**Head Office:** Università degli Studi di Padova
**Department:** Information Engineering
**Ph.D. Course in:** Information Engineering
**Curriculum:** Information Science and Technology

# Improving Anomaly Detection for Industrial Applications

Thesis written with the financial contribution of Pietro Fiorentini S.p.A.
Grant: Data Analytics for Multi-phase flow meters

**Coordinator:** prof. Andrea Neviani
**Supervisor:** prof. Gian Antonio Susto
**Co-Supervisor:** dr. Enrico Feltresi

**Ph.D. student:** Tommaso Barbariol

## ABSTRACT

In the last decade the availability of Big data and Cloud computing has pushed the research community towards the development of algorithms able to automatically learn patterns from data. Despite the huge versatility of these models, they usually require a large number of data and computational power, that are seldom met in industrial scenarios where data are expensive to obtain and devices have strict constraints on their computing capabilities. This work seeks to address these challenges, developing algorithms that can perform in the mentioned scenario and can improve the reliability of industrial devices. This indeed can be achieved by tacking advantage both of data coming from machine' sensors and algorithms that are able to detect anomalies or possible malfunctions. In the present thesis multiple algorithms will be proposed to reach this goal, explaining the industrial challenges by means of an industrial use case. The main focus will be on industrial Anomaly Detection and how to enhance the reliability of industrial devices, however the faced challenges pushed us to formulate new research questions and consequently to contaminate this topic with many other approaches in the context of Machine Learning.

In this work, we will address the following research questions: i) How to update the model if the user provides feedback? ii) Which sensor is responsible for the anomalies? iii) Is there a way to compress the memory and computations a model needs? vi) Which are the most interesting data points to show a model? v) Is it possible to adapt a detector between different machines? We will investigate such research directions by providing new algorithms and performing experiments showing performances on real scenarios.

## SOMMARIO

Negli ultimi dieci anni la disponibilità di grandi quantità di dati e potenza di calcolo ha spinto la comunità scientifica verso lo sviluppo di algoritmi capaci di imparare autonomamente dai dati. Sebbene questi modelli si siano dimostrati estremamente versatili, solitamente richiedono una grande mole di dati e di capacità computazionali, che sono rare in contesti industriali dove ottenere dei nuovi campioni è costoso e dove i dispositivi hanno poche risorse di calcolo. Questo lavoro si propone di affrontare queste sfide, sviluppando algoritmi che aumentino l'affidabilità dei dispositivi industriali combinando dati provenienti dai sensori della macchina con algoritmi capaci di individuare anomalie o possibili malfunzionamenti. In questa tesi sono illustrate numerose sfide che il contesto industriale pone nei confronti di tali metodi; a titolo di esempio verrà utilizzato un particolare caso d'uso proveniente dal mondo dell'Oil&Gas. Sebbene il principale obiettivo sia l'aumento dell'affidabilità dei macchinari industriali attraverso il rilevamento delle anomalie, durante le svolgimento della tesi sono emersi nuovi quesiti che hanno richiesto l'utilizzo di altre tecniche di apprendimento automatico. Ci si è chiesto infatti come includere feedback sul comportamento del modello forniti dall'utente oppure se fosse possibile capire la causa radice che ha portato un rilevatore di anomalie a segnalare un certo comportamento. Sono state esplorate anche problematiche di tipo computazionale cercando di trovare strade per ridurre le risorse necessarie all'algoritmo. Per quanto riguarda i dati ci si è chiesto quali fossero i campioni più importanti su cui allenare il modello in modo da ridurre il numero di dati necessari ad allenarlo. Infine è stata esplorata l' applicazione del modello a macchine leggermente differenti, trasferendo informazioni da una macchina all'altra. Per tutti questi quesiti sono stati proposti degli algoritmi che sono stati successivamente testati in scenari industriali reali.

# PUBLICATIONS

During the PhD activities that culminated with this thesis, several works have been submitted and/or published in various venues:

[1] Alessio Arcudi, Tommaso Barbariol, Chiara Masiero, and Gian Antonio Susto. “ExIFFI: Extended Isolation Forest Feature Importance.” In: *Submitted to Expert Systems with Applications* (2022).

