89 (72.28)	52.22 (159.72)	13.50
Anomaly	1.06	100.39	4.17	14.39
Both	9.78	9.94	72.11	1.06
Normal	9.56	139.56 (679.00)	5.72 (22.06)	257.61

(b)

Table 4.4: The MMD_b (a) and Wald-Wolfowitz (b) methods for event of intensity $I = 900$, added on data with anomalies. Every time window can contain concept drift, anomaly, both or nothing. Each entry is the total number of windows (over the entire dataset) that correspond to the particular combination of the concept drift and RDA detections, given the true event, averaged over 18 runs. Only the Wald-Wolfowitz thresholds is tuned at 38.5.

The RDA in the presence of concept drift [Q.B3]

Looking again at tables 4.2 and 4.3, it can be seen that the RDA has some tolerance to low intensity artificial events. For low intensities of concept drift very few non-anomalous instances are classified as anomalies. This can be seen in the anomaly column (second), where only windows containing anomalies, or both concept drift and anomalies, have caused the RDA to detect anomalies. For high intensities however (table 4.4), the RDA essentially stops detecting anomalies and produces a high number of false positives. Note the number of windows where normal instances classified as anomalies (cell $[4, 2]$), or concept drift is classified as both (cell $[1, 3]$). In one window there can be more than one false detections. The actual number of RDA false detections that correspond to these windows is reported inside the brackets. The RDA false positives are attributed both to the transition period (windows that contain concept drift), but also to "normal" windows corresponding to environment B (see figure 4.21), where the added background signature is present at the constant intensity I_{max}. For the latter case, the number of false detections (cell $[4, 2]$) is 676 (for MMD_b) to 679 (for Wald-Wolfowitz)[3], when the total number of labelled anomalies in the ICARIS dataset is 245. This behaviour is expected, as there is no adaptation in response to the concept drift.

4.6 Summary and Discussion

The results show that it is possible to detect concept drift using statistical hypothesis testing. No one method outperforms all the others in every aspect. The Mann-Whitney test detects a concept drift event accurately, but it also produces a much higher false positive rate than the rest of the methods. However, for small window sizes, this rate could be tolerable, depending on the application. The MMD_b, which has a comparable performance (apart from the very low intensity/speed cases) is slow to compute and may introduce unwanted lag, as the detection of the concept drift has to be done in real-time. Finally, the Wald−Wolfowitz and Smirnov tests have a lower accuracy, which can be attributed to their inability to accurately detect the beginning and ending of the artificial event (the first and last window), but they are very stable, in terms of low variance in the results,

[3]This number is different because some "normal" windows might be detected by one of concept drift tests as well, moving the detection to cell $[4, 3]$. Other than that, the statistical test used for concept drift detection does not affect the RDA.

have barely any false positives and are very fast to compute on runtime.

The size of the window used proves to be a very important parameter and its selection depends on a number of requirements. Large windows have the advantage of higher accuracy, but they are very expensive in terms of computational time needed to process the data contained in large windows. At the current sliding window setup (see figure 4.7), the concept drift that happens any time between time t_1 to t_0, is detected at time t_0. This means that there is a delay up to $t_d = t_w + t_p$. The first term, t_w, is the time corresponding to a window with w samples and depends on the size of the window and the sampling frequency of the instrument or sensor used[4].The second term, t_p, is the time needed for processing the data contained in the two windows w_1 and w_2, which also depends on the window size and the method selected. The dimensionality of the data (in this case it is 270, the number of channels of the mass spectrometry data) also significantly affects the speed of processing; the higher the dimension, the costliest the computation of the test statistics. The selection of the size of the window depends on all these factors and is determined by the particular data and the application and the time constraints associated with it.

The addition of concept drift to the mass spectrometry data confirms the hypothesis, that the RDA produces a very high number of false positives, and it is no longer able to perform anomaly detection. For the gradual drift tested in the data with anomalies, the concept drift can be generally successfully detected. The results show that the cases where anomalies are detected as concept drift are very few. This indicates that statistical hypothesis testing not only can detect concept drift, but it can also differentiate between anomaly and concept drift. From the two methods compared, the $\mathrm{MMD_b}$ is able to detect the concept drift, using the universal threshold, set to correspond to an $\alpha = 0.05$ significance level. This is an important advantage, as it means that the method does not have to be tuned for different data or intensities. On the other hand, the exponential computation time of the test statistic, in contrast to the very rapid calculation of the Wald-Wolfowitz test statistic, acts in favour of the latter. The selection between the two is therefore a trade-off mainly between universal application of the method and computational cost, and depends on the application and the computational resources available.

Although linear transition is considered in the experiments presented in this chapter,

[4]if f is the sampling frequency (samples/second) and w the size of the window in terms of samples contained in it, then $t_w = w/f$

there is nothing particular to detecting linear transitions on the solution proposed, however it could be useful to examine other types of transitions.

The purpose of detecting concept drift is to trigger adaptation. The fact that the performance of the RDA degrades with concept drift, with a high number of false positive detections supports the argument that adaptation is necessary. The tests analysed in this chapter detect concept drift over one or more windows (depending on the duration of the transition and the window size). After they stop detecting concept drift it can be inferred that the system is now in a new stable environment. Adaptation can be triggered after the transition to a new stable environment is completed. The implication of attempting to adapt before the transition is completed, is that the the concept does not stay stable enough for a prolonged time. Minku et al. [101] define an intermediate concept as "a concept that is not active for enough time to be learned". Therefore any updated system runs the risk of very soon becoming again obsolete. Very gradual and prolonged transitions can be considered as a corner case in future work, but they are not in the scope of this thesis.

An adaptation mechanism that is triggered when a new stable environment has been reached and utilises knowledge form the old environment to adapt to the new, unknown environment is proposed and analysed in the following chapter.

Chapter 5

Adaptation Using Ensembles and Implicit Performance Metrics

This chapter addresses the adaptation in response to concept drift. Using the RDA [4] in a chemical detection task involves training the algorithm on data collected in a specific background environment. Training the RDA can be accomplished by hand tuning the parameters [5], or it can be treated as an optimisation problem; finding the parameter set that will lead in optimal RDA performance on the training data. As such, the selection of a parameter set can be tackled using an evolutionary algorithm, as proposed in [65]. The trained RDA should have reasonable performance during runtime (test data), assuming that the environment which generates the data remains stable, i.e. that the training data and the test data are drawn from the same distribution. If, however, that environment changes during runtime, the RDA will stop behaving as trained. A change in the background environment is referred to as concept drift. As demonstrated in Chapter 4, the RDA will produce a very high number of false positives because of a background concept drift event, artificially added to mass spectrometry timeseries data. For this reason, the system needs to be able to adapt to the changing environment; the number of false positives needs to be decreased, whilst ensuring that anomalies can be detected with reasonable precision (low false negative rate).

In Chapter 4 it is proposed that concept drift detection can be performed in parallel with anomaly detection, by monitoring the incoming data. Following that, a framework for detecting change using statistical hypothesis testing has been established. Statistical tests applied on consecutive windows, have been found effective in detecting whether there is concept drift in the incoming data and in differentiating between that and anomalies.

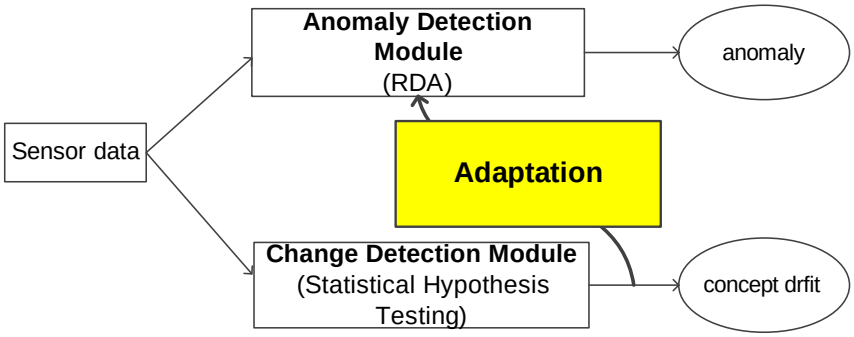


Figure 5.1: The proposed system employees separate modules for the detection of concept drift and the adaptation. Presented in Chapter 4, a statistical hypothesis testing method runs in parallel with the RDA, detecting concept drift in the incoming sensor data. The detected concept drift can be reported to the end user for analysis, but its main purpose is to trigger the adaptation module. The actions that can be taken in order to adapt the anomaly detection module are the focus of this chapter.

As illustrated in figure 5.1, the detection of concept drift triggers an adaptation module. This way, the adaptation process can be initiated only when there has been true change.

Revisiting the application scenario, the concept drift detection module detects the transition between two stable environments with different backgrounds. Concept drift is detected over one or more windows and when the detection of concept drift stops, it is assumed that the robot is in a new stable background environment, i.e. the transition period is over. At that moment, adaptation to the new environment/concept can be triggered. This is demonstrated in figure 5.2.

Environment A is considered to be the known environment, for which there are labelled training data to train the RDA on. For this work, training the RDA using an Evolutionary Algorithm (EA) will be adopted, as it does not require exhaustive hand tuning of the parameters. An EA maintains a population of solutions (RDA parameter sets) and iteratively searches for the optimal solution[1], through promoting the best from the population and applying genetic operators that encourage diversity among the solutions and effective exploration of the search space. At the end of the evolutionary process, a population, i.e. a set of solutions of varying optimality, is produced. From this final population, the best parameter set, i.e. the best RDA, can be selected for environment A.

[1]The optimal solution means the RDA, or to be more precise the RDA with a specific parameter set, that, when run on a training dataset associated with the specific environment, produces the lowest number of false detections, missed anomalies, or in general optimises one or more of the performance metrics outlined in section 2.4.1

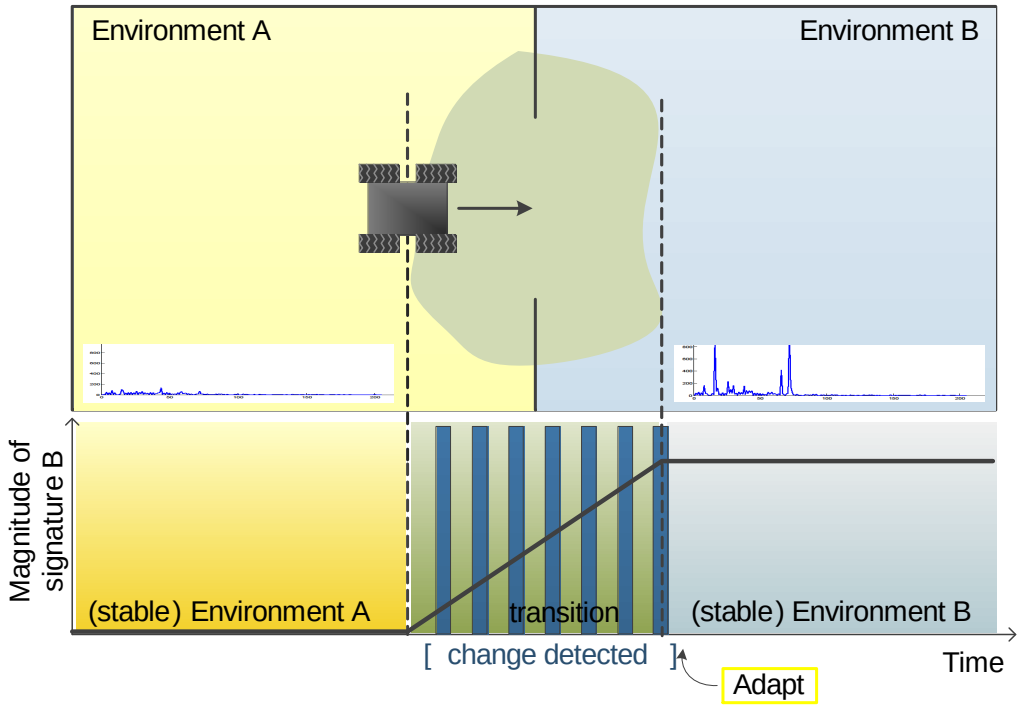


Figure 5.2: Application scenario: a robot moving between environments with different backgrounds. Environment A and environment B have different backgrounds, shown here through a characteristic background signature. If the environment is sensed through a mass spectrometer, the background signature can be thought of as a measurement snapshot, corresponding to the robot being in this environment when anomalies are absent; it is a spectrum corresponding to the set of chemicals that are considered normal in this specific environment. The transition from environment A to environment B, is detected through statistical hypothesis testing, over a series of consecutive windows. The bars at the bottom indicate a detection of concept drift over the corresponding window. The cease in detection of concept drift indicates that a new stable environment has been reached, and triggers the adaptation process.

When the robot moves to environment B, the selected RDA stops being optimal, because concept drift has occurred. In fact, the whole previously evolved population is now at best suboptimal. Algorithms that adapt in response to concept drift, typically use a batch of the data from the new concept (environment B) to retrain or update the current model (see relevant review in sections 3.2.2 and 3.3.2). In order to evolve a new optimal solution, a new training set, i.e. a set of labelled normal and anomalous data, would be needed. However, in an autonomous, real-time system, the existence or timely acquisition of such data cannot be assumed [116]. For this reason, it is proposed here that previous acquired knowledge can be exploited by combining solutions from the evolved, now suboptimal, population into an ensemble. Ensembles that are created directly or indirectly through evolution, are called evolutionary ensembles.

An ensemble, as discussed in section 3.3.1, is a set of classifiers (in this case anomaly detectors) whose decisions are combined into a single ensemble decision. It has been found that, given reasonably accurate and diverse base learners, the ensemble error rate decreases compared to that of the base learners [10]. The first of these two prerequisites, diversity, is ensured because the multi-objective evolutionary process used in [65] to optimise the RDA parameters, in principle promotes the existence of diverse solutions in the population. The second, accuracy, cannot be computed directly without new labelled data. Instead, a novel method of estimating the accuracy is proposed: extracting *implicit* performance information when the RDA operates in the new environment.

In this work, implicit performance metrics are defined as metrics that can be calculated from the model used for detection, without requiring explicit knowledge of the model's performance. If the model makes a detection and then the true label is provided by some external feedback mechanism, the detection error can be directly calculated. The performance of the system in this case is explicitly calculated. In contrast to that, when there is no feedback there can be properties of the model that indicate whether it performs as it should. In section 3.2.2, the detection of drift by monitoring the properties of the model was discussed. The properties of the model, for example the coverage of the input data by a self-organising network, is an implicit performance metric, as it does not state with certainty that the network is outdated, but it is an indication strong enough to trigger some adaptation action.

The implicit performance concept is here extended to the RDA. Estimating the performance of the members of the population on the new environment B, allows for the selection of the estimated more accurate base learners for the ensemble.

From the proceeding analysis, the research question addressed in this chapter is twofold: (1) Can evolutionary ensembles reuse the existing evolved population in order to achieve low false positive and false negative rates in the new environment? (2) Can implicit information performance be extracted from the RDA and used to improve these rates, by selecting the members of the ensemble based on their estimated accuracy?

5.1 Structure of this chapter

Section 5.2 presents relevant work on the combination of evolutionary computation and ensemble learning. In section 5.3 the methodology for using an evolutionary ensemble to adapt the RDA is presented, along with the implicit performance selection mechanism.

The experimental design is presented in section 5.4 and the results in section 5.5. Finally, a discussion and summary is provided in section 5.6.

5.2 Evolutionary ensembles

Kuncheva in [83] notes that the ensemble can be "perceived as a living population expanding, shrinking, replacing and retraining classifiers, taking on new features, forgetting outdated knowledge" and consequently parallels can be drawn with evolutionary computation. In Chapter 3 it has been discussed that a key issue in ensemble learning, both static and adaptive, is a balance between diversity and accuracy. An accurate and diverse population that promotes both exploration and exploitation, is the key to EAs as well [125]. In the last years, there has been increasing interest in the field of evolutionary ensembles, which use the population of an EA as a pool of diverse classifiers for an ensemble. In this work it is argued that the balance between diversity and accuracy that can be achieved through evolution, can be of interest in adaptation. To the best knowledge of the author this paradigm has not been used yet to address changing environments. In this section, relevant work on how evolutionary computation can be exploited in ensembles in the fixed environment case will be presented.

5.2.1 Evolutionary Algorithms and relevant concepts

Evolutionary algorithms (EAs) are a population-based class of optimisation algorithms [126]. Drawing inspiration from natural evolution, an EA maintains a population of individuals that represent solutions to an optimisation problem. The efficiency of every individual, i.e. how well it solves the problem, is determined through the fitness function. The basic Genetic Algorithm (GA) runs iteratively for a number of generations and evolves the population by applying the genetic operators of selection, crossover and mutation: at every generation all the individuals are evaluated through the fitness function and the fittest are selected as parents to the next population. Crossover recombines two parents and mutation randomly modifies the offspring. By inserting randomness, the diversity of the population and the exploration of new regions of the search space is promoted. This process is repeated, with the offspring population replacing the whole or part of the parent population. The fitness of the population is eventually improved because of the selection pressure; the fittest of a generation are selected to be used for the creation of the new

population.

In many cases there are more than one objectives to be optimised by evolution. Multi-objective optimization (MO) seeks to optimize a fitness function with more than one components. Usually the objectives are competing; optimising the fitness across one objective can lead to degradation across the other objective(s). As a result, a Multi-Objective Genetic Algorithm (MOGA) looks for a family of points, known as the Pareto-optimal set [127]. Every point on this set is optimal, in the sense that there can be no improvement across one objective without the degradation across at least one of the other objectives.

In the case of single objective EAs, diversity in a population is typically achieved through the operator of mutation. In the case of multi-objective EAs, diversity is also ensured because multiple solutions spread across the pareto front are produced. However, in cases where increased diversity is needed, both in the single and multi-objective case, niching can be used. Niching methods promote the formation of stable, spatially separated sub-populations, which cover spatially separated neighbourhoods of different optimal solutions [128]. The two most commonly used methods of niching are fitness sharing and crowding. Fitness sharing promotes the formation of isolated sub-populations by reducing the pay-off (through the fitness function) of individuals that are similar. Crowding methods, on the other hand, insert new elements in the population, replacing similar individuals.

5.2.2 Evolutionary ensembles

There are a few ways that EAs can be used to assist in the creation of diverse and accurate base classifiers for ensembles, in order to achieve improved performance. Commonly, given a training set, EAs are used to generate a diverse pool of base classifiers that perform accurately on the training set. To generate a classifier, or a learning model, evolution acts as a training method: a typical example is evolving the weights and topology of an ANN (also called neuroevolution [127]). In general, for any learning model, an EA iterates through consecutive generations of model parameters, to find the set of parameters that will optimise the performance of the learner.

When a **single objective** is used, that objective is maximising the accuracy (or minimising the error) of the classifier. A widely used approach of this category is evolving neural networks using Negative Correlation Learning (NCL) [129]. Neural networks, as base learners of an ensemble, are evolved and the objective is to minimise the error func-

tion of every network. The diversity of the ensemble is promoted through fitness sharing: the error function is augmented by a correlation term, which correlates negatively the error of this classifier to the error of the ensemble (created from the whole population). Thus, individuals self-organise into species that cover different patterns of the training set. The ensemble, created either by the whole or a subset of the population, outperforms several classification algorithms on two benchmark classification datasets. Similarly, a set of classifiers is evolved in [130] as a pool of candidates for an ensemble. In this study expression trees are shown to be the best candidate base algorithm, outperforming ANNs or mixed models. The contribution of diversity in the population is examined by comparing two crowding strategies, deterministic crowding and probabilistic crowding, against each other and against a standard EA with no crowding. It is found that an ensemble formed by the best 20 members of a population evolved with deterministic crowding has a significantly higher accuracy than an ensemble with members evolved without crowding, i.e. not as diverse. In [131] a diverse pool of candidates is produced by using a co-evolution fitness function; the fitness function is associated with the hardness of the training examples which an individual correctly classifies. The ensemble is iteratively created by adding members selected from the population, based on a margin training set error criterion. This method outperforms boosting [87], producing a lower test error rate in four out of six problems tested. EAs do not have to be restrictively applied to evolving the models. In [101], following the training data manipulation paradigm of bagging [87], an EA is used to evolve optimal partitionings of the training set, that will be used to train different base classifiers. The ensemble is formed using not only the final population but all the intermediate populations as well. Interestingly, including the less accurate and more randomised classifiers of the early generations can also contribute to the ensemble accuracy; the individual accuracies of these classifiers may not be very high, but their existence does increase the diversity, in this way improving the overall performance.

Multi-objective GAs, also find application in evolutionary ensembles, as they produce a population of diverse solutions scattered across the Pareto front. In [132], the MPANN [133], an MOGA for evolving ANNs, is used to evolve the members of an ensemble. The two objectives used, are the ensemble accuracy on two disjoint parts of the training set. The final ensemble consists of the non-dominated (Pareto) front of the final population. Evaluated on two classification datasets, this approach outperforms a simple BP trained network and gives similar results to the NCL approach [129], with smaller

ensemble size. Finally, diversity can be explicitly stated as one of the objectives of the MOGA, as proposed in [134]. Similarly to [132] an evolution process that promotes and selects the ensemble members based on their Pareto dominance is used. One of the main differences, is that the two fitness objectives are accuracy and diversity, i.e. the base classifiers are evolved to be explicitly diverse from each other. The diversity objective is either to minimise the correlation between the output of one individual and that of the ensemble, or to maximize the pairwise differences between the individuals. Both variants of the algorithms achieve competitive and mostly better generalisation performance when tested with multiple established learning algorithms as base learners. Additionally, the variant that maximises the pairwise differences is competitive to the MPANN [132]; it achieves lower means of test error, but has marginally higher variance in the results.

Evolutionary ensembles, to the best knowledge of the author, have not been used to deal with concept drift. One exception is [135], where an EA is used as a training method for a new ANN based classifier regularly added to the ensemble. The inherent diversity, however, of an evolved population of base learners is not exploited. In this chapter, the idea of "ensembles for free" [131] is used. When a GA is used to optimise a learning model, multiple individuals from the final population can be used in an ensemble, instead of the single best member, without the added cost of explicitly designing an ensemble.

5.3 Materials and methods

In this section the methodology and main elements of the ensemble adaptation approach will be presented. The notation that will be followed throughout this analysis is presented below:

- Environment A is the old, pre-drift environment, or in concept drift terms the old concept. It is assumed that training data is available for this environment, so the RDA can be trained/optimised for use in this specific environment. The training data consists of the mass spectrometry readings, plus a file where the anomalies present in the data are labelled.
- Environment B is the after-drift environment. It is assumed that, since it is a new unknown environment, no labels are available at the time of adaptation. This data is the testing data.
- Both environments, A and B, are associated with a specific dataset. The ICARIS

dataset [117] is divided into seven files. Each file can act as the dataset associated with an environment. The notation environment A is file i, means that the RDA has been optimised using the labelled data in file i, and similarly environment B being file j means that the RDA ensemble has been tested on file j.

- Training or optimising the RDA is equivalent to the optimisation of its parameter set. The notation "an RDA" is the RDA that uses a specific parameter set. A set of different RDAs does not mean that the algorithm itself is different, but that different parameter sets are used, which in turn determine the performance of the RDA.

- The performance of the RDA can be measured in any of the anomaly detection evaluation metrics that have been presented in section 2.4.1. In this chapter, the performance of the RDA or the ensemble will refer to the number, or rate of false positives and false negatives. A false positive occurs if a detection is made when no true anomaly exists and a false negative occurs when an anomaly is not detected. The exact metric used will be clarified as appropriate.

5.3.1 Training for environment A

As discussed earlier, a convenient and effective way to optimise the RDA is using an evolutionary algorithm. The NSGAII (Non-dominated Sorting Genetic Algorithm) [136], which has been used to this end in [65], is a MOGA which evolves a population trying to simultaneously minimise multiple goals. The algorithm produces a set of solutions that spread across the Pareto front, while it employs a crowding mechanism to promote diversity in the evolved population. When faced with more than one objective, the Pareto front, or the non-dominated front, is the set of solutions for which there is no other individual in the population that performs better in all the objectives. In the case of optimising the RDA, parameter sets for the RDA[2] are evolved and the two goals used are the number of false positives and the number of false negatives. The NSGAII has been found to evolve consistently good parameter sets [65]. Additionally, it introduces diversity, both because of evolving solutions across a spread out Pareto front, but also because it uses a crowding function to remove similar solutions. This diversity makes it suitable for use in

[2]Referring back to the two equations of the RDA, 2.8 and 2.9, the set of parameters that is evolved is $\{\beta, l, \alpha, g, d, b\}$. Please refer to section 2.5 for a detailed description of these parameters.

an evolutionary ensemble. For these reasons, it is the EA that will be used in this work as well.

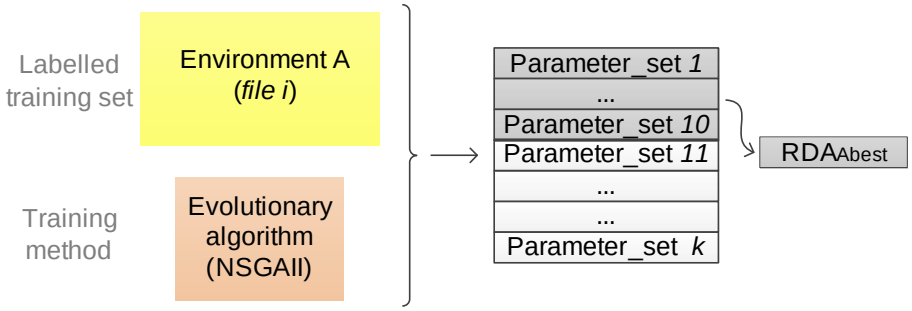


Figure 5.3: Given a labelled training set on which the RDA needs to be optimised an evolutionary algorithm will output a population of k RDAs. This set of solutions, often consists of an elit of the best individuals, or a non-dominated front, and the rest of the population. The best individual of the final population, RDA_{Abest} will be selected as the solution to the optimisation problem at hand.

As shown on figure 5.3, the NSGAII produces a set of solutions/RDAs, i.e. a population of parameter sets that minimise the number of false positives and false negatives of the RDA, measured on the specific training dataset associated with environment A. From the final population, possibly consisting of an elit of best RDAs and the rest of the population, the best RDA can be selected. This trained, best for environment A, RDA will be referred to as RDA_{Abest}.

5.3.2 Forming an ensemble of RDAs

It is proposed in this work that when concept drift happens, an ensemble of RDAs could improve the performance over a single RDA, especially considering that RDA_{Abest} will be suboptimal in the after-drift environment B. Here the method for combining the evolved population of RDAs in an ensemble is presented.