[2] Tommaso Barbariol, Filippo Dalla Chiara, Davide Marcato, and Gian Antonio Susto. “A review of tree-based approaches for anomaly detection.” In: *Control Charts and Machine Learning for Anomaly Detection in Manufacturing* (2022), pp. 149–185.

[3] Tommaso Barbariol, Enrico Feltresi, and Gian Antonio Susto. “Machine learning approaches for anomaly detection in multiphase flow meters.” In: *IFAC-PapersOnLine* 52.11 (2019), pp. 212–217.

[4] Tommaso Barbariol, Enrico Feltresi, and Gian Antonio Susto. “Validity and consistency of MPFM data through a Machine Learning-based system.” In: *Proceeding 37th North Sea Flow measurement Workshop* (2019).

[5] Tommaso Barbariol, Enrico Feltresi, and Gian Antonio Susto. “A machine learning-based system for self-diagnosis multiphase flow meters.” In: *International Petroleum Technology Conference*. OnePetro. 2020.

[6] Tommaso Barbariol, Enrico Feltresi, and Gian Antonio Susto. “Self-diagnosis of multiphase flow meters through machine learning-based anomaly detection.” In: *Energies* 13.12 (2020), p. 3136.

[7] Tommaso Barbariol, Enrico Feltresi, Gian Antonio Susto, Diego Tescaro, and Silvia Galvanin. “Sensor Fusion and Machine Learning Techniques to Improve Water Cut Measurements Accuracy In Multiphase Application.” In: *SPE Annual Technical Conference and Exhibition*. OnePetro. 2020.

[8] Tommaso Barbariol, Julien Lesouple, Gian Antonio Susto, and Jean-Yves Tourneret. “Unbalanced Optimal Transport for Domain Adaptation in Anomaly Detection tasks.” In: *Submitted to Engineering Applications of Artificial Intelligence* (2022).

[9] Tommaso Barbariol and Gian Antonio Susto. “TiWS-iForest: Isolation forest in weakly supervised and tiny ML scenarios.” In: *Information Sciences* 610 (2022), pp. 126–143.

[10] Tommaso Barbariol, Gian Antonio Susto, Davide Masiero, and Enrico Feltresi. “Time series forecasting to detect anomalous behaviours in MPFM.” In: *Proceeding 38th North Sea Flow measurement Workshop available at arXiv preprint arXiv:2301.00014* (2020).

[11] Luca Frau, Gian Antonio Susto, Tommaso Barbariol, and Enrico Feltresi. “Uncertainty Estimation for Machine Learning Models in Multiphase Flow Applications.” In: *Informatics*. Vol. 8. 3. MDPI. 2021, p. 58.

[12] Elisa Marcelli, Tommaso Barbariol, Vincenzo Savarino, Alessandro Beghi, and Gian Antonio Susto. “A Revised Isolation Forest procedure for Anomaly Detection with High Number of Data Points.” In: *2022 IEEE 23rd Latin American Test Symposium (LATS)*. IEEE. 2022, pp. 1–5.

[13] Elisa Marcelli, Tommaso Barbariol, and Gian Antonio Susto. “Active Learning-based Isolation Forest (ALIF): Enhancing Anomaly Detection in Decision Support Systems.” In: *arXiv preprint arXiv:2207.03934* (2022).

[14] Davide Sartor, Tommaso Barbariol, and Gian Antonio Susto. “Bayesian Active Learning Isolation Forest (B-ALIF): a Weakly Supervised Strategy for Anomaly Detection.” In: *Submitted to Engineering Applications of Artificial Intelligence* (2022).

# ACKNOWLEDGMENTS

This work has been possible thanks to the support of Pietro Fiorentini S.p.A. a manufacturing company settled in Italy that offers technically-advanced solutions for the energy supply chain, that meet the needs of a wide range of different customers: from renewable energy producers to oil and natural gas production companies (both onshore and offshore), gas transmission companies and distribution utilities.

They are also investing in green gas production, in particular in startups and projects that exploit the potential of alternative sources such as biomethane, hydrogen and power-to-gas, with the aim of decarbonising the energy sector and developing zero-emissions systems.

Concerning oil & gas production, they offer a wide range of solutions for companies operating in the extraction, storage and treatment of oil and natural gas, ensuring the highest level of reliability even under the most challenging conditions.

Pietro Fiorentini produces systems, components and services both for the pipeline network and for LNG transportation, covering every stage from processing facilities to the entry into distribution networks.

Thousands of their systems and components are installed in medium and low pressure gas distribution networks all over the world indeed they are specially designed to regulate gas pressure to the correct level depending on use, reaching even the most remote places.

Figure 1: Pietro Fiorentini S.p.A. logo

# CONTENTS

# ACRONYMS

| | |
|---|---|
| MPFM | Multiphase Flow Meter |
| VFM | Virtual Flow Meter |
| TS | Test Separator |
| WLR | Water Liquid Ratio |
| GVF | Gas Volume Fraction |
| Qtot | Total flow rate |
| ML | Machine Learning |
| SftS | Soft Sensing |
| UQ | Uncertainty Quantification |
| AD | Anomaly Detection |
| AL | Active Learning |
| XAI | Exlainable Artificial Intelligence |
| TinyML | Tiny Machine Learning |
| DA | Domain Adaptation |
| SMS | Smart Monitoring Systems |
| RUL | Remaining Useful Life |
| R2F | Run To Failure |
| PvM | Preventive Maintenance |
| PdM | Predictive Maintenance |
| AS | Anomaly Score |
| IF | Isolation Forest |
| RF | Random Forest |

# 1 INTRODUCTION

The fourth industrial revolution is a game-changer in the automation of industrial processes, allowing for self-optimization, self-cognition and self-customization. Rather than relying on direct programming and guidance by a human operator, the industrial devices are now expected to operate autonomously managing their own upkeep. This new development means that the machines can now collect, understand and draw conclusions from data on its own.

Three main approaches to maintenance are envisioned in literature [47, 53, 59, 277] which differ in the way a possible malfunction is handled:

- Run To Failure (R2F) is the simplest possible approach to maintenance, indeed it waits for the fault to intervene. This choice can be very dangerous as a sudden breakdown might interfere with other production activities linked to the broken equipment.

- Preventive Maintenance (PvM) is the classic maintenance strategy, indeed in this case the maintenance is scheduled at fixed intervals in order to anticipate the failure. It is more safe than the previous choice, but it can be expensive: some intervention might be useless and some spare parts might be substituted before they are actually worn.

- Predictive Maintenance (PdM) represents a paradigmatic change in maintenance operations, indeed this strategy makes use of models able to predict when the piece needs to be replaced or maintained. This allows to avoid unnecessary operations but at the same time to promptly intervene when a fault is approaching.

A key factor of this transition process are Smart Monitoring Systems (SMS): the equipment must be capable of measuring its health status and to act accordingly. If the SMS detects a possible breakdown, the equipment might ask for human intervention or might try to compensate the fault automatically. In the development of SMS two opposite but complementary approaches can be followed: the diagnostic and the prognostic approach. The first concerns with what is happening or what has already happened in the past: the machine detects a *fault* or perceives a possible *anomaly* during its operations. After that, the SMS should be able to understand which was the root cause and take an autonomous decision. On the contrary, the prognostic approach refers to what is likely to occur in the future: the machine should be able to predict in sufficient time when it will experience a fault, so the Remaining Useful Life (RUL) of a component.

A challenging application of these concepts is the Multiphase Flow Meter (MPFM): developed in the ’80s, the MPFM has begun to spread since the ’00s. It is used to measure the individual phase flow rates of a multiphase flow and indeed it is widely adopted by the oil companies that install it on oil extraction systems: it provides real-time simultaneous measurements of the commingled flow of oil, water and gas of a well combining the measurement of several sensors. It is very appreciated because it measures the fluid properties in a non-intrusive way and therefore it doesn't need to separate the phases and to stop the production [79]. The two most important measurements provided by the instrument are the ratio of water over the liquid flow and the fraction of gas over the total flow. These data are subsequently used by the oil company to measure the performance of the deposit and to forecast the future production. The MPFM experiences a variety of fluid flows [78]: depending on the well conditions that can strongly vary between different geographical areas, the meter measures a process that continuously evolves in time and draws new fluid patterns.