It is reminded to the reader that at every timestep the RDA produces a classification signal $c(t)$, depending on whether one or more receptors have exceeded the detection threshold l; $c(t) = 1$ signifies that the specific timestep belongs to an anomaly, while $c(t)$ remains at zero when an anomaly is not present (see also equations 2.8 and 2.9). Typically, the output of the RDA over a portion of the data has the form depicted in figure 5.4. The final output groups as one anomaly all the consecutive timepoints that have been classified as anomalous. Each of these anomalies is then associated with a signature, calculated based on the activity of all the receptors over the duration of the anomaly. The signatures will not be used for this work, as their use for identification of

the anomalies are considered a post-processing step. Instead, the focus will be the correct detection of anomalies, i.e. minimisation of false negatives and false positives.

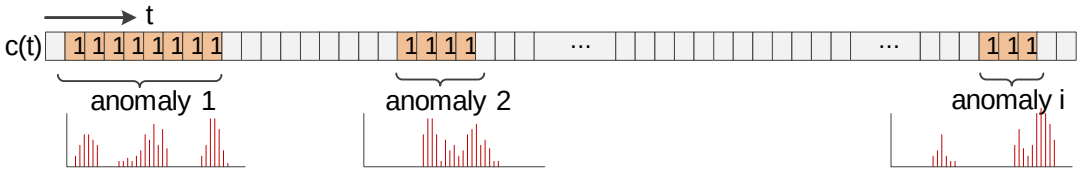


Figure 5.4: Typical output of a single RDA. At every timestep (with every new reading), the RDA outputs a classification signal. Each anomaly takes place over a number of consecutive timepoints (duration), for which a classification signal of 1 is produced. The first - anomaly detection - level of the RDA is the binary, classification signal output, indicating the presence or not of anomalies. For the second level - identification -, a signature is extracted for every anomaly and this signature can be used to recognise the anomaly. Here only the first level will be used.

Multiple versions of the RDA can be combined in an ensemble, as illustrated in figure 5.5. Every RDA has a different parameter set, which leads to a different set of detections. The simple majority vote is selected as the combination method: as described in equation 5.1, at every timepoint the decision of an ensemble, i.e. the classification signal $c_E(t)$ is determined through a majority vote between the classification signals of n individual detectors, $c_i(t)$; a timepoint is anomalous if more than half of the detectors classify it as such.

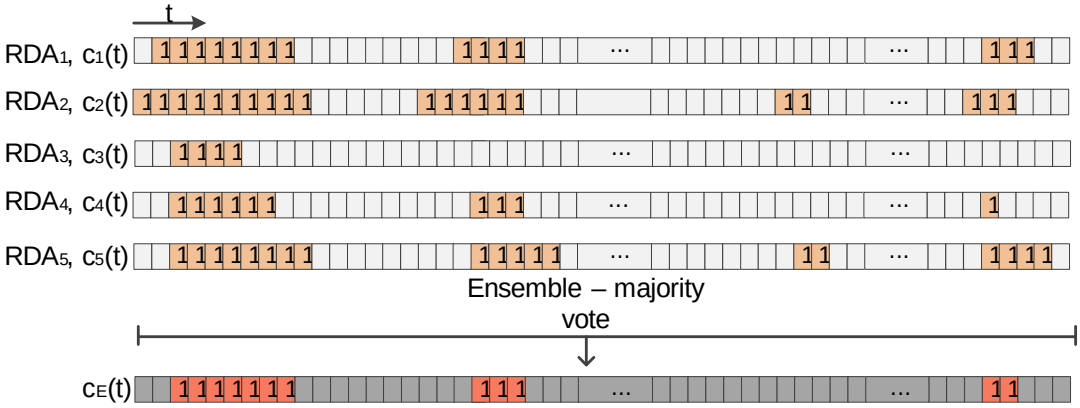


Figure 5.5: Output of an ensemble of 5 RDAs, combined by majority vote. Each RDA makes an independent decision. The ensemble follow the majority, i.e. the decision of over half of the member-RDAs.

$$c_E(t) = \begin{cases} 1, & \text{if } \dfrac{\sum_1^n c_i(t)}{n} \geq 0.5. \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

The notion of using an already evolved population as the candidate pool for an en-

semble (instead of explicitly evolving a set of ensemble candidates) is in agreement with the "ensembles for free" notion adopted in [131]. The difference is that in this case it is extended to concept drift, which changes the problem from generalisation to adaptation. Ensembles, as discussed in Chapter 3, need relatively diverse and accurate base learners. Diversity is introduced by the NSGAII. Relative accuracy of the base RDAs, however, cannot be directly calculated in the new environment, but as will be discussed in the following sections it can be implicitly estimated.

5.3.3 Selecting the members of the ensemble

It is common in ensembles to use some form of performance information to select the base learners. Especially in concept drift cases, the base learners that are used to form the ensemble are the ones that are most relevant/accurate on the current concept. This has been discussed in section 3.3.2. Selecting members in this way is possible with supervised learning problems, where there is a labelled dataset, or with a forecasting problem, where the true label/value is measured or observed after the prediction. In real-time detection, feedback in the form of labels for recent data cannot be assumed.

However, the behaviour of detection algorithms can be described by their dynamics. In the case of the RDA, it can be argued that the behaviour could possibly be judged by a set of implicit performance metrics. The RDA updates in every timestep the position p and the negative feedback n values for each receptor. By recording the values of these two over the duration of a dataset, as well as using information about the detections of the RDA (e.g. duration of an anomaly, D) in the same duration, some implicit metrics of performance can be calculated. Such can be the mean position and negative feedback, the equivalent standard deviations and the mean duration of the detected anomalies (details on the calculation of the implicit metrics are given in the next subsection). These metrics will reflect to some extent the behaviour of a specific RDA, i.e. a specific parameter set, as the calculation of p and n at each timestep, includes the parameter set associated with this RDA (see equations 5.2 and 5.3).

For any given dataset, or a finite partition of a dataset, and an RDA these metrics can be calculated, and combined in an Implicit Metrics vector, IM. The implicit vector $IM_{i,j}$ is associated with a specific RDA, indexed RDA_j and a specific environment, environment i, on which it is calculated. For example the notation $IM_{A,5}$ means that the IM vector has been calculated when RDA_5 runs (performs anomaly detection) on environment A.

The notation $IM_{B,5}$ means that the IM vector has been calculated when the same RDA_5 runs on environment B.

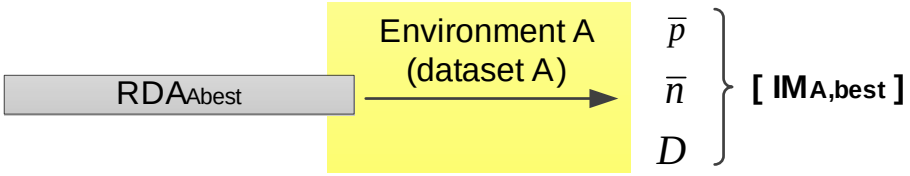


Figure 5.6: Calculation of the reference implicit vector. The implicit vector is associated with a specific RDA and a specific environment on which it is calculated. When an RDA runs through a specific dataset/environment, a log of the position and negative feedback values along with details about the output detections can be used to calculate an implicit performance vector for the RDA. The reference implicit vector, $IM_{A,best}$ is the one that corresponds to the best trained RDA, RDA_{Abest} on the training data (environment A under the current notation).

If this is applied to the evolved RDA_{Abest} in the training environment/dataset A, a reference implicit metrics vector can be calculated, $IM_{A,best}$ (see figure 5.6). This reference implicit vector can be considered to reflect the behaviour of a "good" RDA. When the detection system then has to deal with an unknown environment B - a new unlabelled dataset - then the implicit vector of a possible RDA can be calculated on data from environment B. This implicit vector can then be used as an indication of how good this parameter set is on environment B. The assumption is that the closest the implicit vector of a candidate is to the reference implicit vector, the more likely it is that this candidate will be a good one.

Given a population of solutions evolved for environment A, a set of implicit vectors, one for each individual, can be calculated on a new unknown environment B, as shown in figure 5.7. It cannot be guaranteed that the RDA, among a set of candidates, whose implicit vector is closest to the reference is going to be the best for environment B, but it is argued that this proximity can be used in order to select, as members of the ensemble, the candidates with the highest *estimated* accuracy.

Calculating the implicit vector

The RDA has been presented earlier in Chapter 2 and can be summarised by two equations, which are reminded here:

$$p_{i,t} = bp_{i,t-1} + K(S_t) - an_{i,t-1} \tag{5.2}$$

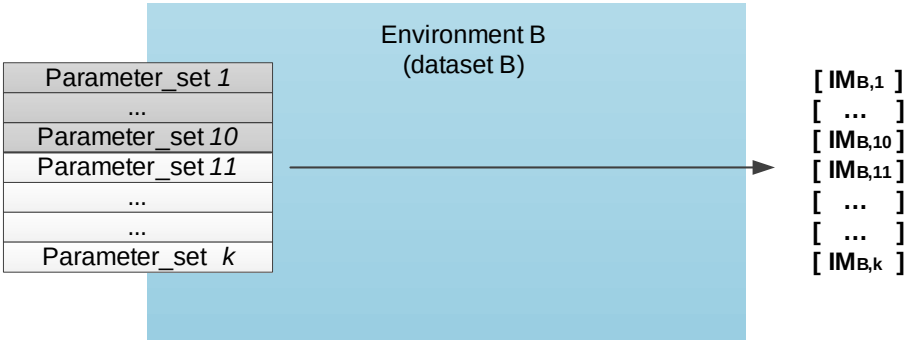


Figure 5.7: Calculation of the implicit vectors of a set of solutions. A population of k solutions will result in a set of k implicit vectors.

$$n_{i,t} = dn_{i,t-1} + gH(p_{i,t-1} - \beta) \tag{5.3}$$

In equations 5.2 and 5.3, $p_{i,t}$ and $n_{i,t}$ are the position and negative feedback for receptor i at timestep t. This can be formulated as two vectors $\overline{p_t}$ and $\overline{n_t}$ being output at each timestep, with $\overline{p_t} = \{p_{1,t}, p_{2,t}, ..., p_{w,t}\}$ and $\overline{n_t} = \{n_{1,t}, n_{2,t}, ..., n_{w,t}\}$, where w is the number of receptors. After T timesteps have passed, the RDA will have produced a set of z anomalies, $A = \{a_1, a_2, ..., a_z\}$, each associated with a set of receptors involved in the specific anomaly, a start time and an end time. From the last two, the set of anomalies can be associated with a set of durations $D = \{d_1, d_2, ..., d_z\}$.

Using $\overline{p_t}$, $\overline{n_t}$ and D, a set of features can be extracted that will form the implicit vector for a portion of the dataset with size (in time) T, $IM|_T$. After some preliminary tests and observations, the following scalar features are used: the mean across the whole duration of the data portion, T, $p_{avg}|_T$ of the mean position across all the receptors for every timestep. The mean across the whole duration of the data portion, T, $p_{std}|_T$ of the standard deviation of position across all the receptors for every timestep. The equivalent metrics are calculated for the negative feedback. Additionally, two ratios of these quantities are used, $R_1|_T$ and $R_2|_T$, and the mean duration of all the detected anomalies in this data portion, $d_{avg}|_T$. The use of the specific metrics was decided after a preliminary study on the impact of different RDAs on $\overline{p}$, $\overline{n}$ and D. The metrics described are outlined in the following equations.

$$p_{avg}|_T = \frac{\sum_{t=1}^{T} mean(\overline{p_t})}{T}$$
$$p_{std}|_T = \frac{\sum_{t=1}^{T} std(\overline{p_t})}{T}$$
$$n_{avg}|_T = \frac{\sum_{t=1}^{T} mean(\overline{n_t})}{T}$$
$$n_{std}|_T = \frac{\sum_{t=1}^{T} std(\overline{n_t})}{T}$$
$$R_1|_T = \frac{p_{std}\ |_T}{p_{avg}\ |_T}$$
$$R_2|_T = \frac{n_{avg}\ |_T}{p_{avg}\ |_T}$$
$$d_{avg}|_T = \frac{\sum_{i=1}^{k} d_i}{z}$$

The resulting implicit vector will be:

$$IM|_T = [p_{avg}|_T \quad p_{std}|_T \quad n_{avg}|_T \quad n_{std}|_T \quad R_1|_T \quad R_2|_T \quad d_{avg}|_T]. \tag{5.4}$$

Implicit ensemble creation algorithm

From the previous proposed methods, an *implicit ensemble*, denoted as *i.ens* can be created, using the top RDAs based on their implicit performance. The top RDAs are the ones whose implicit vector has the highest similarity to $IM_{A,best}$. This is outlined in the algorithm in figure 5.8. The algorithm uses the reference Implicit vector, $IM_{A,best}$ and the set of IM vectors for all the population, calculated on environment B. The algorithm creates an ensemble of the *ens_size* estimated best individuals in the population.

```
Create ensemble(Pop, {IMB}pop, IMA,best)

for i=1:pop_size
Pdi= dist(IMB,i,IMA,best)
end
rank Pop based on Pd
pop_ind = Select ens_size top ranking members
i.ens=Pop(pop_ind)
```

Figure 5.8: Pseudocode for creating an ensemble based on implicit performance. The similarity of $IM_{A,best}$ and $IM_{B,i}$ is calculated as their Euclidean distance ("dist" function).

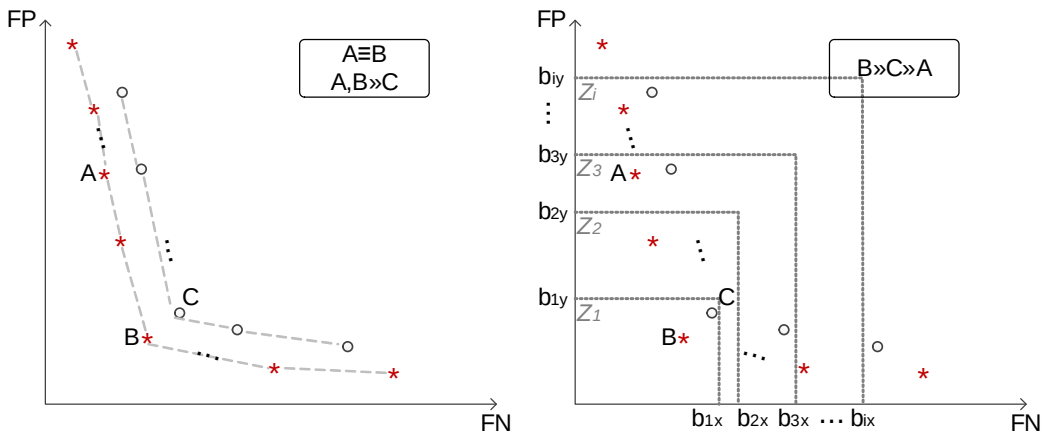
5.3.4 Selecting the best from a population

The NSGAII is a multi-objective genetic algorithm which evolves a population trying to simultaneously minimise two goals, in this case the number of false positives and the number of false negatives (misses). The algorithm produces a set of solutions that spread across the Pareto front. In the example given in figure 5.9, the Pareto front can be seen in red asterisks. In this example, individual C does not belong in the Pareto front because it is dominated, i.e. B is better than C in both objectives. The final output of the algorithm is a set of solutions, the Pareto front, which are in theory equally good. The user can select the one that is best for the application at hand. If the individuals need to be ranked, Pareto optimality is not necessarily the best way to do it. For example, in figure 5.9a it is clear that B is better than C, but whether A is better than C, or whether A is equivalent to B is not.

For the cases where an automated choice of the best individual is needed, or a fairer ranking of the population, a *bounded ranking*, or *bounded optimality* method is proposed. As depicted in figure 5.9b, a set of bounded areas Z_i can be defined, enclosed in boundaries $\{b_{ix}, b_{iy}\}$. All the solutions belonging to a lower area will be ranked higher. Again in the example of figure 5.9b, solutions B and C that belong to Z_1 will rank higher (they are better) than solution A, which belongs to Z_3. Within a bounded area, Pareto optimality stands, which is why B is still better than C. The number of areas as well as the boundaries $\{b_{ix}, b_{iy}\}$ can be set to divide the results space in equal areas, or can be set by the user according to whether an objective is prioritised against the other. An alternative way, which will be used in the experiments to follow, is to set the bounds according to the spread of the values across the two axes. All the x values are ranked in $X = [x_1 x_2 ... x_n]$ and the number of areas l is selected. Then b_{ix} are selected so that the vector X is split in l equal parts, i.e. $b_{ix} = x_{i\frac{n}{l}}$. The limits b_{iy} are set in the same way. It should be noted that this method is not proposed as a substitute of the Pareto ranking used during evolution by the NSGAII; bounded ranking does not provide the necessary diversity for a GA to explore the search space efficiently.

Bounded ranking is used in the following sections, in order to locate the best individual for the training set, environment A, RDA_{Abest} from a population of evolved possible solutions. Additionally, both bounded and Pareto ranking can be used to select ensemble members based on their performance on the training set. A *Pareto ensemble*, *p.ens*, is an ensemble of all the individuals in the non-dominated front. A *bounded ensemble*, *b.ensx*

consists of the top x best individuals based on bounded optimality. These two ensemble can be used for comparison to *i.ens*.



(a) Ranking based on Pareto optimality

(b) Ranking based on bounded optimality

Figure 5.9: Ranking of a set solutions, based on Pareto optimality (a) or bounded optimality (b). The ranking in each case is shown in the top right. Note that the ideal solution minimises both goals, i.e. the "best" in this graph is at point (0,0). The notation $B \gg A$ means that B is better, while $A \equiv B$ means that they are equivalent. Bounded ranking on the right is proposed as a way to make a decision about the ranking of the different points.

5.3.5 Training and test sets

The ICARIS dataset [117] (described in section 4.3.5) is divided into seven files. Every data file corresponds to a different data collection session and a different set of anomalies. Each file can act as a training or a test dataset. In order to simplify the analysis, it is assumed that the data can be split into clearly defined training and test sets. A training set is equivalent to a known environment, i.e. the environment before drift. This is called training set because it is the one used to optimise the RDA. The training set is equivalent to environment A and it can be one of the files of the ICARIS dataset.

The test set is equivalent to the new, unknown after-drift environment, i.e. environment B. Again, any file from the dataset can be used as the test dataset. However, for the test data there has been concept drift. Concept drift is again simulated as in Chapter 4, with the addition of a background signature. In section 4.3.5, the concept drift was simulated with the addition of an artificial event. The artificial event was the linear addition, over a transition period of a background signature until it reached a given maximum intensity. The intensity of a signature, in accordance with the previous chapter, refers to

the intensity of the highest peak of the added signature. The transition period is not of interest here, as it is assumed that the previous step of concept drift detection will have processed it. Here, the test data is the new "stable" environment, i.e. with the added background signature stable at its highest maximum intensity. As a consequence, when a data file is used as environment B, the background signature, with maximum intensity, or simply intensity, I_{max}, is added straight from the beginning of the file - there is no transition period.

An example of a pre-drift and after-drift environment/dataset is given in figure 5.10. A portion of the data, which also contains an anomaly, is visualised with or without concept drift, i.e. with or without the added background signature shown in the figure.

5.3.6 Offline versus online creation of the ensemble

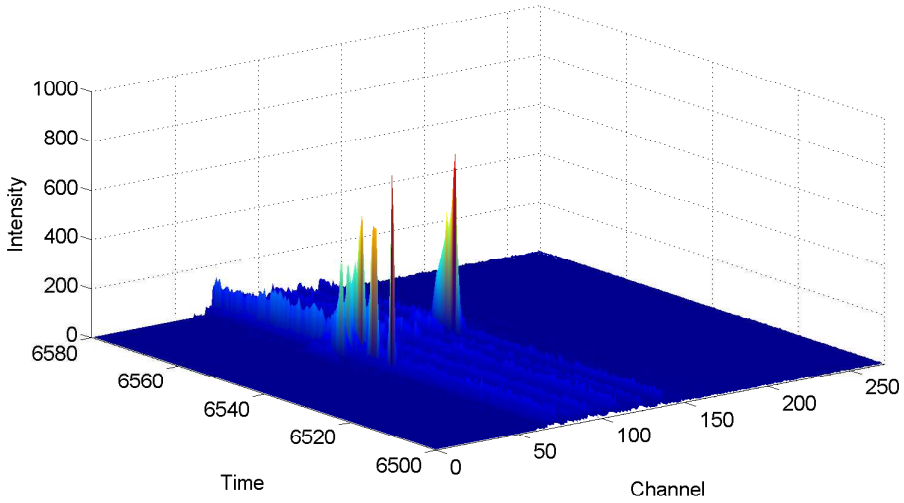
In the previous sections it has been assumed that the entire test set (all the data from environment B) is available for the calculation of the implicit metrics vectors (see figure 5.7). This corresponds to the case of offline anomaly detection, where data from environment B would be collected and then processed at a later time. However, for online, real-time anomaly detection the ensemble has to be created as the data is being collected.

Figure 5.11 shows the methodology for constructing the ensemble online. Once the system is in a new environment, the RDA has to run in this environment for a minimum time $[0 \quad t_1]$ until the first implicit vectors can be calculated. Then, as more data is collected, the implicit vectors can be recalculated and the ensemble updated. For every time point, t_i, at which an implicit vector is calculated, all the data from the beginning of the environment is used, i.e. the implicit metrics vector $IM|t_i$ is calculated over $[0 \quad t_i]$. For every set of implicit metrics vectors that is calculated at a given time for the whole population, $\{IM_B|t_i\}_{pop}$, a corresponding ensemble, $i.ens|t_i$, is constructed.

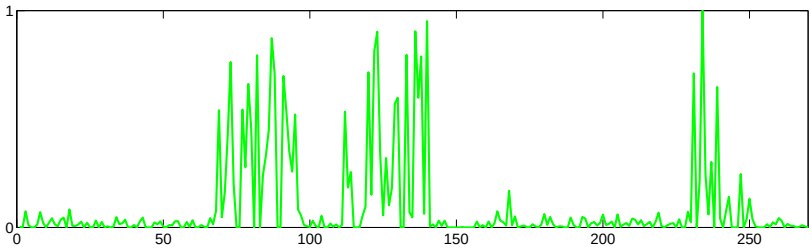
5.4 Experiments

Four sets of experiments have been carried out, in order to address the research question of this chapter: Can evolutionary ensembles reuse the existing evolved population in order to achieve low false positive and false negative rates in the new environment and can the proposed implicit ensemble improve these rates?

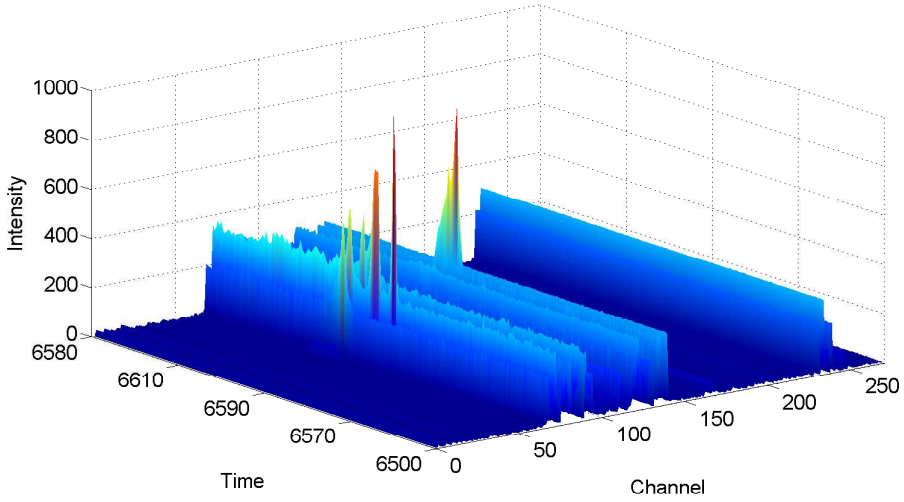
The first three experiments deal with the offline case of having to detect anomalies in



(a) Data before artificial concept drift



(b) The scaled signature that is added



(c) Data after artificial concept drift

Figure 5.10: The same partition of the data before and after the added artificial event. A signature of intensity 300 is added over the entire duration of the dataset. Note that the anomaly happening at timesteps 6525-6550 is present in both cases.

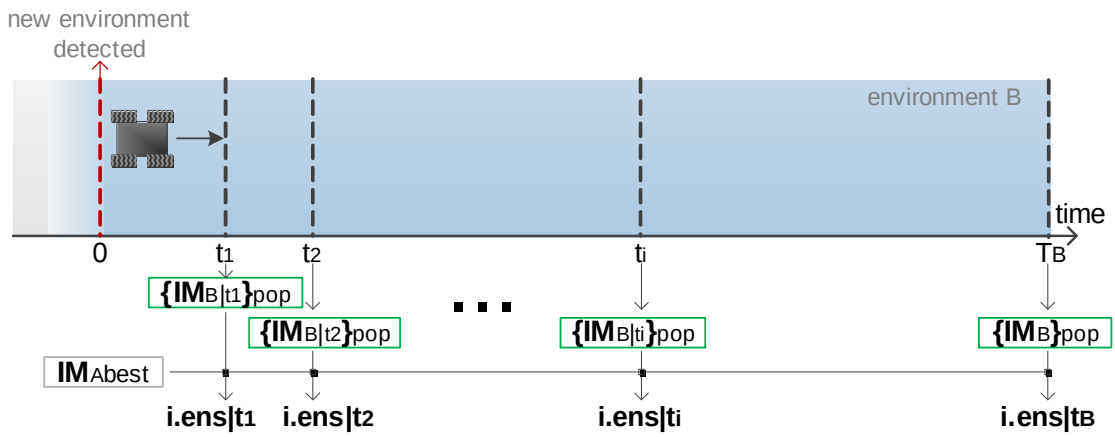


Figure 5.11: Constructing the ensemble online. The ensemble is created by constructing the implicit vector for the population over environment B. In the case where the ensemble is created online, the IM vector also has to be calculated online. By updating the IM vector at every timestep, a new ensemble, $i.ens|t_i$, can be regularly created based on the IM that incorporates the most recent data. If the duration of the whole file is t_B, then the ensemble $i.ens|t_B$ will be the same as the one created offline.

an unknown environment B, after all the data has been gathered. The fourth deals with the case of having to adapt in environment B, online, i.e. while the data is being collected. The experiments aim to determine:

Q.off1. The effectiveness of an ensemble using the entire evolved population, in improving the performance of the system in the new environment, over a single RDA.

Q.off2. Whether using implicit metrics to select the members of the ensemble improves the performance of the system.