As it can be imagined, the MPFMs are installed in remote areas of the world where maintenance can be very expensive. A potential fault is costly: from the point of view of the supplier a malfunction forces the service to an unscheduled maintenance, while from the point of view of the oil company the fault forces the production to stop. This not only reduces the profit, but also affects the physical behaviour of the oil reservoir.

Other characteristics make the adoption of SMS on MPFM very challenging. The computation of semi-physical model that provides the phase flow rates exploits a large part of the computational resources of the instrument, forcing the SMS to cope with this limitation.

The aim of this project is to study the Smart Monitoring challenges previously described in the MPFM application scenario and to develop new data-driven methodologies for their solution. Great attention will be paid to Machine Learning (ML) approaches that in recent years are showing their abilities to model complex patterns from huge amounts of data. In this context a big effort has been dedicated to the development and application of *Soft Sensing* techniques able to improve the flow rate estimation over the physical model, and *Anomaly Detection* tools to detect unusual sensor behavior, possibly related to ongoing faults.

This thesis is organized as follows: Chapter 2 accurately describes the use-case and the challenges this work tries to address, then Chapter 3 focuses on Soft Sensing estimation of water extracted by the oil field, while Chapters 4,5,6,7,8,9,10 describe the solutions proposed by this work to the anomaly detection challenges. In the final Chapter 11 the achieved goals are summarized and future directions are drawn.

# 2 USE CASE AND RELATED CHALLENGES

The motivation behind the present work is discussed in this Chapter together with the main challenges that have been faced.

As we push forward the performances of industrial equipment increasing its complexity, maintenance and reliability become central aspects. In this context machines are expected to self-diagnoses and to collaborate with humans in order to solve potential issues that might compromise the production. This ability is mainly allowed by the advance of a new set of techniques that come under the name of Machine Learning (ML) and proved to be very effective not only in difficult popular tasks like audio and image recognition but also in predicting the remaining useful life of industrial equipment [178]. ML includes many different techniques that are associated by the ability to learn in autonomous fashion from a large set of data. This allows to reduce the human intervention and to find complex patterns that would be otherwise difficult to identify with traditional techniques.

ML models can be divided into two main categories, depending on which data they rely on: data describing the link between *input* and *output*, often named *labelled* data, are handled by *supervised* techniques; the most common tasks belonging to this category are *regression* and *classification*. On the other side data that lacks a label describing a desired output are named *unlabelled*; *unsupervised* techniques can be used to infer some information from these data, for example data can be clustered in groups or samples having an anomalous behaviour can be detected. In practical scenarios, having labels associated to each sample is uncommon [217] as the labelling procedure is often difficult to perform or expensive: to assign a label requires a human operator that is expert in the specific domain that is asked to model, moreover obtaining some labels might be dangerous or might require some expensive destructive experiments.

In this work the goal is to expand the Predictive Maintenance techniques using the ML approach, both with unsupervised and supervised tools depending on the available data. The techniques developed in the present thesis are intended to be general, but were developed keeping in mind a specific use case that is described in the following Section 2.1. This use case, named Multiphase Flow Meter, is very interesting as it poses numerous interesting challenges in the context of PdM that can be generalized to other industrial equipment like engines [77], satellites [158], natural gas compressors [183] or robotic arms [266].

## 2.1 USE CASE: THE MULTIPHASE FLOW METER

The Multiphase Flow Meter (MPFM) is an example of a complex measuring system composed of many measuring modules and requiring data fusion to

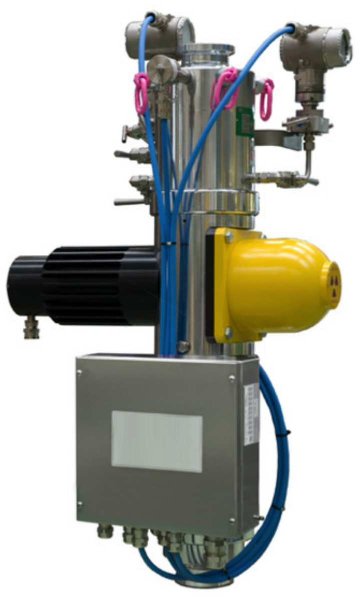


Figure 2: A Multiphase Flow Meter built in a modular way: it is possible to observe the Venturi tube placed in vertical position with the pressure probes and the densitometer placed in horizontal position. The impedance probes are installed inside the tube in close contact with the fluid.

provide the target measure (Figure 2). The MPFM is an in-line measuring instrument able to quantify the individual flow rates of oil, water, and gas coming from an oil well; as it does not separate the phases, it is able to provide measures in real-time, differently from other systems like separators that stop the flow until the three components are fully unmixed.

The Gas Volume Fraction (GVF), the Water Liquid Ratio (WLR), and the Total flow rate (Qtot) are important production parameters in the estimation of the individual flow rates of oil, water, and gas. The WLR and GVF are defined as functions of the single-phase and total flow rates as follows:

$$GVF = \frac{Q_{gas}}{Q_{total}} \qquad WLR = \frac{Q_{water}}{Q_{oil} + Q_{water}}$$

and the total flow rate is simply:

$$Q_{total} = Q_{oil} + Q_{water} + Q_{gas}$$

An accurate estimation of these parameters is essential not only for the extraction, production, and management activities of a single well but of whole fields and reservoirs too. In particular, an accurate WLR measurement is required to determine the maturity of a field and to estimate its lifetime, to understand the amount of water produced, as it is an expensive by-product, since it can greatly increase the extraction cost, and to optimize the chemical injection needed to avoid scaling, corrosion, emulsion, or formation of hydrates, which can even stop the production. Directly measuring the WLR is not an easy task, as a consequence, multiple solutions have been presented in the literature to tackle this issue [78, 105, 239].

### 2.1.1 *Alternatives to MPFM*

The most widespread technology for flow measurement is the Test Separator (TS), which consists of a pressure vessel where the multiphase flow is separated into its single phases. Indeed, the mixture coming from the reservoir is transported into the tank, and it is generally split into three phases (oil, gas, and water) by means of gravity and additional chemicals. Once the phases are separated, on each leg of the TS, the measurements are performed with technologies such as Venturi or Coriolis meters [105, 188]. In most situations, the test separator method is sufficiently accurate and reliable; however, it is not able to operate in real time, since the separation of the three-phase flow might take a lot of time, especially in the presence of emulsions. Moreover, the separator occupies a lot of space, causing logistic problems, for example, in platform installation, where the room available is limited. In addition, it is not able to deal with different flow regimes and requires long stabilization times before reliable measurements can be obtained [188, 239]. Because of these drawbacks, it is convenient to operate directly on the three-phase flow by introducing the MPFM, a tool that combines different measurement technologies with the purpose of obtaining information from the multiphase flow without performing the time-consuming separation. Moreover, its physical footprint is negligible compared to that of the TS, and it is almost independent of the flow regime. The MPFM described here provides precise measurements, but it is quite expensive, and it requires manual intervention if there is a sensor failure, which increases the operational cost [33, 128]. Furthermore, MPFMs have an operation range beyond which the accuracy of the flow rate parameter estimates can dramatically decrease. Therefore, recently, there has been much research on the search for a more reliable and cheaper measurement system. An alternative to the MPFM is the Virtual Flow Meter (VFM) [11, 33], a tool that is able to provide flow rate estimations using field data that are already available for the production site without the employment of expensive measurement instruments. The usual data available are:

- Bottomhole pressure and temperature
- Wellhead pressure and temperature upstream of the choke
- Wellhead pressure and temperature downstream of the choke
- Choke opening (i.e., the percentage of opening of the choke valve)

These are fed into numerical models that make the predictions. This approach is clearly advantageous for both maintenance costs and hardware costs; the VFM is able to perform in real time and can be used both standalone or coupled with an MPFM as a backup in situations in which the latter is unavailable for some of the aforementioned reasons.

### 2.1.2 *The MPFM's equipment*

MPFMs are employed both onshore and offshore, in topside, or subsea configuration. In this work, data have been collected using a MPFM [195], which is made in a modular way. As each module contains a specific set of sensors, in the following paragraphs the words sensor and module will be treated as synonyms.