Q.off3. Whether implicit metrics, as a selection mechanism for the members of the ensemble, offers an advantage over other methods, namely pruning the ensemble using the Pareto front, or the bounded optimality method.

Q.on. If an ensemble created online, using partial information about environment B, converges to the offline performance as more data is used to calculate the implicit metrics and how soon after having moved to environment B the online ensemble can be used.

5.4.1 Use of ICARIS files

In all the experiments, the training set, i.e. environment A, is file 1 of the ICARIS dataset. Environment A is the pre-drift environment, so file 1 is used without any modification.

Table 5.1: The files from the ICARIS dataset used for training/testing.

training	Environment A	file 1
testing	Environment B	file 1 + background signature
		file 4 + background signature
		file 6 + background signature

For the test set, i.e. environment B, the experiments are repeated for three different files: file 1, file 4 and file 6. These three have been selected because their size is comparable. As environment B is the after-drift environment, a signature is added to the corresponding data files. This setup is summarised in table 5.1. For environment B, first file 1 is used, which is the same as the training file but with the addition of concept drift, i.e. a background signature. This is because it was considered as a way to examine the ability of adaptation to concept drift alone (the detection of the same set of anomalies under a different background). Then the experiments are repeated on the other data files, again with added signatures, to ensure generalisation.

5.4.2 Experiment settings

In the previous chapter the main focus, when it came to artificial concept drift, was placed on the transition period, i.e. how fast the new background signature reached its maximum intensity. In this chapter however the interest shifts into the intensity itself and how it affects the performance of the RDA or the ensemble. In line with the previous chapter and the relevant analysis of section 4.4.2, artificial events of concept drift that lay in the 300-1000 region of intensities will be examined. It has been determined that at this range of intensities statistical hypothesis testing can detect that concept drift has happened. Moreover, the RDA produces many false positives, especially with increasing intensities, caused by the existence of the new background signature. Hence, it is meaningful to attempt to reduce the high false positive rate at these intensities through adaptation.

In all the experiments, 18 different signatures are used to create the after-drift environment B. This is done to eliminate effects of an added background signature affecting only a specific band of the spectrum. Moreover, since the population, i.e. the candidates for the ensemble, is generated through evolution, repeated runs are necessary to ensure that the results are not attributed to the stochastic element of the EA. All the experiments are repeated 85 times, which is determined to be a sufficient number of runs to observe

a medium effect at a 95% significance level in [137]. For every file and every intensity, the reported averages are calculated based on all the repetitions, which are 85 runs x 18 signatures = 1530 repetitions.

For the training, NSGAII [136] is used and the two objectives to be minimised are the number of false positives and the number of false negatives. The parameters for the NSGAII are in accordance to the ones used in [65]. The population size used is 40 and in every generation the 10 fittest parents are promoted in the new generation and each of them produces 3 mutated offspring.

5.4.3 Evaluation

In all the experiments the performance of either an RDA or an ensemble of RDAs is assessed in terms of the false positive rate (FPrate) and the false negative rate (FNrate) on the test set. It is reminded here that the FPrate is the number of false positives over the number of all the detections and the FNrate is the number of false negatives (missed anomalies) over the total number of labelled anomalies in the file. These rates range from 0 to 1, with 0 being the best case: no false positives in the case of FPrate and no misses in the case of FNrate. A statistical significance analysis of the results is performed using the two sample Kolmogorov-Smirnov (KS) test.

5.5 Results

The results of the four sets of experiments described in the previous section will be presented and analysed in subsections 5.5.1 - 5.5.4. Additionally, a statistical analysis of the results will be resented in subsection 5.5.5.

5.5.1 Ensemble formed from the evolved population [Q.off1]

The first set of experiments aim to test whether an ensemble formed from the entire evolved population can be effective in reducing the false positive and false negative rate in environment B.

Mean improvement and probability of improvement - Ensemble of the whole population over RDA_{Abest}

The main finding of this set of experiments is that using an ensemble of the whole population can improve the performance only for low intensity concept drift. The overview in figure 5.12 depicts the mean improvement that is achieved when using an ensemble of 40 members (the whole population), compared to when using the evolved best on the training set RDA_{Abest}. If, for instance, for one run and one signature RDA_{Abest} gives an FPrate of 0.5 and the ensemble gives an FPrate of 0.3, the improvement is 0.2. These pairwise improvements are averaged over all the runs and signatures (85x18), and over all the three files used for environment B. Obviously, a negative value means that RDA_{Abest} performs better than the ensemble, i.e. when using the ensemble the FPrate or FNrate increases. From figure 5.12 improvement can be observed both in the FPrate and FNrate, but only in the lower range of intensities, 300-600. In higher intensities, 700-1000, the opposite happens; using the ensemble actually leads to worse performance, up to (-)0.2 points for the FPrate and (-)0.28 for the FNrate.

This trend is confirmed, by looking at the probability of improvement in figure 5.13. The probability of improvement is calculated as the proportion of all the runs, where the improvement is positive, i.e the ensemble results in a lower FPrate (FNrate) than RDA_{Abest}. The probability of improvement for both FPrate and FNrate decreases with increasing intensity. The false negative rate has a high probability of improving, 80%-90% in the lower band of intensities. On the other hand, the probability of improvement of the FPrate is much lower, ranging from 59% in low intensities to 27% in high intensities. This means that there are cases where an increase in the false positive rate is combined with a decrease in the false negative rate. This decrease of false negatives can sometimes be deceptive: an increase in false positive rate means that the algorithm becomes oversensitive, detecting a lot of irregularities in the data as anomalies. Making constant detections means that the actual labelled anomalies are also likely to be detected. As a result the false negative rate can decrease and the reported improvement can be observed.

Detailed performance for the different test files

More detail can be provided on the performance of the ensemble, by examining the boxplots produced, for each datafile (file 1, 4, or 6) as environment B separately. In the boxplots the performance of the ensemble, marked as $ens40$, is reported both in terms

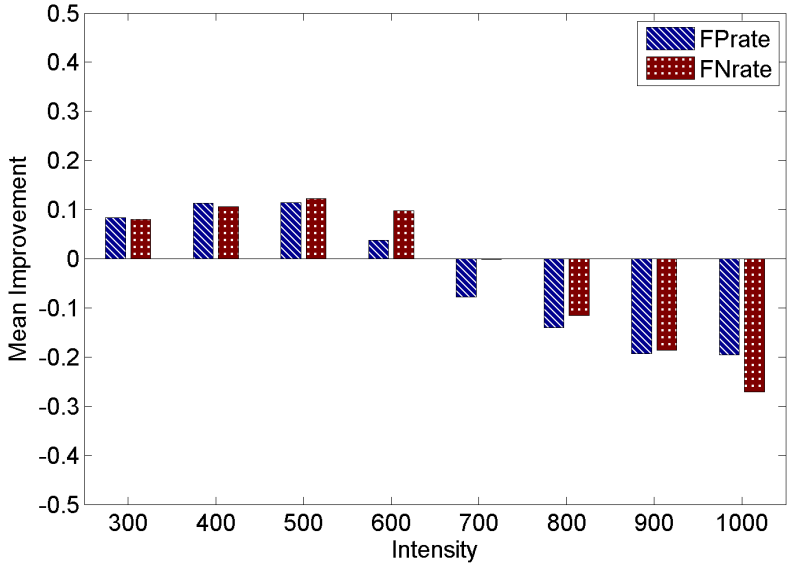


Figure 5.12: Mean improvement when using an ensemble of the entire population (*ens40*), over using RDA_{Abest} in environment B, for all the intensities of concept drift tested. The improvement is defined as the difference between the FP(FN)rate for the two options: $FPrate(RDA_{Abest}) - FPrate(ens40)$. A positive improvement means that the rate of the ensemble is lower (better) that that of RDA_{Abest}. The results are averaged over all the runs and all the three different files for environment B. In low intensities, 300-600, the ensemble on average produces lower FPrates and FNrates compared to RDA_{Abest}, while for higher intensities, 700-1000, it is the RDA_{Abest} that produces lower rates, i.e the use of the ensemble has negative effects.

of FPrate and FNrate. It is compared against the performance of RDA_{Abest}, marked as *Abest*, the average performance of the population *Aavg* and the RDA from the population that performs best in environment B, marked as *Bbest*. Figure 5.14 shows the results for file 1. Note that, for reason of simplicity of visualisation, three intensities are reported. The intensity of $I_{max} = 500$ is characteristic of the lower intensities. Similarly, the intensities $I_{max} = 700$ and $I_{max} = 900$ represent the mid and high intensities for the range tested.

Comparing the ensemble with *Abest*, it is observed that in the relatively low, but still interesting from the point of anomalies and concept drift, intensity of 500, the ensemble achieves a lower range of values, both in false positive and false negative rates. However, as the intensity increases, the performance of the ensemble becomes increasingly unstable. High medians, of over 0.75 for the FPrate are observed, along with high variance, as

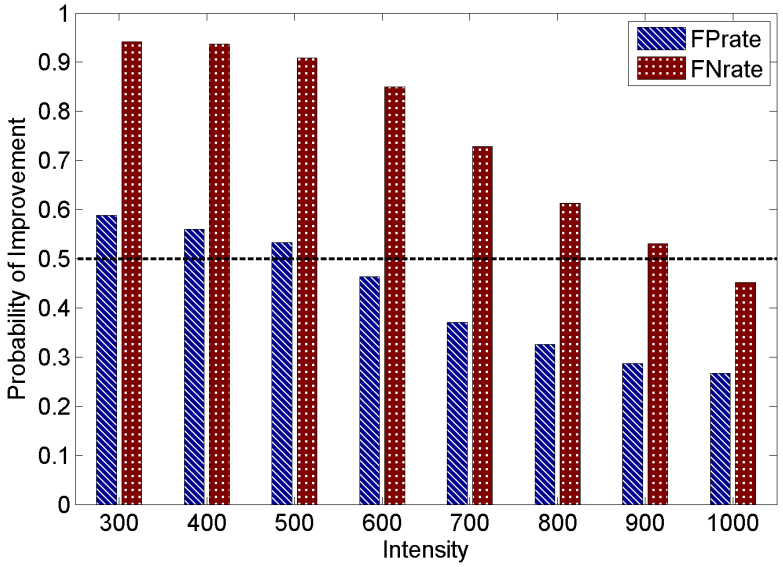


Figure 5.13: Probability of improvement when using an ensemble of the entire population (*ens40*) over using RDA_{Abest} in environment B, for all the intensities of concept drift tested. Probability of improvement is defined as the proportion of runs where a positive improvement has been observed when using the ensemble. The results here are averaged across all runs and all three different files for environment B. The probability of improvement decreases with increasing intensity. A probability less than 0.5 means that RDA_{Abest} is more likely to produce lower FP(FN)rates is environment B.

indicated by the range covered by the respective boxes[3]. The high FPrate medians occur because the accuracy of the ensemble members drops as environment B's background becomes increasingly different than environment A (for which the members were evolved), with increasing intensity of the background signature.

An interesting finding, however, is that it is the average performance of the population which consistently performs worst in both aspects. This indicates that there are a lot of individuals in the population that on their own perform quite poorly, affecting the average, *Aavg*. However, in most cases when they are combined in an ensemble, comparatively lower ranges of FPrates and FNrates are observed, which is something that in principle is consistent with the ensemble theory. This also indicates that the ensemble has some power in decreasing the negative effect of unsuitable base RDAs. Observing that the

[3]On each box, the central mark is the median and the edges of the box are the 25^{th} and 75^{th} percentiles. The upper limit of the box can be useful, as it indicates that 75% of the sample is below this limit. The range covered by the box means that 50% of the sample is contained inside that range. The whiskers extend to the most extreme data points not considered outliers.

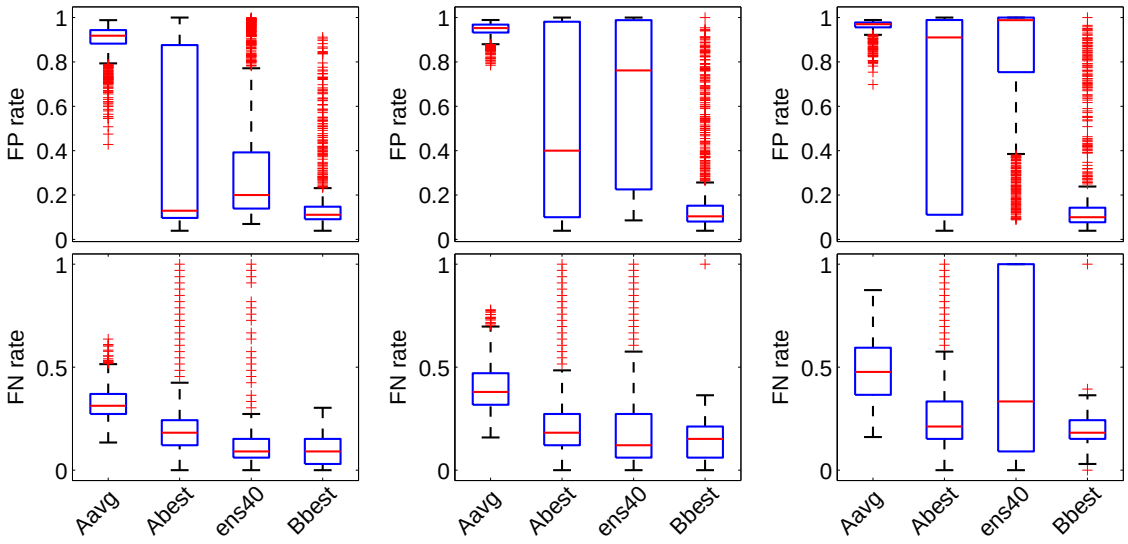
average FPrate and FNrate increases in higher intensities, it can be deduced there are a lot of solutions in the population that perform increasingly worse. The candidates were evolved for a background signature free environment. The higher the intensity of the signature added, the more different the test set will be to the training set. Hence, more and more candidates will not perform well on the test set. The existence of these unsuitable candidates in the ensemble has a grave effect and can explain the low performance in high intensities.

Also presented here is the best individual of the population on environment B, *Bbest*: if one solution was chosen from the population and used on environment B, this would be the best possible outcome. Of course the problem is that there is no way to know *a priori* which is this solution, as there is no feedback/labels available for environment B, i.e. the available solutions cannot be directly assessed on environment B. However, it is included here as a baseline - best case scenario. From figure 5.14, for most runs, there is a solution in the evolved population that leads to very low false positive and false negative rates (near 0). These rates increase as the intensity rises, and there are a few cases where there is no single solution in the evolved population that can lead to reduced values for these rates, indicated by the cluster of outliers in the high FPrate range. This happens because the EA does not always converge to the same areas of the parameter space. Many different parameter sets can lead to good performance in the training set, so the evolution can end up in different solution subspaces. However, the absence in some cases of parameter sets in the final population that can perform well on environment B, shows that some of these subspaces do not achieve any generalisation/adaptation when concept drift is added. This could possibly be dealt with by revisiting the EA.

Using other files as a basis for environment B does not affect the trends of the results, as can be seen in figures 5.15 for file 4 and 5.16 for file 6. This shows that the results are not specific to a test file, but there is good generalisation in other test environments as well. It is reminded here that the file used for training does not change; environment A is always file 1, with no added signature. The same evolved population is tested on three different test files.

Variance analysis

An important observation that cannot be ignored is the high variance in all the results reported in figures 5.14, 5.15 and 5.16. This is considered to be caused by the fact that the



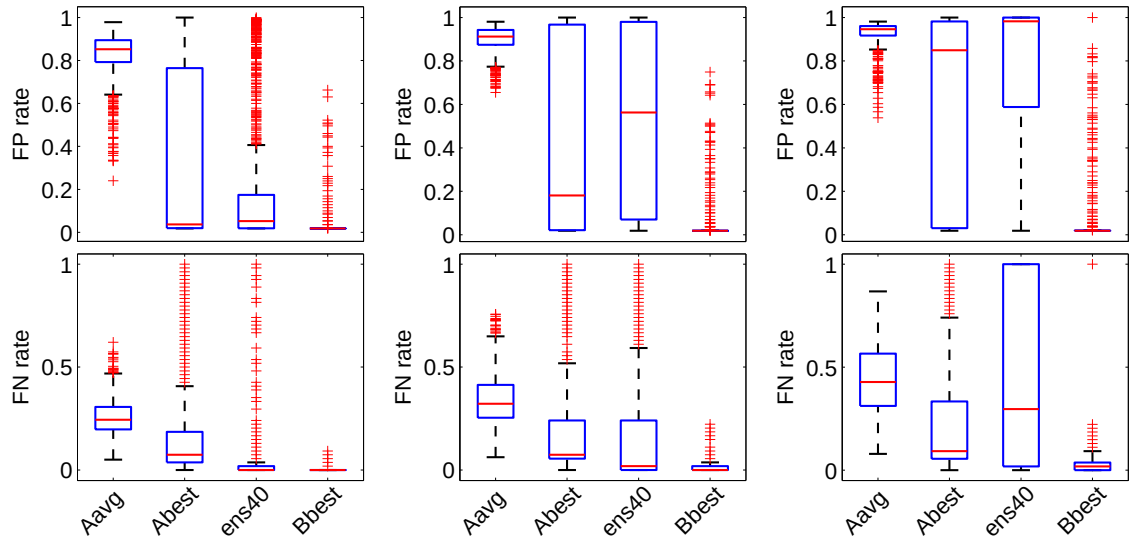
(a) $I_{max} = 500$ (b) $I_{max} = 700$ (c) $I_{max} = 900$

Figure 5.14: FPrate (top row) and FNrate (bottom row) comparisons when file 1 acts as environment B, with added cdrift of intensity $I_{max} = 500$(a), 700(b), 900(c). All the rates are measured for the test set, environment B, over all the repetitions. The boxes from left to right correspond to: *Aavg*, the average FP(FN)rate of all the population, *Abest*, the FP(FN)rate of RDA_{Abest}, *ens*40, the rates of the ensemble that uses all 40 members of the population, and *Bbest* the best rates found in the population. The ensemble does not offer any improvement over RDA_{Abest}, especially in high intensities. Some improvement over *Aavg* can be observed.

population in each run, consists of different sets of individuals. Inspection of the results shows that the high variance is in fact attributed to the evolutionary runs, and not the different signatures or different test files used. For a specific evolved population, the results are reasonably stable across the 18 different signatures used as the background signature. This is positive in the sense that the specific background which the system moves to, does not affect its ability to adapt. However, it creates the need to reduce the variance as well as the rates themselves.

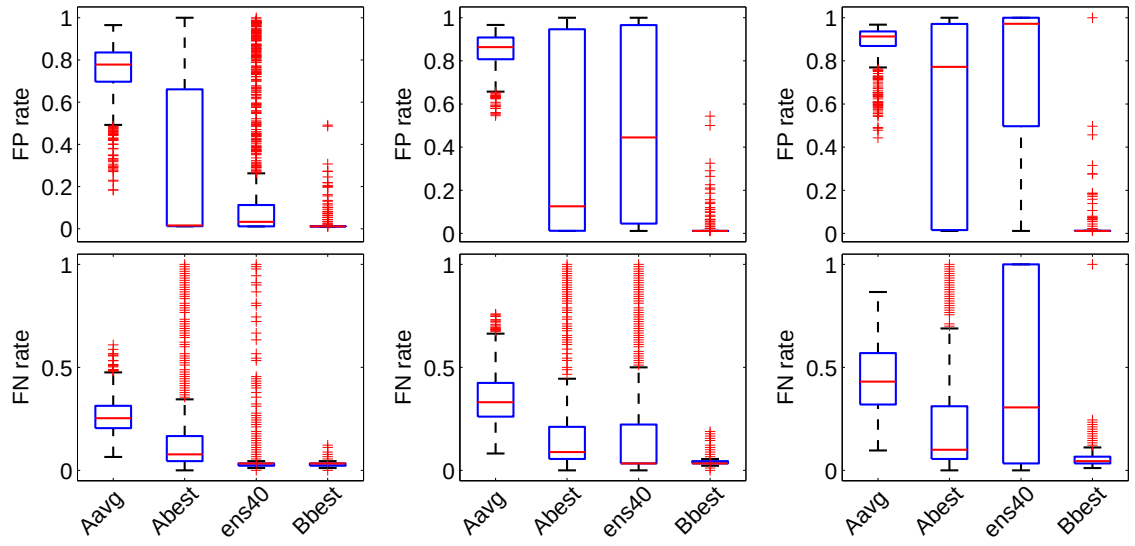
A closer look at the results revealed that the distribution of the results is bimodal, which explains the high number of outliers. The second mode is observed when evolution during training converges to a solution sub-space, which is good for the training set, but does not generalise at all. There are solutions to this, which would entail studying these sub-spaces and driving evolution away from them. However, this is not in the scope of this work.

In this analysis, the results will be assessed given their bimodality. For the results



(a) $I_{max} = 500$ (b) $I_{max} = 700$ (c) $I_{max} = 900$

Figure 5.15: FPrate and FNrate comparisons when file 4 acts as environment B, for intensities $I_{max} =$ 500(a), 700(b), 900(c). The boxes from left to right: *Aavg*, the average FP(FN)rate of all the population, *Abest*, the FP(FN)rate of RDA_{Abest}, *ens*40, the rates of the ensemble of the population, and *Bbest* the best rates in the population. The poor performance of the ensemble is consistent with that observed for file 1.



(a) $I_{max} = 500$ (b) $I_{max} = 700$ (c) $I_{max} = 900$

Figure 5.16: FPrate and FNrate comparisons when file 6 acts as environment B, for intensities I_{max} =500(a), 700(b), 900(c). The boxes from left to right: *Aavg*, the average FP(FN)rate of all the population, *Abest*, the FP(FN)rate of RDA_{Abest}, *ens*40, the rates of the ensemble of the population, and *Bbest* the best rates in the population. The results are consistent with those observed for file 1 and file 4.

presented in boxplots, the bimodality should be taken into account, as the points that are classified as outliers in some cases represent a second mode. However, there is still value in presenting the results in this way, as it provides a good way to visually compare the different settings. In every case the boxplot guarantees that 50% of the data is contained inside the box, and naturally that 50% of the results is below the median. Additional information and alternative ways of presenting the results are employed where necessary, to show in more detail the properties of the results and aid in the correct interpretation.

5.5.2 Selecting ensemble members based on the implicit metrics vector [Q.off2]

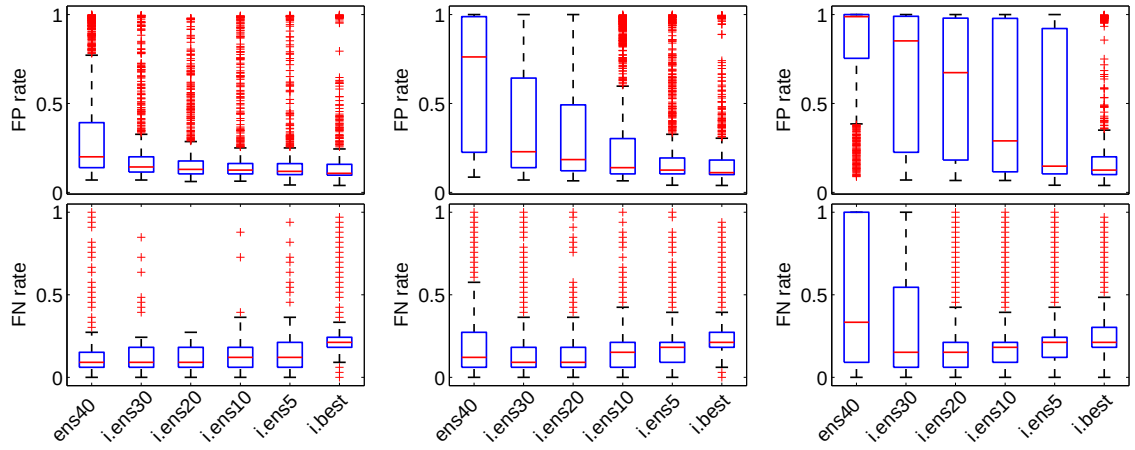
The second set of experiments investigates whether using implicit metrics to select the members of the ensemble, rather than using the whole population, can reduce the observed FPrate and FNrate. The implicit ensemble is formed from the set of solutions whose implicit vectors have the highest affinity to the reference implicit vector, $IM_{A,best}$. In the results that follow, different sizes of this implicit ensemble are tested and compared against an ensemble of the whole population. The ensemble size is varied from 40 to 1, 40 being the full ensemble, $ens40$, that uses the entire population and 1 being just a single RDA, noted as $i.best$, the one whose IM vector is closest to the reference IM vector. For an intermediate size x, the implicit ensemble is denoted as $i.ensx$.

The results presented in figure 5.17 for file 1 as environment B show that, even though there is still high variance and clusters of outliers that indicate bimodality, reducing the ensemble size in every intensity leads to a significant reduction of the false positive rate. Both the median FPrate and the range of the boxes (which represents 50% of the results) decrease as less members are used. In low intensities, $I_{max} = 500$, the median FPrate decreases from 0.2 for $ens40$, to 0.1 for just one member, $i.best$. In higher intensities the decrease is more dramatic, as the full ensemble performs poorly. In particular, the respective FPrates decrease from 0.76 to 0.11 for $I_{max} = 700$ and from 0.98 to 0.13 for $I_{max} = 900$. Comparative medians to the $i.best$ are also found for the ensemble of 5, $i.ens5$ and in low intensities for the ensemble of 10, $i.ens10$.

The FNrates, on the other hand, seem to rise as less members are used (with the exception of the high FNrates for $ens40$), as shown in 5.17. This is due to the trade-off effect between FPrate and FNrate: note that usually the lowest false negative rates are achieved for cases with high false positive rates. As mentioned earlier, this is because a

very sensitive detector tends to detect noise or irregularities in the data, thus increasing the probability of detecting actual anomalies as well. Cases where both the FPrate and FNrate are high, for instance the case of *ens*40 (first box) in figure 5.17c, are usually associated with false detections that span over a big duration. True anomalies in the data that happen in this time are masked by the false detection, hence the number of missed anomalies rises. In the case of the FNrates the best performance, from the three settings with the lowest FPrates, is observed for an ensemble of 10, with FNrates ranging from 0.12 to 0.18, while the ensemble of 5 tracks these rates very closely.

Overall, it can be observed that the best results are achieved for implicit ensembles of sizes 5 and 10. The single best individual, *i.best*, yields comparable and possibly better false positive rates than the ensemble of 5, but this comes at the expense of a slightly higher false negative rate.



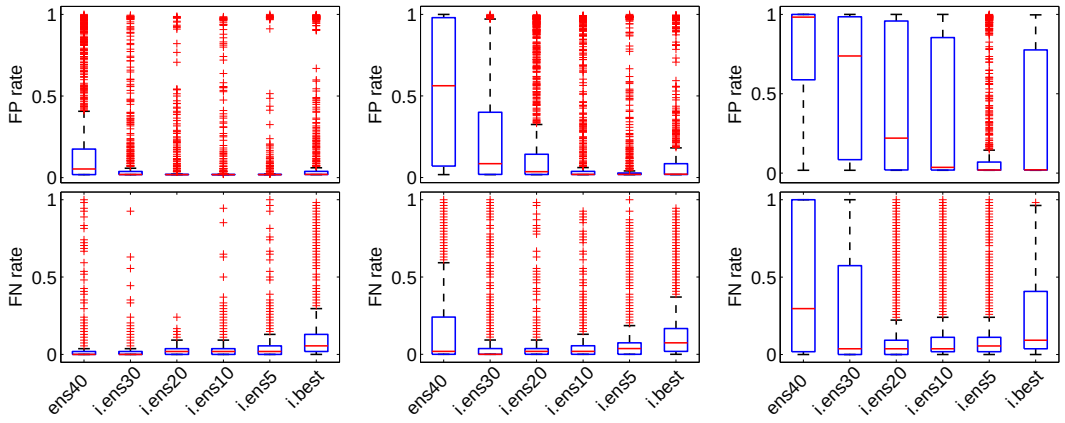
(a) $I_{max} = 500$ (b) $I_{max} = 700$ (c) $I_{max} = 900$

Figure 5.17: FPrate and FNrate comparisons when file 1 acts as environment B, for intensities $I_{max} =$ 500(a), 700(b), 900(c). All the rates are measured for the test set, environment B, over all the repetitions. The boxes from left to right correspond to: *ens*40, the full ensemble; *i.ens*30 to *i.ens*5, the implicit ensemble using 30 to 5 members of the population whose IM vectors have the highest affinity to the reference IM vector; *i.best*, the individual from the population with the highest affinity IM vector. The FPrate decreases as the ensemble is pruned. The small increase in the FNrate happens because less detections are generally made.

In figures 5.18 and 5.19 the advantage of the ensemble compared to a single best individual, based on IM, is more clearly demonstrated. In the case of the FPrate, improvement is observed when decreasing the size of the ensemble down to an ensemble of 5, with FPrates in the 0-0.1 range. In the case of *i.best*, increased FPrates are observed,

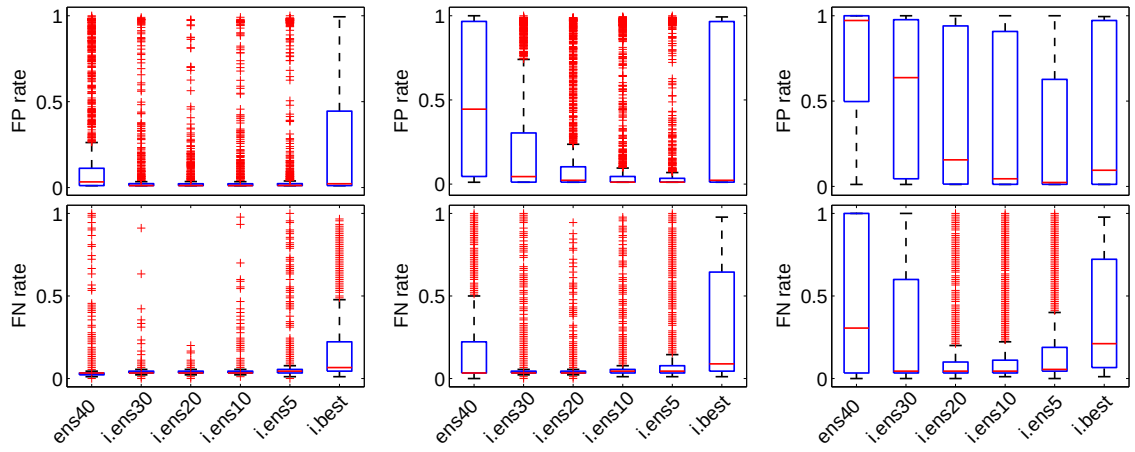
both in terms of median, but also in the upper limit of the box, which indicates the 75^{th} percentile (75% of the sample falls below that value). The FNrates observed for *i.best* are also higher compared to *i.ens*10 and *i.ens*5. The difference of these graphs to figure 5.18 is that a different file is used as a basis for environment B, than the one used for training. It appears that when environment B is based on the same data file as the one used for training, the IM vector is more efficient in estimating the accuracy, as exactly the same set of detections is expected in both environments. However, when IM_B and the reference $IM_{A,best}$ are calculated on environments that are based on different files, containing different sets of anomalies to be detected, the single solution closest to the reference implicit vector does not perform that well.

On the other hand, the ensembles maintain their performance to similar levels independently of the file used as environment B. This shows good generalisation. From this it can be deduced that the implicit metrics selection method cannot be used to accurately locate the best solution in the population, but it is very useful in selecting a set of reasonable RDAs or in pruning out of the ensemble RDAs that do not demonstrate an expected behaviour.



(a) $I_{max} = 500$ (b) $I_{max} = 700$ (c) $I_{max} = 900$

Figure 5.18: FPrate and FNrate comparisons when file 4 acts as environment B, for intensities $I_{max} =$ 500(a), 700(b), 900(c). All the rates are measured for the test set, environment B, over all the repetitions. The boxes from left to right correspond to: *ens*40, the full ensemble; *i.ens*30 to *i.ens*5, the implicit ensemble, using 30 to 5 members of the population whose IM vectors have the highest affinity to the reference IM vector; *i.best*, the individual from the population with the highest affinity IM vector. The difference to file 1 is the high rates observed for *i.best*.



(a) $I_{max} = 500$ (b) $I_{max} = 700$ (c) $I_{max} = 900$

Figure 5.19: FPrate and FNrate comparisons when file 6 acts as environment B, for intensities $I_{max} =$ 500(a), 700(b), 900(c). The boxes from left to right correspond to: *ens*40, the full ensemble; *i.ens*30 to *i.ens*5, the implicit ensemble, using 30 to 5 members of the population whose IM vectors have the highest affinity to the reference IM vector; *i.best*, the individual from the population with the highest affinity IM vector. The observed results are consistent with file 4.

Taking into account the bimodality and variance

There are a few options on how to further analyse the data and address the outliers and the bimodality of the results. One way is to attempt to fit the data in a bimodal distribution and then analyse and compare the two peaks. However, this would involve making assumptions about the generating distribution. Instead, a practical way of comparing the results is proposed. In table 5.2 the percentage of the sample under an FPrate limit of 0.2 is reported. An FPrate of 0.2 means that 20% of the detections are false positives, or 80% of the detections correspond to real anomalies. Observing the results, it is determined that this is an FPrate where the differences between the tested settings are obvious. Additionally, given the presence of concept drift, an FPrate of 0.2 is a large improvement, considering the high (over 0.5) medians and 75th percentile values observed for RDA_{Abest} and *ens*40 in the mid-high intensities (see figures 5.14-5.16). For the interested reader, however, additional tables for an FPrate limit of 0.1 and 0.3 are provided in appendix B (tables B.1 and B.2). In this analysis, only the FPrate is considered, because the results show that it is the most critical of the two, and the one affected more by the added concept drift.

Table 5.2 is a summary of the results presented in figures 5.14-5.16 and 5.17-5.19. In

B	I_{max}	***Abest***	***ens40***	***i.ens30***	***i.ens20***	***i.ens10***	***i.ens5***	***i.best***	*Bbest*
1	500	57.52	48.50	74.38	82.03	85.42	**87.58**	84.18	87.06
1	700	44.44	20.59	42.94	52.35	67.52	75.82	**78.50**	84.05
1	900	32.09	8.63	21.50	26.93	44.51	59.87	**74.97**	81.57
4	500	63.59	76.80	94.25	96.54	96.73	**98.24**	85.10	98.04
4	700	50.46	39.02	67.06	78.37	85.88	**92.29**	79.28	97.39
4	900	37.25	16.86	36.14	48.82	66.01	**80.98**	72.55	95.10
6	500	64.97	81.96	95.42	**97.39**	95.49	93.92	69.54	99.28
6	700	51.57	42.09	70.39	79.87	84.44	**85.95**	62.55	99.28
6	900	39.22	18.95	39.02	51.50	61.05	**71.11**	51.76	98.56

Table 5.2: Percentage of repetitions that achieve an FPrate lower than 0.2, under different ensemble settings. Results from all the files and intensities are reported. The settings compared are RDA_{Abest} (*Abest*); the full ensemble (*ens*40); implicit ensembles of 30, 20, 10 and 5 members (*i.ens*30-*i.ens*5); the implicit best (*i.best*); and *Bbest*, the best solution existing in the population, for reference. Noted in bold is the winning setting, with the highest percentage in the 0-0.2 FPrate range.

the majority of the cases (6/9) the ensemble of 5 achieves the best performance with often more than 85% of the population achieving a false positive rate lower than 0.2. The improvement over both RDA_{Abest} and a full ensemble is clear. It can also be noted that in some cases the performance of an ensemble of five is comparable to the *Bbest*, which comprises of the best individual present in the population. In one of the reported cases (B4, $I_{max} = 500$) the ensemble of five outperforms the baseline. These results show that selecting the members of the ensemble based on their IM is meaningful and leads to improved performance.

Mean improvement and probability of improvement - Ensemble of 5 over RDA_{Abest}

In figure 5.20 the mean improvement over RDA_{Abest} is shown, when an ensemble of 5 is used. This is averaged over all the runs in all the three test environments. Moreover, the probability of improvement when using the ensemble of 5 over the previously known "best" solution, RDA_{Abest}, is depicted in figure 5.21 and it can be used to draw conclusions independently of the bimodality of the results.

In contrast to the improvement of the full ensemble over RDA_{Abest} (figure 5.12) the average improvement, in this case, is positive over all the intensities tested. On average the FPrate improves up to 0.32 points, while the FN rate improves on average around

0.12 points. The probability of having an improved result when using an ensemble of 5 members ranges from 65% to 88%. The fact that the probability of improvement is always well over 0.5 shows that in every intensity there is high probability that the results will be improved than not, i.e. the improvement of the performance is consistent.

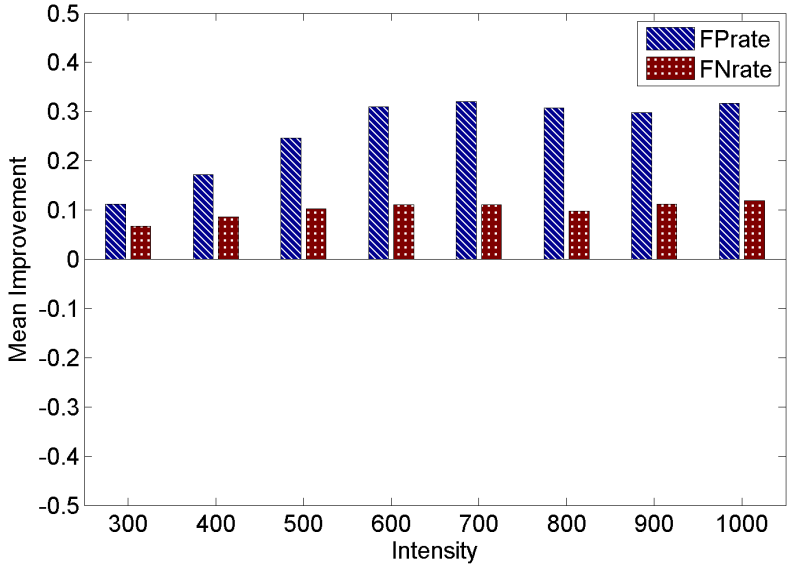


Figure 5.20: Mean improvement when using an ensemble of 5 members, selected based on IM, over using RDA_{Abest} in environment B, for all the intensities of concept drift tested. The improvement is defined as $FPrate(RDA_{Abest}) - FPrate(i.ens5)$. The positive improvement means that the rate of the ensemble of 5 is on average lower (better) than that of RDA_{Abest}, both for the FPrate and the FNrate. The results are averaged over all the runs and over all three different files for environment B.

5.5.3 Implicit metrics against other selection mechanisms [Q.off3]

The results in this section compare the implicit metrics, as a selection mechanism for the members of the ensemble, to other methods: using the Pareto front or the bounded optimality method for selection. Ranking based on these two methods is presented in section 5.3.4. The Pareto front that the EA outputs, typically contains around ten individuals (an effect of the elite size being ten). For this reason, an ensemble of ten members is used for the other two methods as well. Although the previous results indicate that an implicit ensemble of five is the best, here an implicit ensemble of ten is used to ensure that any differences observed are attributed to the selection method and not to the size of the ensemble.

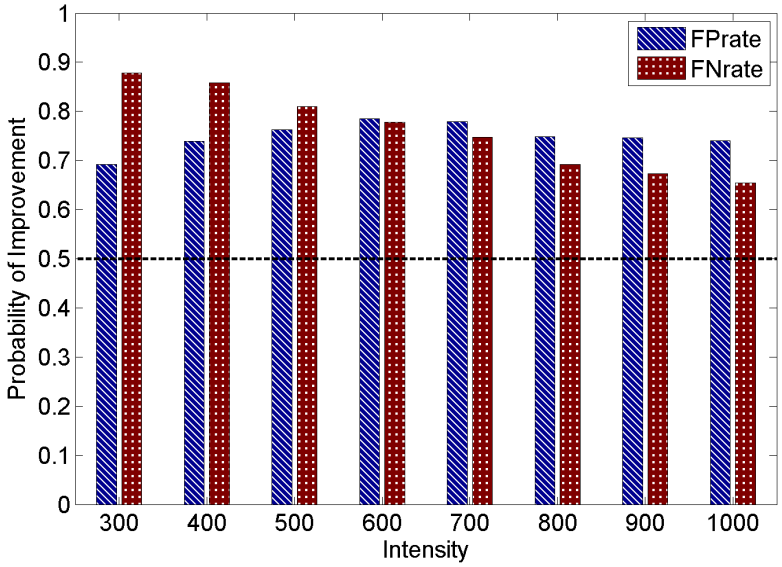


Figure 5.21: Probability of improvement when using an ensemble of 5 members, selected based on IM, over using RDA_{Abest} in environment B, for all the intensities of concept drift tested. Probability of improvement is defined as the proportion of runs where a positive improvement has been observed when using the ensemble. The probability of improving the FPrate or FNrate when using an ensemble ranges from 65% to 88%. The results are averaged over all the runs and all three different files for environment B.

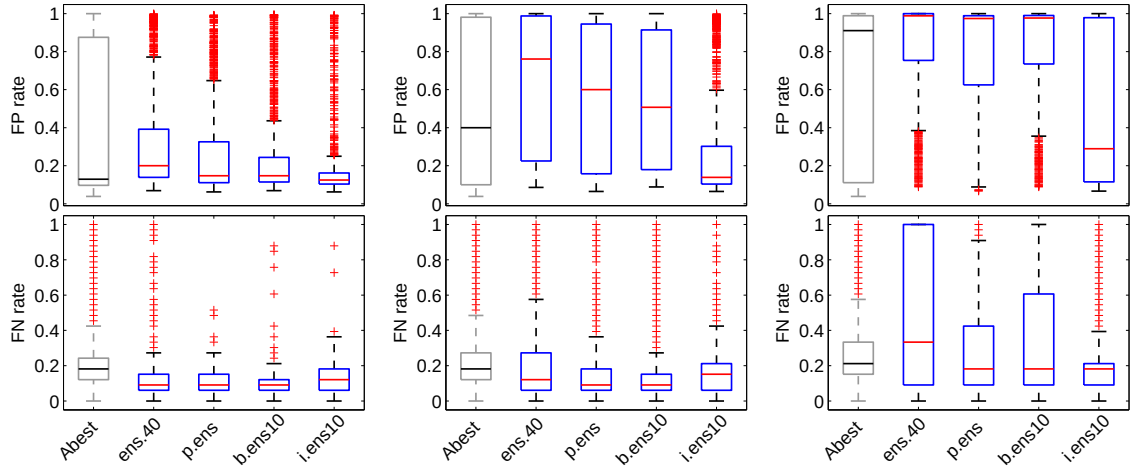
Detailed performance for the different test files.

The boxplots of figure 5.22 include results for these three selection methods, while the RDA_{Abest} rates and full ensemble rates are also presented for reference. The results indicate that the implicit ensemble, $i.ens10$, achieves lower FPrate medians than both the Pareto and the bounded selection method, which perform comparably to each other. In $I_{max} = 500$, the median reduces from 0.15 (both for $p.ens$ and $b.ens10$) to 0.12, while there is a decease of the 75th percentile to 0.16 from 0.32 for $p.ens$ and 0.24 for $b.ens10$. In $I_{max} = 700$, the median reduces from 0.6 ($p.ens$) and 0.5 ($b.ens10$) to 0.14, while the 75th percentile is decreased from 0.9 to 0.3. In the high intensity of 900, there is very high variance in all the cases, but still the implicit ensemble achieves good FPrates in more runs, decreasing the median greatly, from 0.97 to 0.28. These decreases in FPrate are often accompanied by a slightly elevated false negative rate, but as explained earlier this is to be expected.

As the intensities increase, it becomes clear that selecting the ensemble members based on their performance in the training set, which is what both $p.ens$ and $b.ens10$ do, leads to

poor performance. This means that there are solutions in the population that contribute in a good ensemble detection, which do not belong to an environment A elite.

The same observations can be made when other files are used for environment B, as seen in figures 5.23 and 5.24. Again, the biggest improvements are observed for medium and high intensities. For file 4 acting as environment B (figure 5.23), the median FPrate is improved from 0.39 (*p.ens*) and 0.26 (*b.ens*10) to 0.02 for *i.ens*10 in the intensity of 700. There is great improvement in the FPrate median in the 900 intensity as well, but again it is accompanied by an observed high variance. Finally for environment B using file 6, the same trends are observed, with *p.ens* achieving an FPrate median of 0.26 in $I_{max} = 700$, *b.ens*10 a median 0.2 FPrate and *i.ens*10 a median FPrate of 0.01. The variance observation stands here as well.

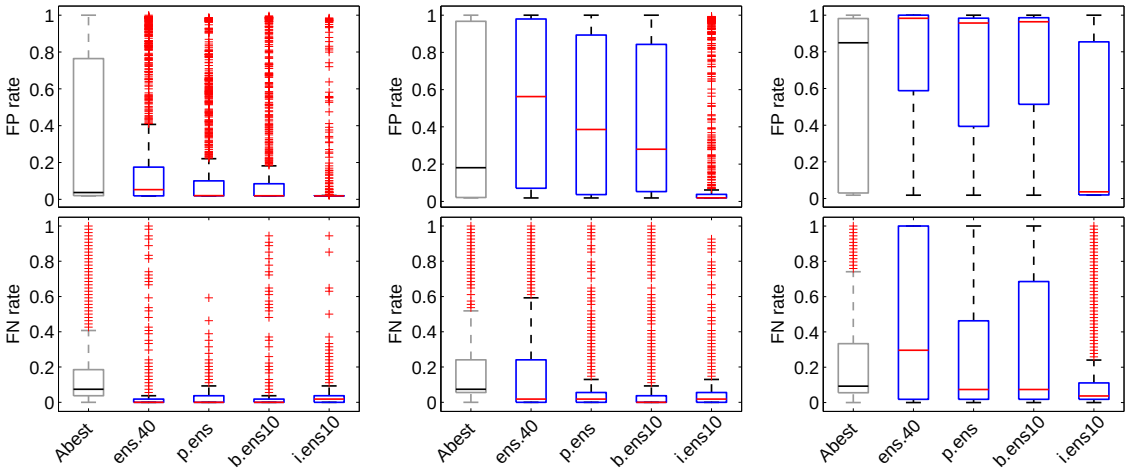


(a) $I_{max} = 500$ (b) $I_{max} = 700$ (c) $I_{max} = 900$

Figure 5.22: FPrate and FNrate comparisons when file 1 acts as environment B, for intensities $I_{max} =$ 500(a), 700(b), 900(c). All the rates are measured for the test set, environment B, over all the repetitions. The boxes from left to right correspond to: RDA_{Abest} (*Abest*), the full ensemble (*ens.*40), the ensemble of the individuals in the Pareto front (*p.ens*), the ensemble formed from the best 10 individuals based on bounded ranking (*b.ens*10) and the implicit ensemble of 10 members (*i.ens*10). The implicit ensemble achieves lower medians in all the intensities.

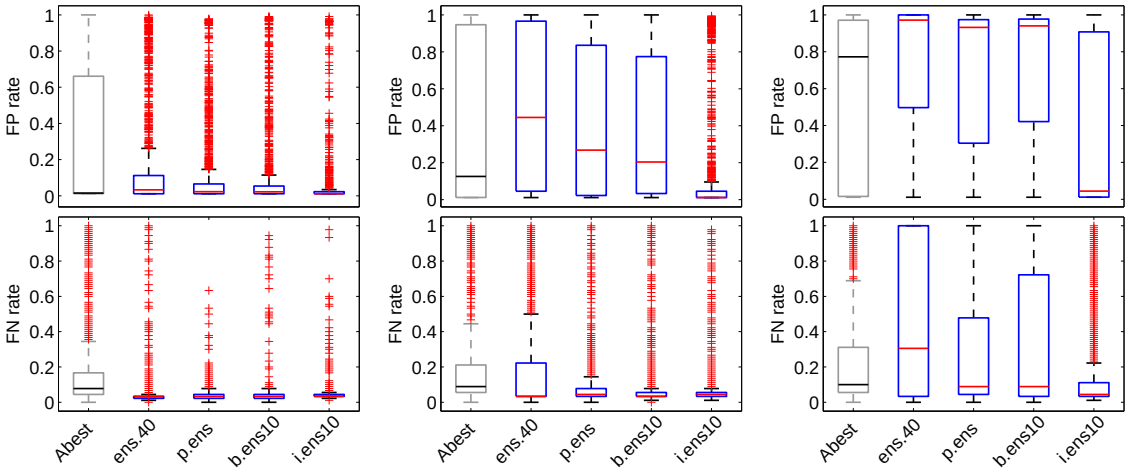
Mean improvement and probability of improvement - implicit ensemble of 10 over Pareto ensemble of 10

The overview figure for all the environments of the mean improvement when using the implicit ensemble over the Pareto ensemble, is presented in figure 5.25. The mean improvement is around zero, or even negative for the false negative rate for a range of in-



(a) $I_{max} = 500$ (b) $I_{max} = 700$ (c) $I_{max} = 900$

Figure 5.23: FPrate and FNrate comparisons when file 4 acts as environment B, for intensities $I_{max} =$ 500(a), 700(b), 900(c). All the rates are measured for the test set, environment B, over all the repetitions. The boxes from left to right correspond to: RDA_{Abest} (*Abest*), the full ensemble (ens.40), the ensemble of the individuals in the Pareto front (*p.ens*), the ensemble formed from the best 10 individuals based on bounded ranking (*b.ens*10) and the implicit ensemble of 10 members (*i.ens*10).



(a) $I_{max} = 500$ (b) $I_{max} = 700$ (c) $I_{max} = 900$

Figure 5.24: FPrate and FNrate comparisons when file 6 acts as environment B, for intensities $I_{max} =$ 500(a), 700(b), 900(c). All the rates are measured for the test set, environment B, over all the repetitions. The boxes from left to right correspond to: RDA_{Abest} (*Abest*), the full ensemble (ens.40), the ensemble of the individuals in the Pareto front (*p.ens*), the ensemble formed from the best 10 individuals based on bounded ranking (*b.ens*10) and the implicit ensemble of 10 members (*i.ens*10). As in the other files, the implicit ensemble achieves lower medians in all the intensities.

tensities, but that is clearly connected to the improvement of the false positive rate: with worse FPrates more irregularities in the data are detected, among them possibly a few anomalies. The false positive rate improves from 0.02 units in low intensities up to 0.34 units in high intensities. In high intensities the FNrate also improves (decreases) up to 0.18 units. This is because in high intensities the Pareto ensemble produces false positives that span over large portions of the data and mask true anomalies. The implicit ensemble manages to reduce this type of behaviour. The probability of improvement as shown in figure 5.26, is consistently over 50% for both rates. For most intensities the FPrate has a probability of improvement near 80%, and if combined with the mean improvement results in figure 5.25, the improvement is likely to be quite high.

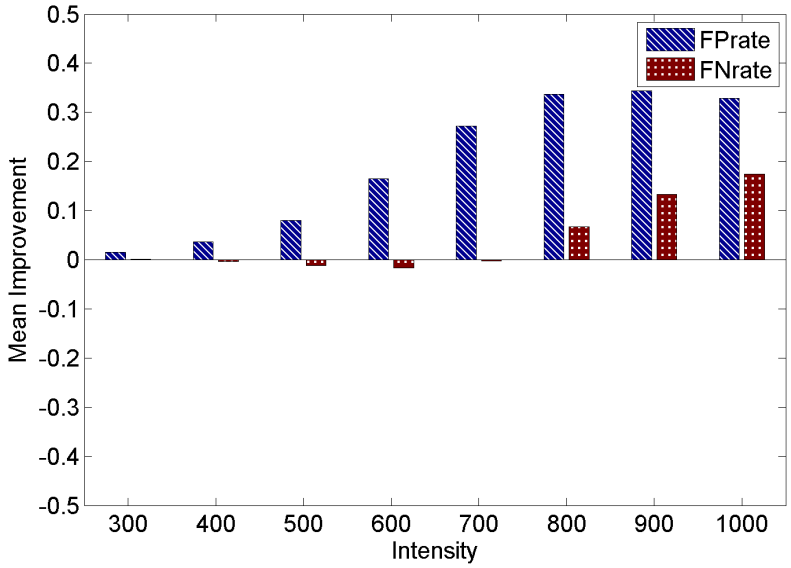


Figure 5.25: Mean improvement when using an implicit ensemble of 10 members over using a Pareto ensemble in environment B, for all the intensities of concept drift tested. The improvement is defined as the difference between the FP(FN)rate for the two options: $FPrate(p.ens) - FPrate(i.ens10)$. A positive improvement means that the rate of $i.ens10$ is lower (better) than that of $p.ens$. The results are averaged over all the runs and all three different files for environment B. The implicit ensemble results in mean improvements of up to 0.34 (FPrate) and 0.18 (FNrate) in the higher intensities.