The most common instruments employed by MPFMs are [105]:

VENTURI TUBE It is the most popular instrument [268], low cost and quite cheap compared to other equipment as it consists of a pipe with a shrinkage in the middle, named throat. In this location, the temperature (T) and absolute pressure (P) are usually measured, together with the pressure difference (DP) between the inlet and the throat. The Venturi meter structure can be split into three parts: a first restriction of the pipe, called *contraction section*, a middle part called *throat* and a final part called *diffusion section* where the pipe section expands again. The physics behind the Venturi Meter is the continuity equation and the Bernoulli equation of fluids: $p + \frac{1}{2}\rho v^2 + \rho g h = constant$, where $p$,$v$,$h$,$\rho$ are the pressure, the velocity, the height and the density of the fluid. The principle states that the sum of these three terms must be constant in every point of the pipe: therefore, when the fluid goes through the contraction section, the velocity increases, and the static pressure decreases. Thus, a positive differential pressure is created before and after the contraction section. On the other hand in the diffusion section the flowrate slows down creating a negative differential pressure. These two pressure differences are labeled as dP1 and dP2 and are proportional to the density and the velocity of the fluid, which are useful to understand the flow composition. Usually, only the first differential pressure is used, as well as in the current work [27]. The main drawback of the Venturi meter is that it needs to measure a fluid with an almost constant composition [105]: to overcome this problem, the Venturi meter is usually combined with other sensors like gamma densitometers.

IMPEDANCE SENSOR Electrical impedance sensors are made by electrodes placed around the pipe in electrical contact with the mixture [105]: a current is injected and the voltage on the electrodes is measured. Cross correlating these measurement, the velocity of the fluid can be inferred. In the presented work, there are three sets of electrodes that provide three sets of signals, which can be useful to increase reliability: if a sensor fails, there are still two that can be used for the measurement. The sensors work in different modes depending on the flow composition [27]: if the concentration of water is higher than the oil one the fluid is *water continuous*, therefore the flow is conductive and the sensor measures the conductivity of the fluid. In an *oil continuous* situation, instead, the permittivity is measured since the fluid is capacitive. The main drawback is the high uncertainty in the transition region, i.e. between water and oil continuous phases. To be oil continuous, water must be dispersed in oil in a way water droplets do not form a continuous path between the electrodes: as long as the WLR is below 60% - 70%, the flow remains oil continuous [55].

GAMMA RAY DENSITOMETER This technology uses a radioactive source to retrieve information on the composition of the multi-phase flow. The source radiates gamma rays to a detector through the mixture, therefore the sensor is able to tell how much the fluid has absorbed the radiations: since the attenuation is dependent on the composition of the mixture, this can give information on the fraction of each phase.
A first drawback of this method is of course the presence of a radioactive source, which can be harmful both to operators and the environment [271]; moreover, depending on the beam energy emitted from the radioactive source, the accuracy of the measurement may vary depending the salinity of the water. In fact, for low-energy sources, freshwater and saltwater have different attenuation coefficients, leading to different results as the salinity changes [105]; instead, for high-energy sources, the attenuation is independent from the salinity content [30]. In the presented work, the source of gamma rays is the radioactive isotope of Cesium $^{137}$Cs, which has a half-life of almost 30 years and a beam energy of 662 keV, thus making the measurement insensitive to salinity variations. Finally, as already said before, a gamma densitometer should be coupled with another measurement source, like a Venturi meter: otherwise, it can only measure the average density of the mixture.

The stream of data coming from these sensors is then fed into a model that is able to estimate the three quantities of interest. This is usually based on physical conservation equations, but it requires closure equations to get a solvable system of equations. Unfortunately, due to the complexity of the modelled process, the closure equations are represented by empirical equations that need to be accurately tuned to the specific working condition. In recent years, there has been a growing interest in the application of Machine Learning (ML) data fusion techniques to this kind of instruments to obtain easier and more accurate models to predict the flow composition [25, 207].