Taking into account the bimodality and variance

A table that documents the percentage of runs that achieve an FPrate below the limit of 0.2 (table 5.3) is presented here as well (for equivalent tables for an FPrate limit of 0.1 and 0.3 please refer to appendix B, tables B.3 and B.4). The percentage of resulting FPrates up to 0.2 is higher in the case of implicit metrics being used for selecting the ensemble

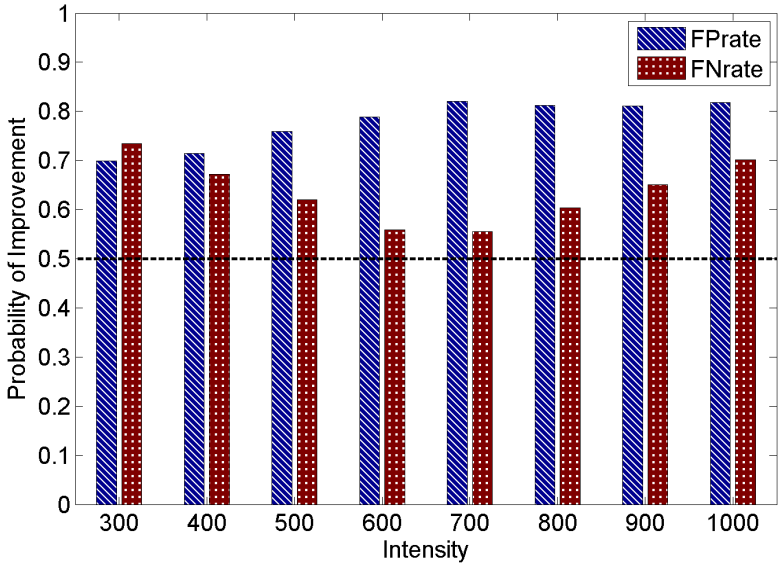


Figure 5.26: Probability of improvement when using an implicit ensemble of 10 members over using a Pareto ensemble in environment B, for all the intensities of concept drift tested. The probability of improvement is defined as the proportion of runs where a positive improvement has been observed when selecting the ensemble members based on the implicit metrics, rather than Pareto optimality. The results are averaged over all the runs and all three different files for environment B. The FPrate has a probability of 69%-82% of improving, and the FNrate a probability of 55%-73%, when using an implicit ensemble of 10.

members, for every environment and every intensity tested. On the other hand, the two methods of selection based on Pareto optimality or bounded optimality exhibit similar performance, which is poor compared to IM pruning.

5.5.4 Online Ensemble formation [Q.On]

The results presented in this section refer to the case of creating the ensemble online. The convergence to an offline performance is examined as well as the minimum time that needs to have elapsed before the IM vector for environment B can be calculated. In order to aid with the interpretation of the results, the online ensemble creation setup is reminded in figure 5.27. The ensemble can be created online, by calculating the IM_B on the data collected so far. Normally the updated ensemble would then be used from then on. However, to facilitate comparison, here all the ensembles created at different timepoints are evaluated over the entire file, acting as environment B.

In figure 5.28 the average FPrate and FNrate is presented, for all the files and intensities, $I_{max} = 500, 700, 900$. An overall observation is that, for all files and intensities, the performance is relatively unstable when only the first 2000-4000 timesteps are used to cre-

B	I_{max}	*Abest*	*ens40*	*p.ens*	*b.ens10*	*i.ens10*
1	500	57.52	48.5	61.90	67.19	**85.42**
1	700	44.44	20.59	31.76	28.10	**67.52**
1	900	32.09	8.63	12.88	10.65	**44.51**
4	500	63.59	76.80	81.63	85.49	**96.73**
4	700	50.46	39.02	44.64	45.16	**85.88**
4	900	37.25	16.86	20.26	17.78	**66.01**
6	500	64.97	81.96	85.82	89.61	**95.49**
6	700	51.57	42.09	47.19	49.67	**84.44**
6	900	39.22	18.95	22.22	19.15	**61.05**

Table 5.3: Percentage of repetitions that achieve an FPrate lower than 0.2, under different selection strategies of the members of the ensemble. Results from all the files and intensities are reported. The settings compared are RDA_{Abest} (*Abest*), the full ensemble (*ens*40), the ensemble of the Pareto front individuals (*p.ens*), the ensemble formed from the best 10 individuals based on bounded ranking (*b.ens*10) and the implicit ensemble of 10 members (*i.ens*10). Noted in bold is the winning setting, with the highest percentage in the 0-0.2 FPrate range.

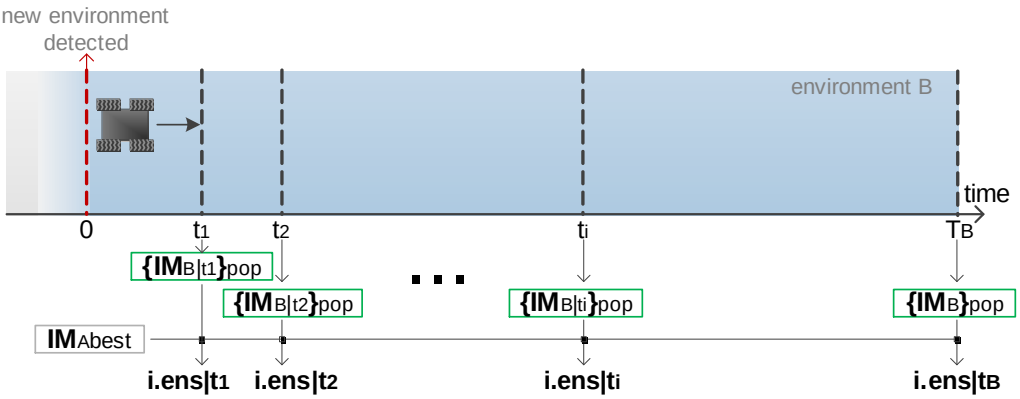


Figure 5.27: Constructing the ensemble online. The ensemble is created by constructing the implicit vector for the population over environment B. In the case where the ensemble is created online, the IM vector also has to be calculated online. By updating the IM vector at every timestep, a new ensemble that incorporates the most recent data can be regularly created (at times t_i) based on the IM. For every time point, t_i, at which an implicit vector is calculated, all the data from the beginning of the environment is used, i.e. the implicit metrics vector $IM|t_i$ is calculated over $[0 \quad t_i]$. From these an updated ensemble, $i.ens5|t_i$, can be created. Assuming that the size/duration of environment B is t_B, then the final ensemble, $i.ens5|t_B$, will be the same as the offline ensemble, created using the whole data file.

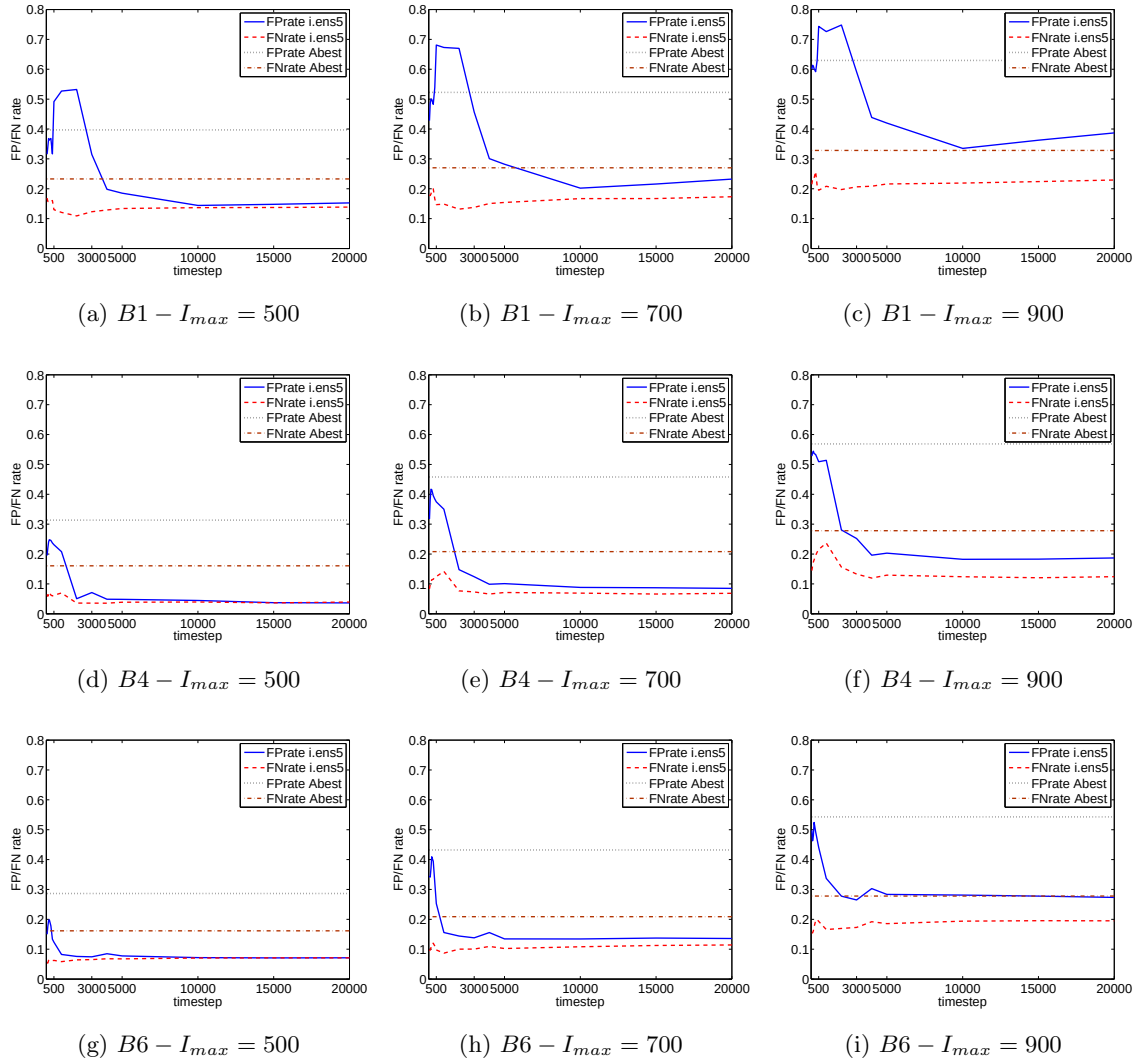
ate the ensemble, but after t=4000 the performance begins to converge. Considering that the frequency of the mass-spectrometer used to collect the data is 3Hz, 4000 timesteps correspond to approximately 22 minutes. The average rates of RDA_{Abest} are also depicted.

For environments B4 and B6 even when as little as 100 timesteps are used to calculate the implicit vector, the performance of the ensemble, both in terms of FPrate and FNrate, is on average still better (the rates are lower) that the performance of RDA_{Abest}.

In environment B1 (file 1), the performance using the first timepoints is better than RDA_{Abest}, but as more timesteps are added, the FPrate increases approximately by 0.15 over the one observed for RDA_{Abest}. This big increase, which is not present in the other environments, is caused by the specific file, file 1. In this file many high strength anomalies are introduced in the first 2000 timesteps. The implicit metrics vector is used to quantify the steady state performance of the algorithm. However, the first portion of the data in file 1 does not correspond to a steady state situation, as the frequency of anomalies is higher than the overall frequency of anomalies in the whole file/environment. As a result, the ensemble constructed from implicit performance indicators for this portion of the data, performs poorly. As more data is collected, the detection patterns converge to a steady state behaviour. The FPrate falls to 0.15 for $I_{max} = 500$ and 0.33 for $I_{max} = 700$.

An interesting effect observed in B1 (file 1 used as environment B), is that the performance is better only when a few timesteps are used: the FPrate is lower in the beginning (100-200 timesteps), then it rises (1000-2000) and finally it converges. The early relatively low FPrate is attributed to the fact that there are no anomalies usually in the files for the first 100-200 timesteps. This situation is somewhat closer to the steady state system, because the portions of normal measurements in the data is generally larger than the portion that corresponds to anomalies (as is true for most anomaly detection applications). This pattern is also present in B4 and B6, but it is more clear in environment B1, because of the composition of the specific file.

Similarly to previous experiments, a table is presented here as well with the percentage of FPrates under 0.2, for all the timesteps at which the ensemble is tested (table 5.4). Similar trends are observed. The worst performances are documented when 0-2000 timesteps are used for the construction of the ensemble, while after timestep 4000 the performance starts becoming stable. A secondary pattern that is observed here, but also in the boxplots of figure 5.28, especially for B1, is that the best performance is not necessarily achieved when the entire file is used (best performance achieved around timestep 10000). Again, this is associated with the composition and the distribution of the anomalies in the files. However, the fluctuations in performance are small after 5000 timesteps. After 4000-5000 timesteps the FPrate can be considered to be converging towards the offline FPrate (the



(a) $B1 - I_{max} = 500$ (b) $B1 - I_{max} = 700$ (c) $B1 - I_{max} = 900$

(d) $B4 - I_{max} = 500$ (e) $B4 - I_{max} = 700$ (f) $B4 - I_{max} = 900$

(g) $B6 - I_{max} = 500$ (h) $B6 - I_{max} = 700$ (i) $B6 - I_{max} = 900$

Figure 5.28: Online experiments. Average performance of an implicit ensemble of 5 members ($i.ens5$), as a function of the timesteps used to calculate the implicit vector. Even though the ensemble is created at different timesteps, the entire file is used for evaluation to facilitate comparison. Every row corresponds to a different file: in figures a-c, file 1 is acting as environment B (B1). In figures d-f, file 4 is acting as environment B (B4). In figures g-i, file 6 is acting as environment B (B6). Both the FPrate and FNrate are plotted, while the corresponding rates for RDA_{Abest} ($Abest$) are also shown for reference.

one corresponding to T=20000). Therefore, it could be argued that the minimum time that has to have elapsed for the calculation of the IM vector, so that low FPrates can be achieved, is 4000-5000 timesteps (22-28 minutes). For equivalent tables for an FPrate limit of 0.1 and 0.3 please refer to appendix B (tables B.5 and B.6).

B	I_{max}	T (timestep at which implicit metrics are calculated)														
		50	100	150	200	300	400	500	1000	2000	3000	4000	5000	10000	15000	20000
1	500	62.35	58.82	54.97	55.49	54.44	60.78	34.71	28.24	*25.95*	54.12	74.25	77.52	**88.43**	88.24	87.58
1	700	46.80	43.40	43.01	43.86	45.49	35.69	16.54	16.41	*13.73*	36.73	60.33	63.53	**79.22**	78.30	75.82
1	900	32.09	33.59	35.03	35.36	35.69	28.30	13.92	13.59	*11.70*	26.86	49.15	51.63	**64.71**	62.48	59.87
4	500	75.03	73.27	69.93	69.74	*68.89*	69.15	70.46	73.33	96.08	93.46	96.34	96.14	97.12	98.10	**98.24**
4	700	60.26	55.03	*50.33*	50.78	52.61	54.71	56.08	58.95	83.27	86.99	90.20	90.20	92.09	92.22	**92.29**
4	900	39.48	*38.95*	39.15	40.13	39.28	40.59	42.42	42.42	69.74	70.65	78.95	78.17	**80.98**	80.85	80.98
6	500	80.72	76.41	74.12	*72.81*	74.84	81.05	83.07	91.24	92.88	93.01	91.83	92.81	**93.92**	93.79	93.92
6	700	56.27	57.52	54.38	*47.52*	48.56	58.10	67.91	81.70	84.18	84.84	82.94	85.75	**85.95**	85.56	85.95
6	900	37.52	43.46	42.75	*36.86*	41.18	44.25	47.71	60.78	68.56	70.52	66.47	68.56	69.93	70.52	**71.11**

Table 5.4: Comparison of ensembles of 5 members created at different timepoints, under the online ensemble paradigm. Percentage of resulting FPrates lower than 0.2. The timestep, at which the maximum FPrate is observed, is noted in bold and the minimum in italics.

5.5.5 Statistical significance

The results presented in the previous sections are tested for statistical significance. Treating the FPrate and FNrate separately, pairs of settings are tested, for example $i.ens20$-$i.ens10$. The null hypothesis is formulated as H_0: the FP(FN)rate observed for setting1 is the same as the FP(FN) rate observed for setting2. The null hypothesis is tested using the non-parametric two sample Kolmogorov-Smirnov (KS) test at a 5% significance level. A rejection of the null hypothesis means that the FP(FN)rates of the two settings are significantly different. The numbers reported in tables 5.5 - 5.7 correspond to the number of files (out of the three tested for environment B) for which the difference between the two settings is found significant. If the number 0 is reported, this means that the differences were not found significant in any of the three environment B's. Similarly the numbers 1,2,3 mean that the differences were found significant in one, two or three out of the three files used for environment B. A pair of settings will be considered here significantly different, if one or both the FPrate, FNrate is found significantly different on all three files: for instance, having a significantly different FP rate with a comparable FNrate (no significant differences) makes the two settings different.

In table 5.5, the existence of significant differences between the use of RDA_{Abest} and an ensemble and then between successive reductions of the ensemble size is examined. In the case of the RDA_{Abest}-to-$ens40$, $ens40$-to-$i.ens30$ and $i.ens5$-to-$i.best$ pairs there is significance for all the intensities. However, because of the high variance, the null hypothesis is not rejected for all the files/intensities in the other pairs tested. Given the variance, bigger reductions in the ensemble size are considered in a second set of tests, the results of which are presented in table 5.6. For intensities over 500 the differences are found significant for these additional pairs tested. For lower intensities of the added concept drift, the

I_{max}	***Abest-ens40***		***ens40-i.ens30***		***i.ens30-i.ens20***		***i.ens20-i.ens10***		***i.ens10-i.ens5***		***i.ens5-i.best***	
	FPr	**FNr**	**FPr**	**FNr**	**FPr**	**FNr**	**FPr**	**FNr**	**FPr**	**FNr**	**FPr**	**FNr**
300	1	**3**	**3**	2	0	0	1	0	0	1	**3**	**3**
400	2	**3**	**3**	2	1	0	0	0	0	1	**3**	**3**
500	**3**	**3**	**3**	2	1	0	0	0	0	1	2	**3**
600	**3**	**3**	**3**	0	2	0	1	1	0	1	2	**3**
700	**3**	**3**	**3**	**3**	1	0	2	**3**	0	1	2	**3**
800	**3**	**3**	**3**	**3**	0	0	1	**3**	0	1	2	**3**
900	**3**	**3**	**3**	**3**	0	0	0	2	0	2	**3**	**3**
1000	**3**	**3**	**3**	**3**	0	0	1	1	**3**	0	2	**3**

Table 5.5: Statistical significance of different settings comparisons, separately for the FPrate and the FNrate. Significant differences are tested for, when reducing the ensemble size. Reported are the number of files for environment B (out of the three used), for which the difference in FP(FN) rate was found significant at the 5% significance level, using the KS two-sample test. The difference between a pair of settings is found significant at a specific intensity level, if either the FPrate or FNrate is found significantly different in all 3 files.

difference in performance is not always found statistically significant, as the RDA can to some extent tolerate low intensities of concept drift, so there is not guaranteed gain from using an ensemble. Additionally, the difference between an ensemble of 5, which has been considered as the best size in the previous sections, is shown to significantly improve the performance over RDA_{Abest}.

Finally, the difference in performance for different ways of selecting the ensemble members is tested for significance. The results of this are presented in table 5.7. Constructing an ensemble using implicit metrics is found to lead to significantly different performance, for intensities higher than 400, compared to using the best individuals on the training set based both on Pareto optimality and bounded optimality. These two, however, are not found to be statistically different.

5.6 Summary and discussion

In this chapter the problem of adaptation in response to concept drift is addressed; a system which detects anomalies in time series data has to adapt because of a change in the background. The application scenario is a mobile sensor platform moving to a new environment. The new environment has been simulated using the mass spectrometry data available from the DSTL ICARIS competition to which an artificial new background signature has been added. It is assumed that once the system is in a new environment, it

I_{max}	***i.ens30-i.ens10***		***i.ens20-i.ens5***		***i.ens30-i.ens5***		***i.ens10-i.best***		***i.ens5-Abest***	
	FPr	**FNr**	**FPr**	**FNr**	**FPr**	**FNr**	**FPr**	**FNr**	**FPr**	**FNr**
300	1	1	1	1	1	1	**3**	**3**	2	**3**
400	1	0	1	1	2	1	**3**	**3**	**3**	**3**
500	1	2	1	2	2	2	2	**3**	**3**	**3**
600	**3**	2	1	**3**	2	**3**	2	**3**	**3**	**3**
700	**3**	**3**	2	**3**	**3**	**3**	2	**3**	**3**	2
800	**3**	2	2	**3**	**3**	**3**	**3**	**3**	**3**	2
900	**3**	**3**	**3**	**3**	**3**	**3**	2	**3**	**3**	1
1000	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	1

Table 5.6: Statistical significance separately for the FPrate and the FNrate. Significance tests are repeated for pairs with greater differences in ensemble size. Reported are the number of files for environment B (out of the three used), for which the difference in FP(FN) rate was found significant at the 5% significance level, using the KS two-sample test. The difference between a pair of settings is found significant at a specific intensity level, if either of FPrate or FNrate is found significantly different in all 3 files.

has no new information about this environment. Adaptation is triggered by the concept drift detection module that was presented in Chapter 4.

The RDA is the anomaly detection algorithm that is used. The RDA is trained for a specific environment, using a training labelled dataset. Training in this work involves evolving the parameters of the algorithm to optimise performance on the training set. When the environment changes with the addition of a new background signature, the evolved solution is no longer viable. This was demonstrated both in Chapter 4, but also in this chapter, where a decreased performance of RDA_{Abest} in environment B was reported. This means that adaptation is needed; a new parameter set has to be found for the RDA to restore the performance to pre-drift levels.

The solution proposed is, instead of retraining/trying to find a new parameter set, to reuse a set of existing RDAs (each one having a different parameter set) and combine them in an ensemble. This is possible and comes at no extra cost, because during evolution a population of candidate parameter set is evolved, not just a single parameter set/RDA. The population that was evolved for the training set is the pool of candidates for the ensemble. Diversity and accuracy of the base learners are the two conditions for a successful ensemble. The diversity of the base learners is promoted through the multi-objective NSGAII used for the evolution of the parameters. The accuracy, however has to be estimated. This is solved by introducing implicit performance metrics. The implicit metrics (IM) vector is created

I_{max}	***i.ens10-b.ens10***		***i.ens10-p.ens***		***b.ens10-p.ens***	
	FPr	**FNr**	**FPr**	**FNr**	**FPr**	**FNr**
300	1	1	1	0	2	1
400	2	0	2	2	0	1
500	**3**	1	**3**	**3**	0	0
600	**3**	2	**3**	**3**	0	0
700	**3**	1	**3**	**3**	0	1
800	**3**	1	**3**	2	0	1
900	**3**	1	**3**	**3**	0	0
1000	**3**	**3**	**3**	**3**	0	0

Table 5.7: Statistical significance of difference between different ensemble member selection strategies, separately for the FPrate and the FNrate. Reported are the number of files for environment B (out of the three used), for which the difference in FP(FN) rate was found significant at the 5% significance level, using the KS two-sample test. The difference between a pair of settings is found significant at a specific intensity level, if either of FPrate or FNrate is found significantly different in all 3 files.

over a duration where data is collected and reflects the dynamics of the RDA in this time partition. A reference implicit vector that is created during training for the best RDA, reflects to some extent the behaviour of a good detector. When the system is in a new environment, every candidate is run through the data collected in this new environment and an implicit vector for this candidate can be calculated. Then the performance of this candidate can be assessed using the proximity of its implicit vector to the reference implicit vector; if the two are close, the candidate is likely to perform reasonably. Hence, the candidates that are most likely to perform reasonably in the new environment can be selected to participate in the *implicit* ensemble.

First the offline case is examined. All the data for environment B are assumed to be available and the implicit vector is calculated over the entire new environment - test set. In other words, the data is not processed in real-time, but at a later stage. The experiments performed show that an ensemble of the whole population does perform better than the previously known (from training) best RDA, RDA_{Abest} only for the low-medium range of the intensities tested. For higher intensities, the performance of a lot of individuals of the population decreases significantly, and so does the ensemble's performance. Using the IM vector, these individuals can be pruned from the ensemble, improving the performance. The ensemble is tested for different sizes, every time selecting the members with the highest similarity to the reference implicit vector. The performance is found to significantly

increase (especially in terms of false positive rates) as the size of the ensemble is reduced. However, it is useful to use an implicit ensemble over a single detector; a single 'best' solution (the one with the highest affinity to the implicit vector) produces significantly higher FNrates than an ensemble of 5, without significantly improving the FPrate. This means that the implicit performance vector cannot accurately locate the best candidate for the new environment, but it can be very useful in excluding unsuitable candidates. An ensemble of 5 members is found to be significantly better than the RDA_{Abest}, the previously known best solution. False positive rate is decreased by up to 0.32 points on average, and the false negative rate by up to 0.12 points. It has also been shown that this method for selecting the ensemble members is significantly better than populating the ensemble based on the performance of the solutions on the training set - predrift environment.

The main limitation of the system comes from the fact that the results are bimodal. Even though it has been demonstrated that for an ensemble of five, in approximately 80% of the cases the FPrate will be under 0.2 (compared to around 55% for RDA_{Abest}), there is some probability that the resulting FPrate and FNrate of the ensemble will be very high. This cannot be addressed at the stage of the ensemble formation, because it is attributed to the evolution. In some cases, the final population has converged in a solution space, where the RDAs evolved do not have good generalisation capabilities, therefore they perform very poorly when concept drift is added. Although this reduces the usability of the system as it is, it can be solved by revisiting the GA used, trying to steer evolution away from these areas of the solution space, or using some form of validation set to promote generalisation. This is left to future work, as the scope of this work was to examine whether using the already existing populations (evolved from the same GA used for the optimisation of the RDA), the accuracy of the system in the new environment can be improved.

Finally, the online case is addressed. This means that the formation of the ensemble happens as the data is being collected. Some data needs to be collected first, in order to start making decisions, as the implicit vector needs to be calculated over a duration where a candidate RDA is working. The performance of the ensembles, as more time passes in the new environment (i.e more data is being collected), is assessed. There is a critical amount of data that needs to be collected, in order for the implicit vector to reflect efficiently the steady state performance of the system and that is found to be 4000-5000 timesteps. The

frequency of the mass spectrometer used for this dataset is 3 HZ, so this corresponds to 22-28 minutes. From then on the ensemble of 5 members achieves a stable good performance, which on most cases improves gradually as time goes by. This means that after the first ensemble is selected at timestep 4000, then it can regularly be updated as more data is collected. Using the ensemble earlier than 4000 timesteps results in unstable performance. It is encouraging, however, that there is convergence to a good performance as early as 4000 timesteps (which is 20% of the file). Also, although 22 minutes is a substantial amount of time where the system cannot be used, if long term applications are considered it can be an acceptable downtime needed to recalibrate the system. The online use of the system could be improved if the observed pattern of better performance in the early parts of the data, where there are no anomalies, is considered. If parts of the new environment where there are no anomalies can be isolated, then a temporary ensemble could possibly be used, created based on that clean - new environment data.

The use of ensembles in principle can be supported in real-time adaptation. It is a highly parallelisable approach, as a different processor/core can be assigned to every candidate or member of the ensemble. The $\overline{p_t}$, $\overline{n_t}$ and D used in the implicit vector calculation can be updated as data is being collected. The calculation of the implicit vector and the calculation of the distance to the reference implicit vector can also be done in parallel for every candidate when an ensemble is to be created. The overhead for these calculations is very small, which allows for this method to be used in real-time. When an ensemble is formed and is being used, the decisions of each member at every timestep are combined through a simple majority vote, which also adds minimal overhead.

5.6.1 General summary of key results

The goal of this work, as stated and discussed in Chapter 1, was to develop an anomaly detection system that adapts autonomously, with no user feedback, online and in real time to the extent possible. The aim of this adaptation is to address the observed decrease in the performance of the RDA in a changing environment scenario. The key results of this chapter relating to these requirements are summarised as follows:

- The RDA, trained for a known environment (noted in this chapter as RDA_{Abest}) as assumed in the beginning of this thesis, suffers from a degradation of performance, indicated mainly by the high false positive rate, observed after the addition of artificial concept drift. This has been demonstrated in Chapter 4, but also in this

chapter, where this rate has been shown to reach median values of up to 0.9 for higher intensities of added concept drift. The goal of adaptation is to reduce these high false positive (and false negative) rates.

- The process of adaptation can be autonomously initiated, by detecting concept drift in the incoming data, as discussed in the previous chapter.

- The proposed implicit ensemble, as a method of adapting to the new environment, has the advantage of not requiring new labelled data (i.e. user input), because the RDA does not have to be retrained. Existing pre-trained, diverse RDAs, whose accuracy on the new environment is estimated using implicit metrics, are reused as base classifiers of an ensemble.

- The proposed implicit ensemble, has been shown to improve the FP/FNrate with a probability of up to 88%. The presented results indicate that after switching from a single RDA (RDA_{Abest}) to an implicit ensemble of 5 RDAs, the probability of observing a low false positive rate, lower than 0.2, increases from around 60% to 90% in low intensities and from around 35% to 70% in higher intensities of concept drift.

- Online and realtime adaptation is possible within certain limits, as a minimum amount of data from the new environment has to be collected before the performance of the individual RDAs can be estimated using the implicit metrics. With the present sampling rate, this corresponds to 22-28 minutes, but naturally this depends on the resources of an application.

Chapter 6

Conclusions and Future work

The thesis concludes with an overview of the work presented, a discussion of the main contributions and thesis novelty and suggestions for possible future research directions. The application of interest that has driven the algorithm development in this thesis is that of autonomous, robot mounted chemical detection, in dynamic changing environments. A robot that has the ability to detect anomalies while moving between different background environments, needs to be able to adapt; otherwise the change in the background environment can result in a degradation of its performance. This thesis addresses the software, algorithmic parts of such a robotic system; the hardware aspect, i.e. the robot itself or an embedded implementation is not in the scope of this thesis. Nonetheless, the proposed algorithms take into account limitations and requirements that are in principle associated with the application, namely online and near real-time processing. The requirement that has been considered of the highest importance in this thesis is the autonomy. An autonomous adaptive system needs to be able to decide *when* there is the need to adapt and *how* to adapt; how to change its mode of operation or its parameters in order to respond to the new environment.

Overview and summary of the proposed system

The system proposed in this thesis consists of three modules, as illustrated in figure 6.1. One that performs chemical detection on incoming sensor data, one that detects change and one that is responsible for adapting the system. The first module, **Anomaly Detection** treats chemical detection as an anomaly detection problem and uses the RDA [4] to detect chemicals of interest in timeseries incoming data. The immune inspired RDA is not a contribution of this work. It has been previously developed and used for chemical detec-

tion [5] and it is retained as the anomaly detection algorithm for this work as well. The other two modules, **Concept Drift Detection** and **Adaptation** have been presented in Chapters 4 and 5 respectively. The Concept Drift Detection module runs in parallel and without interfering with the operation of the RDA. By using statistical hypothesis testing on consecutive time windows enforced on the incoming sensor data, it determines if the generating data distribution has changed significantly; this is indicative of a changing environment. In this case the adaptation module is triggered. In order to adapt quickly and without additional information about the new environment, which would have to be provided by an external user or observer, existing resources are utilised. These come in the form of a population of RDAs that have been previously evolved for the old (before drift environment). Each RDA is evaluated on the new environment using implicit performance information, which is extracted from the dynamics of this specific RDA. The estimated top RDAs in the population are combined into an ensemble. Detailed summaries for every part of the system, as well as the application domain will be provided in the remainder of this chapter.

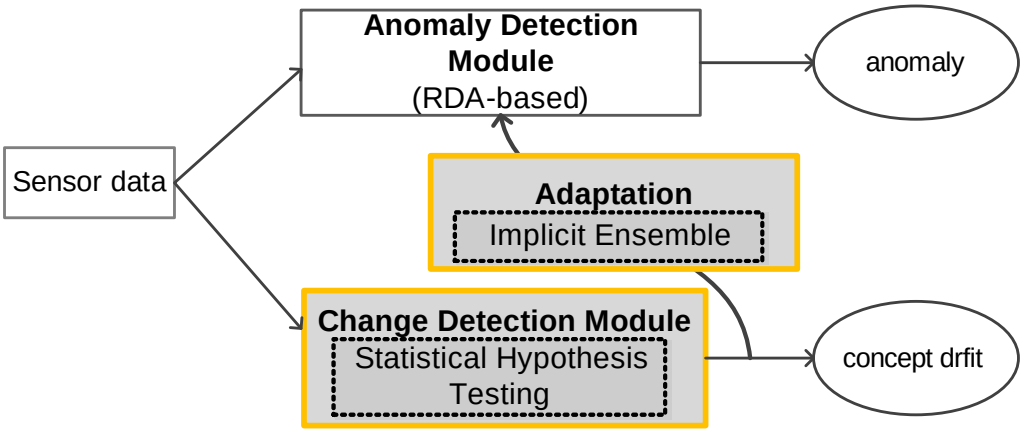


Figure 6.1: An overview of the proposed three-module adaptive chemical detection system. The proposed system employees separate modules for chemical (anomaly) detection, the detection of concept drift and the adaptation. A statistical hypothesis testing method runs in parallel with the RDA, detecting concept drift in the incoming sensor data. The detected concept drift can be reported to the end user for analysis, but its main purpose is to trigger the adaptation module. Adaptation is accomplished by switching to an ensemble RDA mode, where the members of the ensemble are selected based on their implicit performance information.

6.1 Structure of this chapter

The remainder of this chapter is structured as follows: section 6.2 includes a summary of the work presented in every chapter, along with the contributions of the chapter where appropriate. In section 6.3 some concluding observations are presented and the research question of this thesis is revisited. Finally, in section 6.4 future research directions are suggested, based on the findings of this thesis.

6.2 Summary and contributions of each chapter

The work presented in this thesis is summarised here on a chapter by chapter basis:

Chapter 1 introduces the scope and motivation for this thesis. The application scenario of robot mounted chemical detection in dynamic, changing environments is presented. The necessity of an adaptive system and the requirements associated with this application are discussed. The first and most important of these requirements is autonomous adaptation, deciding when and how to adapt, without the direction or feedback of an external user. User feedback, for instance in the form of new labelled data for re-training, cannot be assumed to be available in a deployed autonomous chemical detection system. The other two requirements, online adaptation (using data as it is being acquired) and near real-time adaptation (adapting in a timely manner) are considered secondary, because they depend to some extent on the computational resources available for any specific application. However, an effort has been made to address in principle these challenges too.

Chapter 2 presents a review of the domain of chemical agent detection, identifying key challenges associated with robot based, real-time, continuous chemical detection. The main trends in sensing technology and chemical identification, for a variety of possible sensors and analytical instruments (spectrometers), have been discussed. It has been argued that viewing the problem of chemical agent detection primarily as an anomaly detection problem can be advantageous: in dynamic unknown environments it is often more useful to detect chemicals that deviate from what is expected (anomalies), rather than be limited to recognising a finite number of chemical agents on which a classification algorithm is trained. In this context, the RDA, an immune inspired algorithm that can detect anomalies embedded in time series data, is intro-

duced. Its advantages, which make it suitable for use in this work, include its proven high accuracy and low false positive rate [5], and the fact that it is fast, lightweight and operates in an online manner.

Chapter 3 discusses the problem of concept drift and adaptation. Concept drift is the phenomenon of change either in the target concepts or in the processes that generate the data of the application. In autonomous real-time chemical detection, concept drift can occur because of changes in the environmental conditions, moving to different locations, or sensor drift. In every case, a system trained on data drawn from fixed distributions, experiences a degradation in performance when these distributions change, and needs to adapt. A review is presented on handling concept drift either by updating continuously, or in reaction to some trigger that indicates concept drift. The paradigm of ensembles in the context of adaptation is also discussed along with a review of relevant work. This chapter has indicated that concept drift can be detected without requiring labelled data from new concepts, by monitoring the input data. However, adapting a classifier or an ensemble in response, typically requires some form of labelled data, for model evaluation in the new concept. Adaptive chemical detection often also suffers from the limitation of attempting to build accurate models of the normal space, or the sensor/class relationships, which can be costly to continuously update and unsuitable for high dimensional data (e.g. spectrometry data).

Chapter 4 addresses the problem of detecting changes in the background environment signal (concept drift), by monitoring the incoming multivariate time series data. Statistical hypothesis testing is used to this end. Time windows are enforced on the incoming data and a recent window is compared to a reference window using an appropriate statistical test; if a statistically significant difference is found between the two, then concept drift is detected. In this chapter, five different statistical tests are compared on their ability to detect concept drift that has been artificially added to mass spectrometry data. The five tests are: the Mann−Whitney, the Smirnov and the Wald−Walfowitz tests (all three adapted to the higher dimension using minimal spanning tree approach), and two estimates of the specifically designed for multi−dimensional data MMD method. The experiments show that there is no universal winner: the MMD_b detects drift accurately but it is slow to compute, and

the Wald-Wolfowitz test is fast to compute but not as accurate. However, when tested on data with anomalies and added concept drift, they are found able to differentiate between the two.

Contribution of Chapter 4: The contribution of this chapter is twofold. The first contribution is the application and comparison of statistical hypothesis testing methods in chemical data, and in particular, high dimensional mass spectrometry data. This chapter has shown that artificial concept drift, which simulates a changing environment, can be detected by the tested methods, while relevant limitations are analysed. The statistical methods have also been applied to mass spectrometry data that contain anomalies as well as concept drift. The second contribution of the chapter is establishing the suitability and limitations of statistical hypothesis testing, as a module that performs concept drift detection in parallel with the RDA, which detects the anomalies on the same data.

Chapter 5 addresses the adaptation of the RDA, which is triggered by the detection of concept drift. In the work presented in this chapter, ensembles are used in order to adapt to a new unknown environment, by utilising existing anomaly detectors (RDAs) evolved for the previous, known environment. In particular, after concept drift is detected (environment changed), the suitability of the candidate RDAs is estimated on data collected from the new environment, based on implicit performance information. The latter has been defined as a set of metrics that can be calculated from the dynamics of the RDA, not requiring explicit knowledge of the model's performance. A method to extract such information from the RDA has been proposed, using the values of the position and negative feedback, as well as information about the detections made. Based on that, the proposed implicit ensemble is populated by the candidates whose implicit performance metrics are the most similar to the corresponding metrics of a known reference RDA. The implicit ensemble has been tested on the artificial concept drift enhanced mass spectrometry data. The main finding is that even though the results suffer from high variability, an implicit ensemble of 5 improves significantly the after-drift FPrate and FNrate of the previously known single best RDA. Moreover, it has been demonstrated that the implicit ensemble converges online, after a minimum necessary data-collection period has elapsed.

Contribution of Chapter 5: Chapter 5 makes a number of contributions. The first is the extension of the RDA, as the base learner of an ensemble. The notion

of exploiting the diversity of a set of RDAs that have been evolved for a previous environment, combining them into an ensemble for the new environment, has been investigated. The second contribution is the proposal and extraction of an implicit performance vector that can estimate the performance of a given parameter set for the RDA when used in an unknown environment, without access to any training data for this new environment. The third and final contribution is the development of a novel mechanism to select the members of the ensemble from a diverse pool of candidates, using this implicit performance vector, and the investigation of the application of this mechanism both in the context of offline and online adaptation.

6.3 Concluding remarks

The work presented in this thesis has shown the feasibility of a chemical detection system that can adapt autonomously. Using statistical hypothesis testing in parallel with the RDA on the same incoming data ensures that the detection of concept drift does not rely nor interfere with the main operation of the system, which is the detection of anomalies embedded in the incoming data. The Wald-Wolfowitz and the MMD_b have been found the best in detecting the portions of the data that correspond to a transition between different environments, either in terms of speed or accuracy. The selection of one of the two is left to the final user, as specific application resources and constraints have to be taken into account. However, the fact that both of these methods detect only concept drift and not anomalies is an important finding. This allows any of the two methods to be used in order to trigger adaptation in response to concept drift and not some form of irregularity in the data.

The adaptation has been addressed by reusing old resources and previously acquired knowledge that is mostly outdated in the new environment. This approach is adopted because the assumption of obtaining new labelled data, in order to rebuild the RDA, is considered in this work at best weak, especially in the short term after drift. An ensemble is used in order to exploit the existence of a population of diverse RDA candidates that have been evolved for an old concept. The necessary diversity of the base RDAs of the ensemble is ensured by the EA and the accuracy through the selection of the members based on their implicitly estimated performance. An ensemble that is constructed in this way has been shown to improve over a single outdated RDA. Even though estimating

the performance of a candidate based on implicit metrics cannot locate the best possible candidate for the new environment, it has been shown that pruning the ensemble base on this implicit information can lead to significantly increased performance.

The online and real-time implication of the developed systems have been addressed at a few points throughout this work. The main observations are these: Concept drift can be detected by examining the last two windows of data. While this approach processes the data in batches, rather than on a single datapoint basis, it can still be considered online because a decision is made, without the need to process the entirety of the dataset. The implicit ensemble adaptation is suitable for online use, as the metrics can be calculated in an online fashion; however the improvement begins only after a certain period has passed. Real-time considerations have been addressed where possible: this requirement has been taken into account both in the discussion about the window size for the concept drift detection, and in the discussion about the computational overhead of calculating the implicit vector and the possibility of parallel implementation.

6.3.1 Limitations

The main limitations of the work presented in this thesis are:

- The methods have been tested on **artificial concept drift**. This has the advantage of controlling the artificial drift events and evaluating the methods more accurately. On the other hand, simulating concept drift can be different from the reality. However, constructing a dataset with concept drift could not have been accomplished in the scope of this thesis, and to the knowledge of the author, no such dataset, with both concept drift and anomalies, especially with high dimensional data, was publicly available at the time of developing this work.

- For the addition of concept drift in the data only **linear addition** of an artificial event has been considered. There is nothing particular to detecting linear transitions on the solution proposed, so it can be assumed that the findings will generalise to other types of transitions as well.

- The Wald-Wolfowitz method has been found to benefit from **tuning the concept drift detection threshold**. However, for this tuning exemplar data with concept drift have to be available beforehand.

- The main limitation of the implicit ensemble method is the **high variability** in the results, which has been found to be an artefact of the evolutionary algorithm. As indicated in the relevant chapter, an analysis of the evolutionary process is needed in order to address this problem.

- Finally, it has been determined that the implicit ensemble can be used online and that its performance converges to the offline one. The time needed for convergence is determined to be 22-28 minutes. This time-frame can be unsuitable for applications that cannot tolerate such downtime. However, the results indicate that there is some room for improvement, if anomaly free portions of the new data can be isolated.

6.3.2 Revisiting the research question

The research question around which this thesis was built was formulated in Chapter 1 as:

To what extent can a system that autonomously detects a change in the environment and adapts in response to that detection, address the decrease of the performance of an RDA-based chemical detection system, when it is deployed in changing environments?

Taking into account the contributions of the thesis and the relevant discussion in the previous sections, it is the author's opinion that the developed system achieves to autonomously adapt when required and to address the degradation in the performance of the RDA in a new unknown environment, within the limits that have been outlined.

6.4 Future work

The work presented in this thesis can lead to a number of interesting future work directions, the main of which are outlined in this section:

F1. Robot implementation. A robot implementation was not in the scope of this thesis. However, the next step in order to establish the feasibility of the system for a real world application is to test it on a real robotic platform. A challenge for this would be recreating changing environments for the robot to navigate in a controlled manner.

F2. Extend the implicit metrics strategy to other methods like ANNs and SOMs and possibly to other domains. It would be interesting to disengage the implicit ensemble

approach from chemical detection and investigate its usability as a general adaptation framework.

F3. Investigate the evolutionary process in order to decrease the variability. In this work, the NSGAII has been adopted, as it has been used before for this specific task and it provided a way to reuse existing resources with no added cost. However, there could be value in investigating other EAs as well, and possibly using evolution to explicitly generate good ensembles.

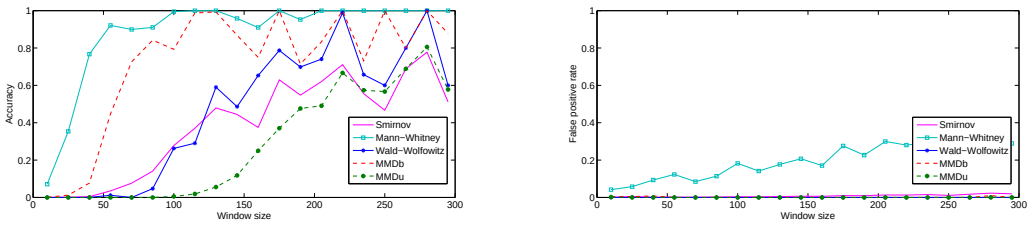
F4. Reaction to very slow drifting behaviour. This work has assumed that the transition period can be ignored as it consists of intermediate concepts that cannot be learnt. However, in cases of very slow drift, the transition can last for a substantial amount of time and adaptation will be needed during the transition.

Appendix A

Additional Results on Concept Drift Detection

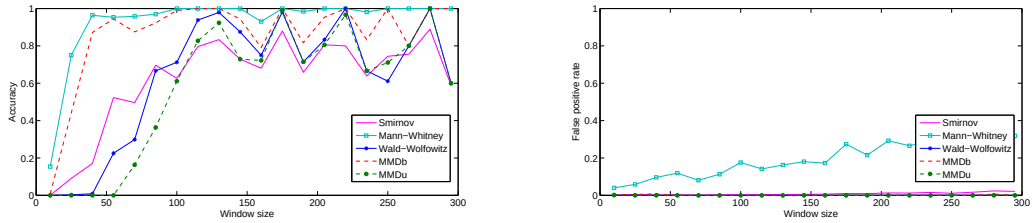
A.1 Varying the window size - Additional intensities

In this appendix some additional results are presented on using statistical hypothesis testing for detecting concept drift. FiguresA.1a-A.4 depict the resulting accuracy and FP rate for all the five methods when the window size is varied. In section 4.5.1, in figure 4.16 a comparison is presented of the five methods for varying window size and for concept drift intensity $I_{max} = 700$. Here we present the equivalent figures for all the intensities tested: $I_{max} = [250, 500, 700, 900]$. The duration of the artificial event is 1000.



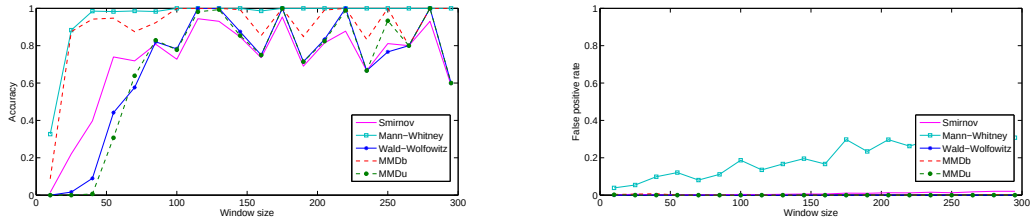
(a) (b)

Figure A.1: Accuracy (a) and false positives rate (b) of all methods over the window size, for $I = 250$.



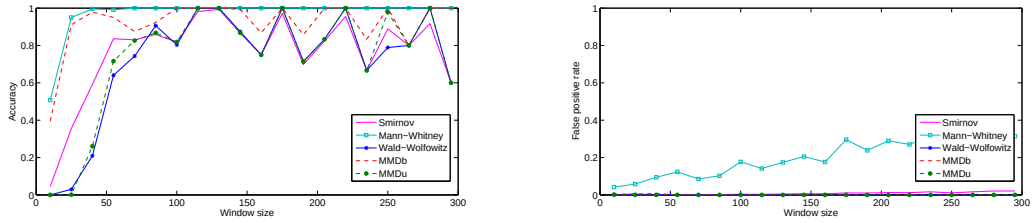
(a) (b)

Figure A.2: Accuracy (a) and false positives rate (b) of all methods over the window size, for $I = 500$.



(a) (b)

Figure A.3: Accuracy (a) and false positives rate (b) of all methods over the window size, for $I = 700$.

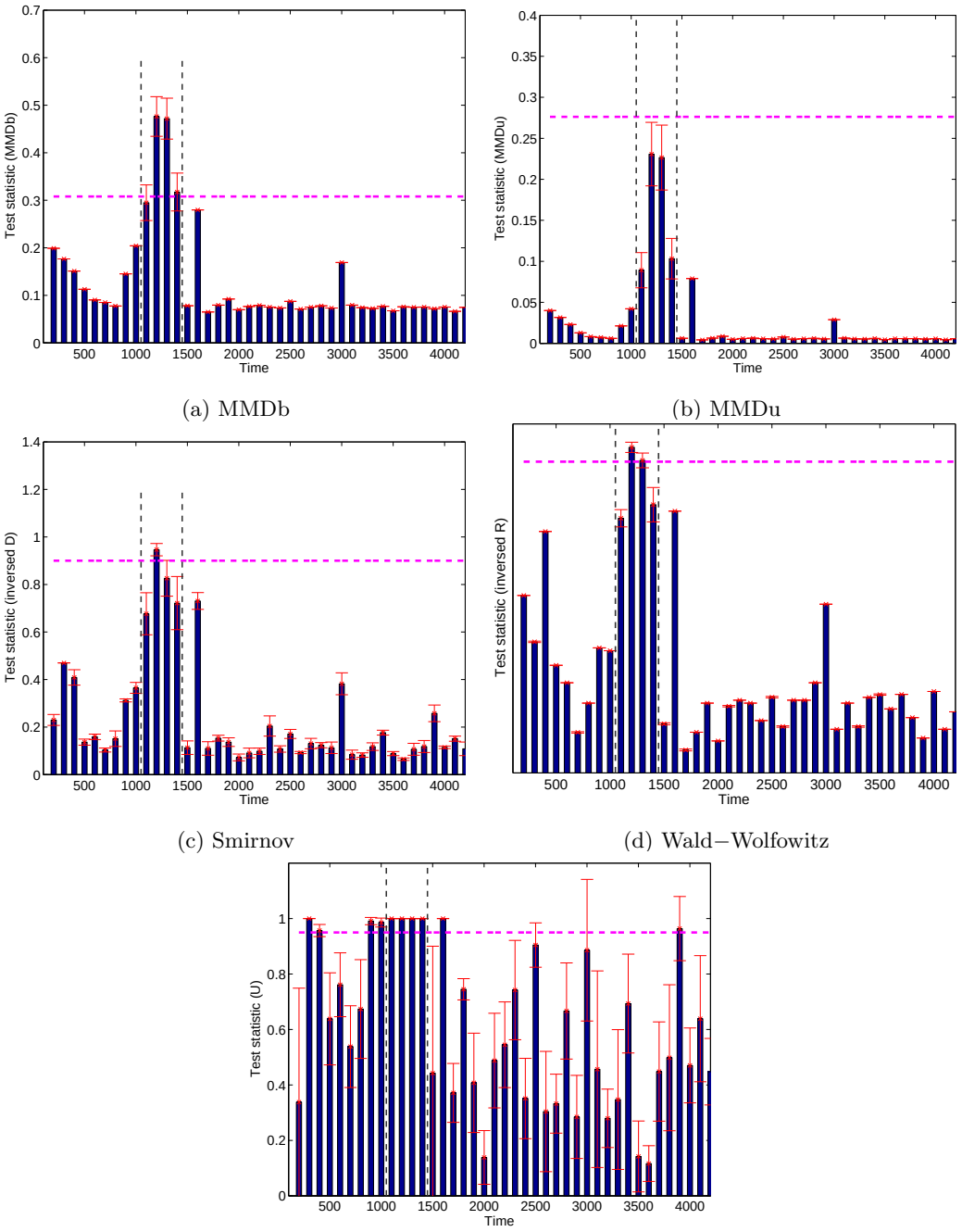


(a) (b)

Figure A.4: Accuracy (a) and false positives rate (b) of all methods over the window size, for $I = 900$.

A.2 Detection of concept drift for all the methods - Additional Intensities

The figure 4.18 presented in 4.5.1, is an overview figure of all the methods, that shows the detection for every window (average over the 18 runs) and the standard deviation. In the main text the results reported correspond to intensity $I_{max} = 400$. Here the additional intensities presented are $I_{max} = [100, 200, 300, 500, 600]$. A different set of intensities is tested, because the duration of the event for these experiments is set to $t_e - t_s = 300$. These figures are not reported in the main test because they do not add significantly to the results observed by other presented figures.



(a) MMDb

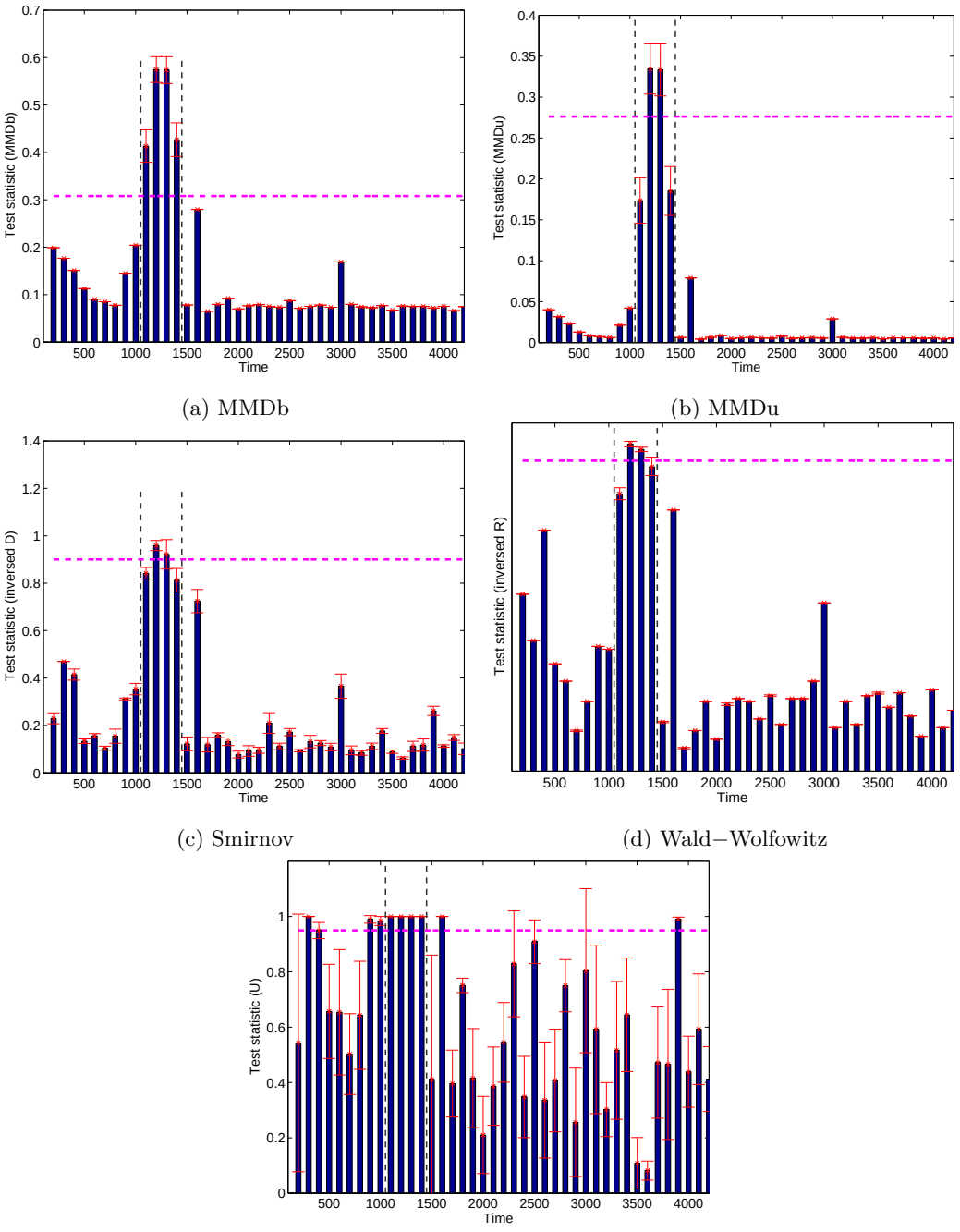
(b) MMDu

(c) Smirnov

(d) Wald−Wolfowitz

(e) Mann−Whitney

Figure A.5: Detection of concept drift for all methods. The event, of intensity $I_{max} = 100$, in this example is introduced at time step 1000 and lasts for 300 time steps, as indicated by the vertical grey lines. The horizontal line is the detection threshold for each method. The bars correspond to the time windows and each bar represents the value of the test statistic as calculated at the end of the window. When the test statistic exceeds the threshold, concept drift is detected for this window. Note that for the Smirnov and Wald−Wolfowitz methods, the results have been inverted for illustration purposes (for these methods drift is detected for low values of the test statistic).



(a) MMDb

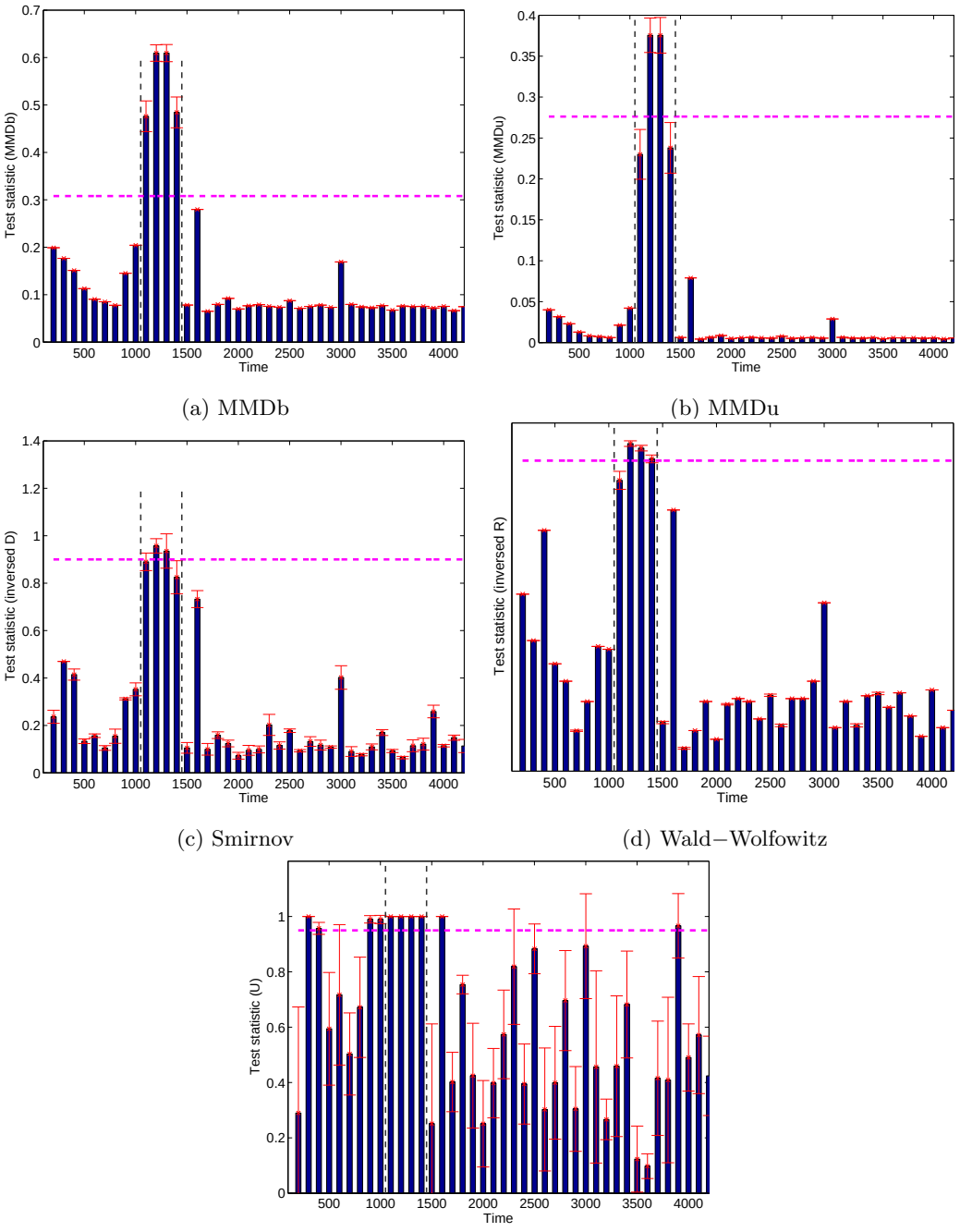
(b) MMDu

(c) Smirnov

(d) Wald−Wolfowitz

(e) Mann−Whitney

Figure A.6: Detection of concept drift for all methods. The event, of intensity $I_{max} = 200$, in this example is introduced at time step 1000 and lasts for 300 time steps, as indicated by the vertical grey lines. The horizontal line is the detection threshold for each method. The bars correspond to the time windows and each bar represents the value of the test statistic as calculated at the end of the window. When the test statistic exceeds the threshold, concept drift is detected for this window. Note that for the Smirnov and Wald−Wolfowitz methods, the results have been inverted for illustration purposes (for these methods drift is detected for low values of the test statistic).



(a) MMDb

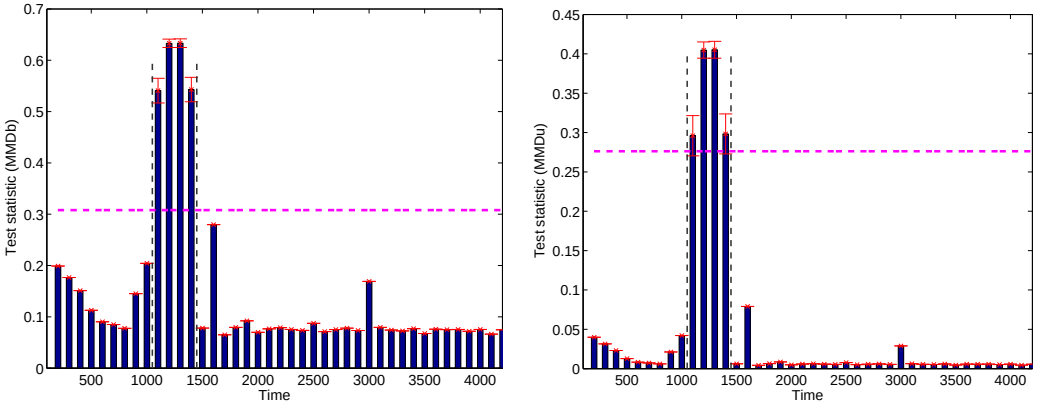
(b) MMDu

(c) Smirnov

(d) Wald−Wolfowitz

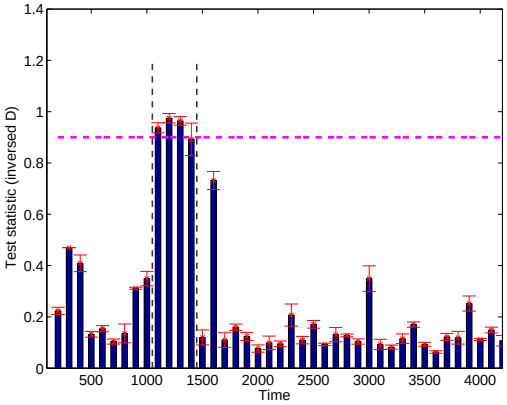
(e) Mann−Whitney

Figure A.7: Detection of concept drift for all methods. The event, of intensity $I_{max} = 300$, in this example is introduced at time step 1000 and lasts for 300 time steps, as indicated by the vertical grey lines. The horizontal line is the detection threshold for each method. The bars correspond to the time windows and each bar represents the value of the test statistic as calculated at the end of the window. When the test statistic exceeds the threshold, concept drift is detected for this window. Note that for the Smirnov and Wald−Wolfowitz methods, the results have been inverted for illustration purposes (for these methods drift is detected for low values of the test statistic).

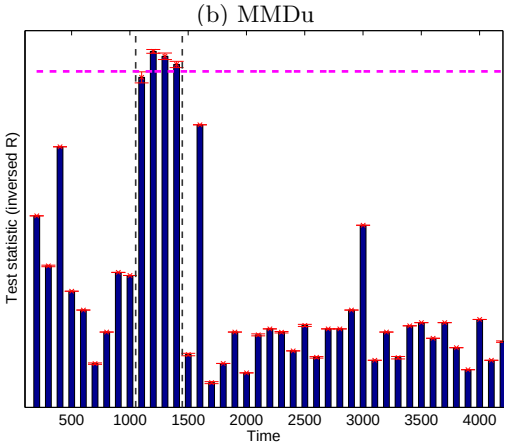


(a) MMDb

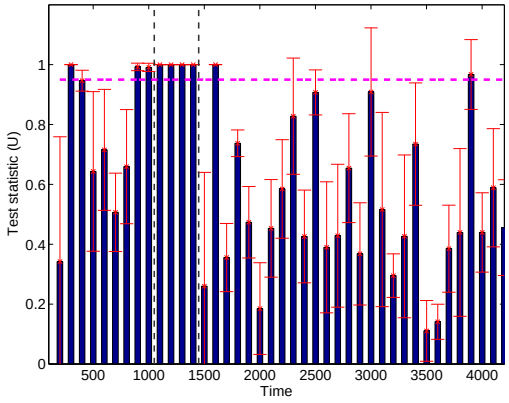
(b) MMDu



(c) Smirnov

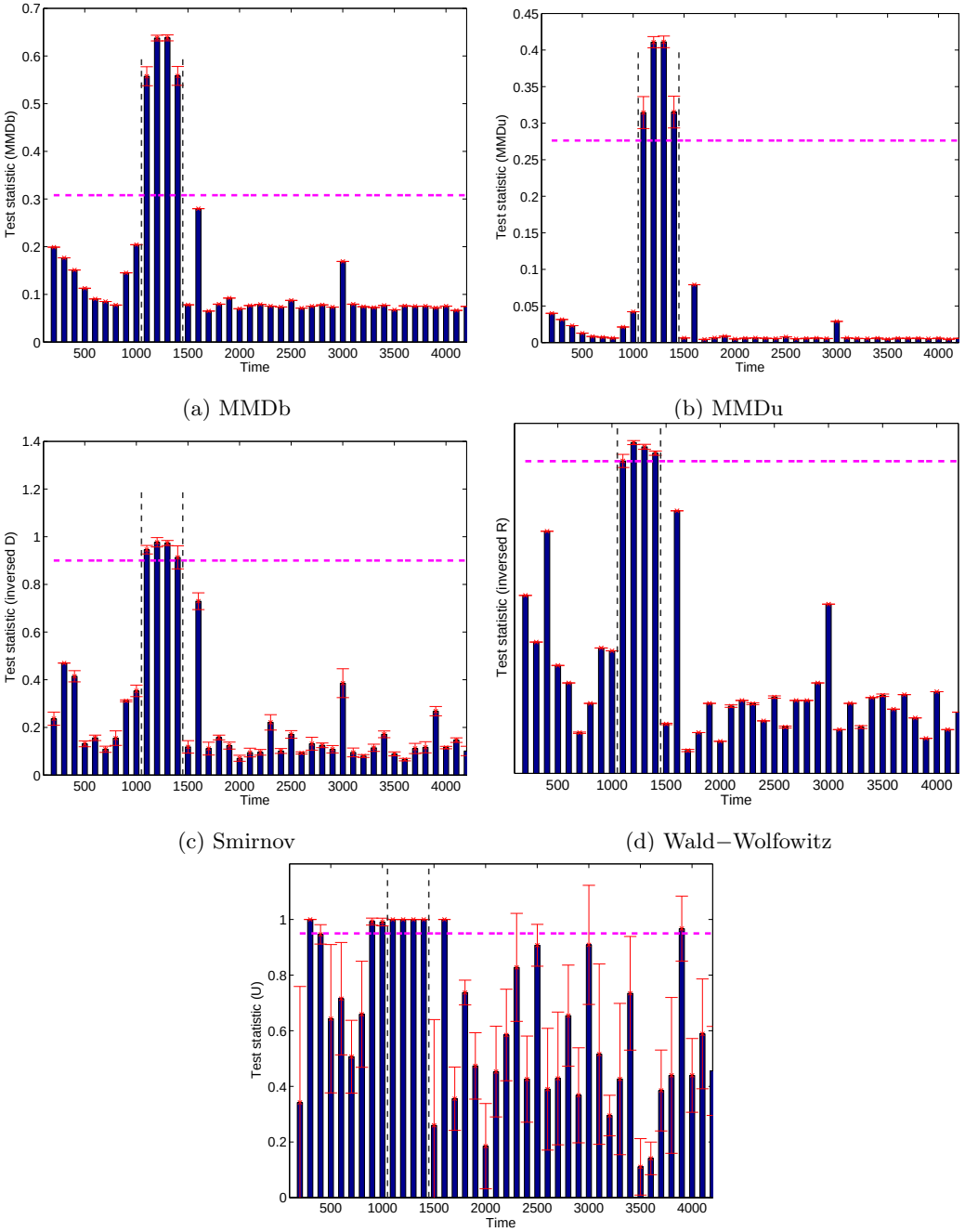


(d) Wald−Wolfowitz



(e) Mann−Whitney

Figure A.8: Detection of concept drift for all methods. The event, of intensity $I_{max} = 500$, in this example is introduced at time step 1000 and lasts for 300 time steps, as indicated by the vertical grey lines. The horizontal line is the detection threshold for each method. The bars correspond to the time windows and each bar represents the value of the test statistic as calculated at the end of the window. When the test statistic exceeds the threshold, concept drift is detected for this window. Note that for the Smirnov and Wald−Wolfowitz methods, the results have been inverted for illustration purposes (for these methods drift is detected for low values of the test statistic).



(a) MMDb

(b) MMDu

(c) Smirnov

(d) Wald−Wolfowitz

(e) Mann−Whitney

Figure A.9: Detection of concept drift for all methods. The event, of intensity $I_{max} = 600$, in this example is introduced at time step 1000 and lasts for 300 time steps, as indicated by the vertical grey lines. The horizontal line is the detection threshold for each method. The bars correspond to the time windows and each bar represents the value of the test statistic as calculated at the end of the window. When the test statistic exceeds the threshold, concept drift is detected for this window. Note that for the Smirnov and Wald−Wolfowitz methods, the results have been inverted for illustration purposes (for these methods drift is detected for low values of the test statistic).

Appendix B

Additional tables for ensembles

In order to analyse the results without he effect of the bimodality, tables with the percentage of resulting FPrates (from a certain ensemble setting) under 0.2 are presented. In this appendix, additional tables for the percentage of FPrates under 0.1 and 0.3 are presented.

B	I_{max}	T (timestep at which implicit metrics are calculated)														
		50	100	150	200	300	400	500	1000	2000	3000	4000	5000	10000	15000	20000
1	500	10.72	9.15	8.37	8.63	8.82	10.65	4.58	2.48	*0.72*	4.31	8.30	9.02	17.91	20.00	**20.65**
1	700	8.63	10.39	9.35	8.50	8.95	7.91	1.37	0.98	*0.33*	2.61	8.37	9.35	15.69	15.82	**16.41**
1	900	6.14	9.35	8.24	7.84	8.63	7.32	1.11	1.11	*0.39*	2.55	8.43	9.15	12.94	12.88	**13.20**
4	500	69.41	66.67	*62.68*	63.27	63.01	62.81	64.05	68.30	93.07	90.20	93.92	93.79	95.36	96.73	**97.25**
4	700	53.14	48.76	46.01	*45.88*	48.56	50.59	51.05	54.58	78.63	82.68	87.25	86.99	89.08	89.41	**89.54**
4	900	*35.10*	36.14	36.99	38.04	36.93	37.52	38.56	38.89	64.97	64.64	74.05	73.66	76.27	76.67	**77.32**
6	500	73.01	70.59	68.30	*65.82*	67.19	74.97	75.82	86.34	88.69	89.15	86.93	88.24	90.07	90.39	**90.59**
6	700	48.50	51.31	49.08	*42.42*	43.66	52.48	61.31	75.69	80.72	81.31	78.89	81.31	83.01	82.88	**83.53**
6	900	*31.70*	39.02	39.54	32.42	36.08	39.93	42.42	55.42	62.94	65.69	60.72	63.73	64.97	65.49	**66.47**

Table B.5: Comparison of ensembles of 5 members created at different timepoints, under the online ensemble paradigm. Percentage of resulting FPrates lower than 0.3. The timestep, at which the maximum FPrate is observed, is noted in bold and the minimum in italics. For file 1, a very low percentage of FPrates under 0.1 is observed. For files 4 and 6, the FPrate percentage starts converging from timestep 2000-4000.

B	I_{max}	T (timestep at which implicit metrics are calculated)														
		50	100	150	200	300	400	500	1000	2000	3000	4000	5000	10000	15000	20000
1	500	69.54	66.21	61.83	62.68	62.55	66.54	42.75	36.41	*35.10*	65.16	85.42	88.17	**95.75**	95.56	95.16
1	700	54.64	49.28	47.71	48.30	50.46	40.39	21.31	20.92	*19.93*	46.73	70.59	73.53	**86.14**	84.58	82.81
1	900	36.99	36.60	37.32	38.37	39.02	31.90	16.60	18.17	*15.62*	34.05	56.14	59.15	**71.05**	67.97	65.69
4	500	82.09	77.97	78.24	*77.71*	79.22	80.26	80.65	83.40	97.32	92.68	95.23	95.36	95.82	95.82	**96.14**
4	700	63.73	61.24	*58.50*	59.48	62.42	64.97	66.93	69.74	89.35	83.79	87.84	88.17	89.02	89.41	**90.13**
4	900	*42.22*	46.27	48.10	51.63	53.73	54.77	55.56	56.80	78.63	74.12	79.74	79.08	79.93	**80.72**	80.72
6	500	79.22	*77.52*	77.52	80.20	83.20	86.60	86.99	91.18	91.50	91.70	89.80	91.24	**92.22**	91.96	92.16
6	700	58.63	58.82	57.12	*53.01*	58.82	69.15	72.42	78.56	81.90	81.63	78.95	81.63	82.42	83.14	**84.12**
6	900	*41.50*	44.31	44.71	46.47	51.11	51.96	58.43	67.32	70.72	69.54	64.44	67.58	67.65	68.10	**69.74**

Table B.6: Comparison of ensembles of 5 members created at different timepoints, under the online ensemble paradigm. Percentage of resulting FPrates lower than 0.3. The timestep, at which the maximum FPrate is observed, is noted in bold and the minimum in italics. The finding are similar to the ones reported on Chapter 5, for an FPrate of 0.2.

B	I_{max}	***Abest***	***ens40***	***i.ens30***	***i.ens20***	***i.ens10***	***i.ens5***	***i.best***	*Bbest*
1	500	**26.99**	4.44	8.30	11.57	16.80	20.65	**26.99**	39.61
1	700	21.63	1.90	3.73	6.60	12.61	16.41	**22.09**	40.39
1	900	15.03	0.92	2.22	3.33	8.04	13.20	**20.13**	41.57
4	500	60.72	64.71	86.73	93.79	95.56	**97.25**	81.24	95.88
4	700	47.65	28.30	52.81	69.28	82.88	**89.54**	76.14	95.29
4	900	34.12	11.11	26.08	40.59	60.39	**77.32**	70.46	93.40
6	500	61.90	72.88	92.81	**94.58**	93.07	90.59	64.05	98.37
6	700	49.15	34.58	62.03	74.05	80.33	**83.53**	59.02	98.56
6	900	36.01	15.49	32.81	45.56	56.01	**66.47**	50.07	97.91

Table B.1: Percentage of repetitions that achieve an FPrate lower than 0.1, under different ensemble settings. Noted in bold is the winning setting, with the highest percentage in the 0-0.2 FPrate range. For file 1, none of the setting achieves a substantial amount of FPrates under 0.1. However, for files 4 and 6, *i.ens*5 has the highest percentage in 5/6 cases.

B	I_{max}	***Abest***	***ens40***	***i.ens30***	***i.ens20***	***i.ens10***	***i.ens5***	***i.best***	*Bbest*
1	500	60.52	68.37	89.15	91.37	92.94	**95.16**	90.33	92.61
1	700	47.84	32.48	58.30	65.69	74.71	82.81	**84.71**	89.61
1	900	34.71	14.25	31.44	34.90	50.46	65.69	**80.00**	84.44
4	500	65.88	81.76	95.23	97.65	96.99	**98.43**	87.39	98.82
4	700	51.76	42.88	72.29	81.63	87.32	**93.20**	81.31	98.17
4	900	39.54	19.22	39.74	53.20	68.63	**82.03**	73.46	96.01
6	500	66.86	85.16	96.41	**98.37**	96.34	94.51	72.35	99.74
6	700	53.07	45.49	74.97	83.20	85.82	**87.39**	63.99	99.80
6	900	41.37	21.18	42.09	54.90	63.92	**72.61**	52.55	98.69

Table B.2: Percentage of repetitions that achieve an FPrate lower than 0.3, under different ensemble settings. Noted in bold is the winning setting, with the highest percentage in the 0-0.2 FPrate range. The implicit ensemble of 5 has the highest percentage in 6/9 cases.

B	I_{max}	***Abest***	***ens40***	***p.ens***	***b.ens10***	***i.ens10***
1	500	**26.99**	4.44	16.14	12.42	16.80
1	700	**21.63**	1.90	7.25	4.38	12.61
1	900	**15.03**	0.92	2.94	1.96	8.04
4	500	60.72	64.71	74.90	76.99	**95.56**
4	700	47.65	28.30	36.80	35.23	**82.88**
4	900	34.12	11.11	15.95	12.81	**60.39**
6	500	61.90	72.88	78.69	83.33	**93.07**
6	700	49.15	34.58	40.33	40.78	**80.33**
6	900	36.01	15.49	18.69	15.42	**56.01**

Table B.3: Percentage of repetitions that achieve an FPrate lower than 0.1, under different selection strategies of the members of the ensemble. Noted in bold is the winning setting, with the highest percentage in the 0-0.2 FPrate range. Again for file 1, none of the setting achieves a substantial amount of FPrates under 0.1. However, for files 4 and 6, $i.ens10$ has the highest percentage in all intensities.

B	I_{max}	***Abest***	***ens40***	***p.ens***	***b.ens10***	***i.ens10***
1	500	60.52	68.37	73.27	81.05	**92.94**
1	700	47.84	32.48	38.43	39.48	**74.71**
1	900	34.71	14.25	17.32	14.90	**50.46**
4	500	65.88	81.76	85.82	89.35	**96.99**
4	700	51.76	42.88	48.17	50.92	**87.32**
4	900	39.54	19.22	23.20	19.93	**68.63**
6	500	66.86	85.16	88.50	91.31	**96.34**
6	700	53.07	45.49	50.85	55.82	**85.82**
6	900	41.37	21.18	24.71	21.57	**63.92**

Table B.4: Percentage of repetitions that achieve an FPrate lower than 0.3, under different selection strategies of the members of the ensemble. Noted in bold is the winning setting, with the highest percentage in the 0-0.2 FPrate range. For all the files an intensities the implicit ensemble of 10 achieves the highest percentage of FPrates under 0.3.

Abbreviations

ADWIN	ADaptive WINdowing
AIS	Artificial Immune Systems
ANNs	Artificial Neural Networks
AWE	Accuracy Weighted Ensemble
BP-ANN	Back Propagation ANNs
DDM	Drift Detection Method
DWM	Dynamic Weighted Majority
EA	Evolutionary Algorithm
EDDM	Early Drift Detection Method
FLORA	FLOating Rough Approximation
FN	False Negative
FP	False Positive
GA	Genetic Algorithm
GC	Gas Chromatography
IM	Implicit Metrics
IMS	Ion Mobility Spectrometry
KS	Kolmogorov-Smirnov

MLP Multi Layer Perceptron

MMD Maximum Mean Discrepancy

MOGA Multi-Objective Genetic Algorithm

MOX Metal Oxide Semiconductors

MST Minimal Spanning Tree

NCL Negative Correlation Learning

NN Nearest Neighbor

NSGA Non-dominated Sorting Genetic Algorithm

PCA Principle Components Analysis

RDA Receptor Density Algorithm

SIMCA Soft Independent Modelling of Class Analogies

SOM Self-Organising Map

SVM Support Vector Machines

TN True Negative

TP True Positive

VOCs Volatile Organic Compounds

References

[1] A. Ladi, J. Timmis, A. Tyrrell, and R. G. Smith, "An automated sniffer dog: Real-time, adaptive molecular signature detection," *Sponsoring Institutions*, p. 64, 2012.

[2] A. Ladi, J. Timmis, A. M. Tyrrell, and P. J. Hickey, "Statistical hypothesis testing for chemical detection in changing environments," in *Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), 2014 IEEE Symposium on*, pp. 77–84, IEEE, 2014.

[3] P. E. Keller, "Electronic noses and their applications," in *Northcon 95. I EEE Technical Applications Conference and Workshops Northcon95*, p. 116, IEEE, 1995.

[4] N. D. Owens, A. Greensted, J. Timmis, and A. Tyrrell, "The receptor density algorithm," *Theoretical Computer Science*, vol. 481, pp. 51–73, 2013.

[5] J. A. Hilder, N. D. Owens, M. J. Neal, P. J. Hickey, S. N. Cairns, D. P. Kilgour, J. Timmis, and A. M. Tyrrell, "Chemical detection using the receptor density algorithm," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 42, no. 6, pp. 1730–1741, 2012.

[6] M. Holmberg, F. Winquist, I. Lundström, F. Davide, C. DiNatale, and A. D'Amico, "Drift counteraction for an electronic nose," *Sensors and Actuators B: Chemical*, vol. 36, no. 1, pp. 528–535, 1996.

[7] R. M. Karp, "On-line algorithms versus off-line algorithms: How much is it worth to know the future?," in *IFIP Congress (1)*, vol. 12, pp. 416–429, 1992.

[8] K. G. Shin and P. Ramanathan, "Real-time computing: A new discipline of computer science and engineering," *Proceedings of the IEEE*, vol. 82, no. 1, pp. 6–24, 1994.

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

[10] T. Dietterich, "Ensemble methods in machine learning," *Multiple classifier systems*, pp. 1–15, 2000.

[11] X. Liu, S. Cheng, H. Liu, S. Hu, D. Zhang, and H. Ning, "A survey on gas sensing technology," *Sensors*, vol. 12, no. 7, pp. 9635–9665, 2012.

[12] C. Ho, M. Itamura, M. Kelley, and R. Hughes, "Review of chemical sensors for in-situ monitoring of volatile contaminants," *Sandia Rep. No. SAND2001*, vol. 643, 2001.

[13] G. Murray and G. Southard, "Sensors for chemical weapons detection," *Instrumentation & Measurement Magazine, IEEE*, vol. 5, no. 4, pp. 12–21, 2002.

[14] D. Wilson, S. Hoyt, J. Janata, K. Booksh, and L. Obando, "Chemical sensors for portable, handheld field instruments," *Sensors Journal, IEEE*, vol. 1, no. 4, pp. 256–274, 2001.

[15] S. Singh, "Sensorsan effective approach for the detection of explosives," *Journal of Hazardous Materials*, vol. 144, no. 1, pp. 15–28, 2007.

[16] A. J. Lilienthal, A. Loutfi, and T. Duckett, "Airborne chemical sensing with mobile robots," *Sensors*, vol. 6, no. 11, pp. 1616–1678, 2006.

[17] R. E. Shaffer, S. L. Rose-Pehrsson, and R. A. McGill, "A comparison study of chemical sensor array pattern recognition algorithms," *Analytica Chimica Acta*, vol. 384, no. 3, pp. 305–317, 1999.

[18] L. Vogt, T. Gröger, and R. Zimmermann, "Automated compound classification for ambient aerosol sample separations using comprehensive two-dimensional gas chromatography–time-of-flight mass spectrometry," *Journal of Chromatography A*, vol. 1150, no. 1, pp. 2–12, 2007.

[19] K. Varmuza, F. Stancl, H. Lohninger, and W. Werther, "Automatic recognition of substance classes from data obtained by gas chromatography/mass spectrometry," *Laboratory Automation & Information Management*, vol. 31, no. 3, pp. 225–230, 1996.

[20] C. Gardner, R. Wentworth, P. Treado, P. Batavia, and G. Gilbert, "Remote chemical biological and explosive agent detection using a robot-based raman detector," in

SPIE Defense and Security Symposium, pp. 69620T–69620T, International Society for Optics and Photonics, 2008.

[21] M. Praisler, I. Dirinck, J. Van Bocxlaer, A. De Leenheer, and D. Massart, "Pattern recognition techniques screening for drugs of abuse with gas chromatography–fourier transform infrared spectroscopy," *Talanta*, vol. 53, no. 1, pp. 177–193, 2000.

[22] M. Westhoff, P. Litterst, L. Freitag, W. Urfer, S. Bader, and J. I. Baumbach, "Ion mobility spectrometry for the detection of volatile organic compounds in exhaled breath of patients with lung cancer: results of a pilot study," *Thorax*, vol. 64, no. 9, pp. 744–748, 2009.

[23] C. Bishop *et al.*, *Neural networks for pattern recognition*. Clarendon press Oxford, 1995.

[24] L. De Castro and J. Timmis, *Artificial immune systems: a new computational intelligence approach*. Springer, 2002.

[25] S. Bell, E. Nazarov, Y. Wang, and G. Eiceman, "Classification of ion mobility spectra by functional groups using neural networks," *Analytica chimica acta*, vol. 394, no. 2, pp. 121–133, 1999.

[26] W. Nunes, A. Da Silva, V. Crispim, and R. Schirru, "Explosives detection using prompt-gamma neutron activation and neural networks," *Applied radiation and isotopes*, vol. 56, no. 6, pp. 937–943, 2002.

[27] D. Dasgupta, Y. Cao, C. Yang, *et al.*, "An immunogenetic approach to spectra recognition," in *Proc. of GECCO99*, pp. 149–155, 1999.

[28] M. Esslinger, G. Lamont, H. Abdel-Aty-Zohdy, and R. Ewing, "An artificial immune system strategy for robust chemical spectra classification via distributed heterogeneous sensors," in *Evolutionary Computation, 2004. CEC2004. Congress on*, vol. 1, pp. 1036–1043, IEEE, 2004.

[29] M. Esslinger, "An artificial immune system strategy for robust chemical spectra classification via distributed heterogeneous sensors," tech. rep., DTIC Document, 2003.

[30] S. Scott, D. James, and Z. Ali, "Data analysis for electronic nose systems," *Microchimica Acta*, vol. 156, no. 3, pp. 183–207, 2006.

[31] M. Bicego, G. Tessari, G. Tecchiolli, and M. Bettinelli, "A comparative analysis of basic pattern recognition techniques for the development of small size electronic nose," *Sensors and Actuators B: Chemical*, vol. 85, no. 1, pp. 137–144, 2002.

[32] C. Distante, N. Ancona, and P. Siciliano, "Support vector machines for olfactory signals recognition," *Sensors and Actuators B: Chemical*, vol. 88, no. 1, pp. 30–39, 2003.

[33] R. E. Shaffer and S. L. Rose-Pehrsson, "Improved probabilistic neural network algorithm for chemical sensor array pattern recognition," *Analytical chemistry*, vol. 71, no. 19, pp. 4263–4271, 1999.

[34] M. Jamal, M. Khan, S. Imam, and A. Jamal, "Artificial neural network based e-nose and their analytical applications in various field," in *Control Automation Robotics & Vision (ICARCV), 2010 11th International Conference on*, pp. 691–698, IEEE, 2010.

[35] P. Vuorimaa, T. Jukarainen, and E. Karpanoja, "A neuro-fuzzy system for chemical agent detection," *Fuzzy Systems, IEEE Transactions on*, vol. 3, no. 4, pp. 415–424, 1995.

[36] J. Cho, R. Anandakathir, A. Kumar, J. Kumar, and P. Kurup, "Sensitive and fast recognition of explosives using fluorescent polymer sensors and pattern recognition analysis," *Sensors and Actuators B: Chemical*, 2011.

[37] H. Ishida, Y. Wada, and H. Matsukura, "Chemical sensing in robotic applications: A review," *Sensors Journal, IEEE*, vol. 12, no. 11, pp. 3163–3173, 2012.

[38] J. Gama and P. P. Rodrigues, "Data stream processing," in *Learning from Data Streams*, pp. 25–39, Springer, 2007.

[39] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "Data stream mining," in *Data Mining and Knowledge Discovery Handbook*, pp. 759–787, Springer, 2010.

[40] S. Vembu, A. Vergara, M. K. Muezzinoglu, and R. Huerta, "On time series features and kernels for machine olfaction," *Sensors and Actuators B: Chemical*, vol. 174, pp. 535–546, 2012.

[41] M. Trincavelli, S. Coradeschi, and A. Loutfi, "Odour classification system for continuous monitoring applications," *Sensors and Actuators B: Chemical*, vol. 139, no. 2, pp. 265–273, 2009.

[42] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

[43] V. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artificial Intelligence Review*, vol. 22, no. 2, pp. 85–126, 2004.

[44] M. Markou and S. Singh, "Novelty detection: a reviewpart 1: statistical approaches," *Signal Processing*, vol. 83, no. 12, pp. 2481–2497, 2003.

[45] M. Markou and S. Singh, "Novelty detection: a reviewpart 2:: neural network based approaches," *Signal Processing*, vol. 83, no. 12, pp. 2499–2521, 2003.

[46] P. Crook, G. Hayes, *et al.*, "A robot implementation of a biologically inspired method for novelty detection," in *Proceedings of the Towards Intelligent Mobile Robots Conference*, 2001.

[47] L. Fang and L. Le-Ping, "Unsupervised anomaly detection based n an evolutionary artificial immune network," *Applications of Evolutionary Computing*, pp. 166–174, 2005.

[48] L. Nunes de Casto and F. Von Zuben, "An evolutionary immune network for data clustering," in *Neural Networks, 2000. Proceedings. Sixth Brazilian Symposium on*, pp. 84–89, IEEE, 2000.

[49] C. Kwan, A. Snyder, R. Erickson, P. Smith, W. Maswadeh, B. Ayhan, J. Jensen, J. Jensen, and A. Tripathi, "Chemical agent detection using gc-ims: A comparative study," *Sensors Journal, IEEE*, vol. 10, no. 3, pp. 451–460, 2010.

[50] W. Chen, S. Hsu, and H. Shen, "Application of svm and ann for intrusion detection," *Computers & Operations Research*, vol. 32, no. 10, pp. 2617–2634, 2005.

[51] G. Barreto and R. Frota, "A unifying methodology for the evaluation of neural network models on novelty detection tasks," *Pattern Analysis & Applications*, pp. 1–15, 2012.

[52] F. Gonzalez, D. Dasgupta, and R. Kozma, "Combining negative selection and classification techniques for anomaly detection," in *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, vol. 1, pp. 705–710, IEEE, 2002.

[53] S. Forrest, A. Perelson, L. Allen, and R. Cherukuri, "Self-nonself discrimination in a computer," in *Research in Security and Privacy, 1994. Proceedings., 1994 IEEE Computer Society Symposium on*, pp. 202–212, Ieee, 1994.

[54] F. Gonzalez and D. Dasgupta, "Neuro-immune and self-organizing map approaches to anomaly detection: A comparison," in *First International Conference on Artificial Immune Systems*, pp. 203–211, 2002.

[55] J. Ilonen, J. Kamarainen, H. Kalviainen, and O. Anttalainen, "Automatic detection and recognition of hazardous chemical agents," in *Digital Signal Processing, 2002. DSP 2002. 2002 14th International Conference on*, vol. 2, pp. 1345–1348, IEEE, 2002.

[56] C. Becher, G. Foresti, P. Kaul, W. Koch, F. Lorenz, D. Lubczyk, C. Micheloni, C. Piciarelli, K. Safenreiter, C. Siering, *et al.*, "A security assistance system combining person tracking with chemical attributes and video event analysis," *Forschungsspitzen und Spitzenforschung*, pp. 277–296, 2009.

[57] T. Kohonen, *Self-organizing maps*, vol. 30. Springer Verlag, 2001.

[58] M. Kramer, "Autoassociative neural networks," *Computers & Chemical Engineering*, vol. 16, no. 4, pp. 313–328, 1992.

[59] X. Song, M. Wu, C. Jermaine, and S. Ranka, "Conditional anomaly detection," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 19, no. 5, pp. 631–645, 2007.

[60] D. Byer and K. Carlson, "Expanded summary: Real-time detection of intentional chemical contamination in the distribution system," *Journal (American Water Works Association)*, pp. 130–133, 2005.

[61] L. Marques and A. T. de Almeida, "Application of odor sensors in mobile robotics," in *Autonomous Robotic Systems*, pp. 82–95, Springer, 1998.

[62] M. Y. Rachkov, L. Marques, and A. T. de Almeida, "Multisensor demining robot," *Autonomous robots*, vol. 18, no. 3, pp. 275–291, 2005.

[63] N. D. Owens, A. J. Greensted, J. Timmis, and A. M. Tyrrell, "T cell receptor signalling inspired kernel density estimation and anomaly detection.," in *ICARIS*, pp. 122–135, Springer, 2009.

[64] N. Owens, *From biology to algorithms*. PhD thesis, University of York, 2010.

[65] J. A. Hilder, N. D. Owens, P. J. Hickey, S. N. Cairns, D. P. Kilgour, J. Timmis, and A. Tyrrell, "Parameter optimisation in the receptor density algorithm," in *Artificial Immune Systems*, pp. 226–239, Springer, 2011.

[66] M. Holmberg and T. Artursson, "Drift compensation, standards, and calibration methods," *Handbook of Machine Olfaction: Electronic Nose Technology*, pp. 325–346, 2002.

[67] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," *Advances in Artificial Intelligence–SBIA 2004*, pp. 66–112, 2004.

[68] A. Tsymbal, "The problem of concept drift: definitions and related work," *Computer Science Department, Trinity College Dublin*, 2004.

[69] R. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *Neural Networks, IEEE Transactions on*, vol. 22, no. 10, pp. 1517–1531, 2011.

[70] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, p. 44, 2014.

[71] J. Ma and S. Perkins, "Online novelty detection on temporal sequences," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 613–618, ACM, 2003.

[72] N. G. Pavlidis, D. K. Tasoulis, N. M. Adams, and D. J. Hand, "λ-perceptron: An adaptive classifier for data streams," *Pattern Recognition*, vol. 44, no. 1, pp. 78–96, 2011.

[73] R. Klinkenberg and I. Renz, "Adaptive information filtering: Learning drifting concepts," in *Proc. of AAAI-98/ICML-98 workshop Learning for Text Categorization*, pp. 33–40, Citeseer, 1998.

[74] M. Baena-Garcıa, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno, "Early drift detection method," in *Fourth international workshop on knowledge discovery from data streams*, vol. 6, pp. 77–86, 2006.

[75] K. Nishida and K. Yamauchi, "Detecting concept drift using statistical testing," in *Discovery Science*, pp. 264–269, Springer, 2007.

[76] H. Richter, "Detecting change in dynamic fitness landscapes," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, pp. 1613–1620, IEEE, 2009.

[77] A. Alexandridis, H. Sarimveis, and G. Bafas, "A new algorithm for online structure and parameter adaptation of rbf networks," *Neural Networks*, vol. 16, no. 7, pp. 1003–1018, 2003.

[78] S. Marsland, J. Shapiro, and U. Nehmzow, "A self-organising network that grows when required," *Neural Networks*, vol. 15, no. 8, pp. 1041–1058, 2002.

[79] H. Lau, J. Timmis, and I. Bate, "Anomaly detection inspired by immune network theory: A proposal," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, pp. 3045–3051, IEEE, 2009.

[80] D. Kifer, S. Ben-David, and J. Gehrke, "Detecting change in data streams," in *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pp. 180–191, VLDB Endowment, 2004.

[81] R. J. C. Bose, W. M. van der Aalst, I. Žliobaitė, and M. Pechenizkiy, "Handling concept drift in process mining," in *Advanced Information Systems Engineering*, pp. 391–405, Springer, 2011.

[82] A. Bifet and R. Gavalda, "Learning from time-changing data with adaptive windowing.," in *SDM*, vol. 7, p. 2007, SIAM, 2007.

[83] L. I. Kuncheva, "Classifier ensembles for changing environments," in *Multiple classifier systems*, pp. 1–15, Springer, 2004.

[84] G. Valentini and F. Masulli, "Ensembles of learning machines," *Neural Nets*, pp. 3–20, 2002.

[85] R. Polikar, "Ensemble based systems in decision making," *Circuits and Systems Magazine, IEEE*, vol. 6, no. 3, pp. 21–45, 2006.

[86] P. K. Varshney, "Multisensor data fusion," *Electronics & Communication Engineering Journal*, vol. 9, no. 6, pp. 245–253, 1997.

[87] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of Artificial Intelligence Research*, pp. 169–198, 1999.

[88] S. E. Lacy, M. A. Lones, and S. L. Smith, "A comparison of evolved linear and non-linear ensemble vote aggregators," in *Evolutionary Computation (CEC), 2015 IEEE Congress on*, pp. 758–763, IEEE, 2015.

[89] H. Wang, W. Fan, P. S. Yu, and J. Han, "Mining concept-drifting data streams using ensemble classifiers," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 226–235, ACM, 2003.

[90] W. N. Street and Y. Kim, "A streaming ensemble algorithm (sea) for large-scale classification," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 377–382, ACM, 2001.

[91] D. Brzezinski and J. Stefanowski, "Reacting to different types of concept drift: The accuracy updated ensemble algorithm," *Neural Networks and Learning Systems, IEEE Transactions on*, vol. 25, no. 1, pp. 81–94, 2014.

[92] B. S. Parker, L. Khan, and A. Bifet, "Incremental ensemble classifier addressing non-stationary fast data streams," in *Data Mining Workshop (ICDMW), 2014 IEEE International Conference on*, pp. 716–723, IEEE, 2014.

[93] N. C. Oza, "Online bagging and boosting," in *Systems, man and cybernetics, 2005 IEEE international conference on*, vol. 3, pp. 2340–2345, IEEE, 2005.

[94] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavaldà, "New ensemble methods for evolving data streams," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 139–148, ACM, 2009.

[95] J. Kolter and M. Maloof, "Dynamic weighted majority: A new ensemble method for tracking concept drift," in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pp. 123–130, IEEE, 2003.

[96] J. Kolter and M. Maloof, "Using additive expert ensembles to cope with concept drift," in *Proceedings of the 22nd international conference on Machine learning*, pp. 449–456, ACM, 2005.

[97] P. Sidhu, M. Bhatia, and A. Bindal, "A novel online ensemble approach for concept drift in data streams," in *Image Information Processing (ICIIP), 2013 IEEE Second International Conference on*, pp. 550–555, IEEE, 2013.

[98] K. Nishida and K. Yamauchi, "Adaptive classifiers-ensemble system for tracking concept drift," in *Machine Learning and Cybernetics, 2007 International Conference on*, vol. 6, pp. 3607–3612, IEEE, 2007.

[99] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine learning*, vol. 51, no. 2, pp. 181–207, 2003.

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

[101] L. L. Minku and X. Yao, "Ddd: A new ensemble approach for dealing with concept drift," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 24, no. 4, pp. 619–633, 2012.

[102] B. C. Sisk and N. S. Lewis, "Comparison of analytical methods and calibration methods for correction of detector response drift in arrays of carbon black-polymer composite vapor detectors," *Sensors and Actuators B: Chemical*, vol. 104, no. 2, pp. 249–268, 2005.

[103] C. Di Natale, E. Martinelli, and A. DAmico, "Counteraction of environmental disturbances of electronic nose data by independent component analysis," *Sensors and Actuators B: Chemical*, vol. 82, no. 2, pp. 158–165, 2002.

[104] T. Artursson, T. Eklöv, I. Lundström, P. Mårtensson, M. Sjöström, and M. Holmberg, "Drift correction for gas sensors using multivariate methods," *Journal of chemometrics*, vol. 14, no. 5-6, pp. 711–723, 2000.

[105] E. Llobet, E. Hines, J. Gardner, P. Bartlett, and T. Mottram, "Fuzzy artmap based electronic nose data analysis," *Sensors and Actuators B: Chemical*, vol. 61, no. 1, pp. 183–190, 1999.

[106] S. Di Carlo, M. Falasconi, E. Sánchez, A. Scionti, G. Squillero, and A. Tonda, "Increasing pattern recognition accuracy for chemical sensing by evolutionary based drift compensation," *Pattern Recognition Letters*, vol. 32, no. 13, pp. 1594–1603, 2011.

[107] A. Perera, N. Papamichail, N. Bârsan, U. Weimar, and S. Marco, "On-line novelty detection by recursive dynamic principal component analysis and gas sensor arrays under drift conditions," *Sensors Journal, IEEE*, vol. 6, no. 3, pp. 770–783, 2006.

[108] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[109] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[110] E. Martinelli, G. Magna, S. De Vito, R. Di Fuccio, G. Di Francia, A. Vergara, and C. Di Natale, "An adaptive classification model based on the artificial immune system for chemical sensor drift mitigation," *Sensors and Actuators B: Chemical*, vol. 177, pp. 1017–1026, 2013.

[111] L. N. de Castro and F. J. Von Zuben, "ainet: an artificial immune network for data analysis," *Data mining: a heuristic approach*, vol. 1, pp. 231–259, 2001.

[112] G. Briem, J. Benediktsson, and J. Sveinsson, "Boosting, bagging, and consensus based classification of multisource remote sensing data," *Multiple Classifier Systems*, pp. 279–288, 2001.

[113] G. Briem, J. Benediktsson, and J. Sveinsson, "Multiple classifiers applied to multisource remote sensing data," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 40, no. 10, pp. 2291–2299, 2002.

[114] A. Szczurek, B. Krawczyk, and M. Maciejewska, "Vocs classification based on the committee of classifiers coupled with single sensor signals," *Chemometrics and Intelligent Laboratory Systems*, vol. 125, pp. 1–10, 2013.

[115] Y. Maghsoudi, M. J. Valadan Zoej, and M. Collins, "Using class-based feature selection for the classification of hyperspectral data," *International journal of remote sensing*, vol. 32, no. 15, pp. 4311–4326, 2011.

[116] A. Vergara, S. Vembu, T. Ayhan, M. A. Ryan, M. L. Homer, and R. Huerta, "Chemical gas sensor drift compensation using classifier ensembles," *Sensors and Actuators B: Chemical*, vol. 166, pp. 320–329, 2012.

[117] ICARIS, "http://www.artificial-immune-systems.org/icaris/2010/competition.html," 2010.

[118] J. H. Friedman and L. C. Rafsky, "Multivariate generalizations of the wald-wolfowitz and smirnov two-sample tests," *The Annals of Statistics*, pp. 697–717, 1979.

[119] A. Gretton, K. Borgwardt, M. J. Rasch, B. Scholkopf, and A. J. Smola, "A kernel method for the two-sample problem," *arXiv preprint arXiv:0805.2368*, 2008.

[120] A. Dries and U. Rückert, "Adaptive concept drift detection," *Statistical Analysis and Data Mining*, vol. 2, no. 5-6, pp. 311–327, 2009.

[121] G. K. Kanji, *100 statistical tests*. Sage, 2006.

[122] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The annals of mathematical statistics*, vol. 18, no. 1, pp. 50–60, 1947.

[123] J. Peacock, "Two-dimensional goodness-of-fit testing in astronomy," *Monthly Notices of the Royal Astronomical Society*, vol. 202, no. 3, pp. 615–627, 1983.

[124] G. Fasano and A. Franceschini, "A multidimensional version of the kolmogorovsmirnov test," *Monthly Notices of the Royal Astronomical Society*, vol. 225, no. 1, pp. 155–170, 1987.

[125] M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: a survey," *ACM Computing Surveys (CSUR)*, vol. 45, no. 3, p. 35, 2013.

[126] M. Mitchell, *An introduction to genetic algorithms*. MIT press, 1998.

[127] D. Floreano, P. Dürr, and C. Mattiussi, "Neuroevolution: from architectures to learning," *Evolutionary Intelligence*, vol. 1, no. 1, pp. 47–62, 2008.

[128] B. Sareni and L. Krähenbühl, "Fitness sharing and niching methods revisited," *Evolutionary Computation, IEEE Transactions on*, vol. 2, no. 3, pp. 97–106, 1998.

[129] Y. Liu, X. Yao, and T. Higuchi, "Evolutionary ensembles with negative correlation learning," *Evolutionary Computation, IEEE Transactions on*, vol. 4, no. 4, pp. 380–387, 2000.

[130] S. E. Lacy, M. A. Lones, and S. L. Smith, "Forming classifier ensembles with multimodal evolutionary algorithms," in *Evolutionary Computation (CEC), 2015 IEEE Congress on*, pp. 723–729, IEEE, 2015.

[131] C. Gagné, M. Sebag, M. Schoenauer, and M. Tomassini, "Ensemble learning for free with evolutionary algorithms?," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pp. 1782–1789, ACM, 2007.

[132] H. Abbass *et al.*, "Pareto neuro-evolution: Constructing ensemble of neural networks using multi-objective optimization," in *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, vol. 3, pp. 2074–2080, IEEE, 2003.

[133] H. A. Abbass, "A memetic pareto evolutionary approach to artificial neural networks," in *AI 2001: Advances in Artificial Intelligence*, pp. 1–12, Springer, 2001.

[134] A. Chandra and X. Yao, "Ensemble learning using multi-objective evolutionary algorithms," *Journal of Mathematical Modelling and Algorithms*, vol. 5, no. 4, pp. 417–445, 2006.

[135] T. Escovedo, A. Abs da Cruz, A. Koshiyama, R. Melo, and M. Vellasco, "Neve++: A neuro-evolutionary unlimited ensemble for adaptive learning," in *Neural Networks (IJCNN), 2014 International Joint Conference on*, pp. 3331–3338, IEEE, 2014.

[136] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, 2002.

[137] S. Stepney, "How many times should you run your simulation?." `http://susan-stepney.blogspot.co.uk/2015/06/how-many-times-should-you-run-your.html`, 2015. [Online; accessed 23-November-2015].