### 2.1.3 *Bench test and data collection*

Dataset point collection consists in gathering flow information with the MPFM technologies. This cannot be done *in situ*: firstly because it would exponentially increase the costs of data gathering and secondly because the flow composition cannot be controlled. In fact, one may want to collect more data in some particular parameter areas: this cannot be done in the production site, because the mixture coming from the reservoir is unknown.
Instead, data are collected in a physical test bench called *flow-loop*: it is a laboratory instrument for investigating the flow characteristics in pipes and for studying the response of MPFM instruments to this flow. The mixture is circulated continuously in a loop, while MPFM instruments, that can be

placed at different deviations from vertical through horizontal, collect signals at that flow condition. The multi-phase properties, fractions and velocities can all be varied, making the exploitation of a desired configuration quite handy. The data, used in this thesis, have been collected at ProLabNL[197], an high pressure multiphase flow loop, using hydrocarbon gas and crude oil, which provides the possibility to test a MPFM against a precise set of references: temperature, pressure and oil, water and gas rates and densities. This is done with the employment of the following instrumentation:

- Gas and Liquid reference meters
- Temperature, pressure and pressure difference transmitters
- Online Gas Densitometer
- Nucleonic Level Measurement

### 2.1.4 *General MPFM requirements*

The main characteristics that customers and producers look for in multiphase flow meters are summarized in [35, 105], and they are:

NON-INTRUSIVITY The technology employed to get information on the flow must not interfere with the flow of the mixture. In this sense the test separator is not a good option as it is a very intrusive instrument.

FLOW REGIME INDEPENDENCE Different flow patterns appear depending on several factors [105, 234], such as the flow properties (phase, velocity, fraction), reference pressure and temperature, pipe physical properties and direction, presence of obstructions (valves, junctions), and flow state (steady state or in transition). The typical flow regimes are bubble (where the gas bubbles are dispersed in the liquid), slug (where the gas, increasing its velocity, tends to form larger and more consistent bubbles), churn (the transition phase), annular (where the gas flows in the internal part of the tube, while the liquid phase runs only in the external part), and dispersed flow (where the liquid part is divided into small droplets). The meter is expected to perform equally well in all of the flow conditions. An ideal meter measures the phases in all of these conditions.

ACCURACY AND RELIABILITY The MPFM measurements should be consistent, coherent, and precise, since measurements and, most of all, their accuracy have significant consequences for the management of wells, fields, and reservoirs. For example, in the presence of unreliable measurements, such as in transition regions, the MPFM should alert the user by assigning a large uncertainty to the provided estimates.

It is important to stress the context in which the MPFM is installed to fully understand the challenges this instrument has to cope with: no matter the precise location where it is installed, the MPFM is always installed in very remote and harsh environments where the instrument itself is severely stressed, the maintenance is impossible or very hard to make and the connectivity is absent. In this scenario the machine must be autonomous both in the normal working operations, and in self-diagnosing potential issues to its instrumentation.

In this context the MPFM poses many interesting and challenging practical problems to solve:

INEXACT PHYSICAL KNOWLEDGE The physical model employed to estimate the individual flow rates is not sufficiently accurate, indeed it needs many empirical closure equations to get a system that can be actually solved. Moreover it needs a time consuming period of calibration to set all the model parameters.

UNRELIABLE MEASUREMENTS IN TRANSITION REGIONS In between flow transitions, the measurements provided by the instrument might be not very accurate. This is because both the model is inaccurate, and the sensors do not get stable measurements. An example is the transition between oil and water continuous phases: in this case the impedance sensor flips between the capacitive and the resistive circuit, and might provide unreliable measurements.

MULTIVARIATE FAULTS The MPFM is made up of many sensors that interact between each other so a malfunction in one of these might affect the others. Moreover the dynamics of the measured flow can change a lot during the lifetime of the instrument. As a consequence, some faults and anomalies can be detected looking at the interaction between the sensors measurements. However, once a fault is detected, it is not easy to understand which is the module responsible of the malfunction, making difficult to analyse the root cause of the problem.

LACK OF GROUND TRUTH LABELS It is very hard to test an instrument like this, as the test bench has to replicate as accurately as possible the real working conditions of an oil well. This means that controlled experiments performed in test benches, named *flow loops*, are very expensive to do and acquisition campaign might be reduced as much as possible. Also field tests are not easy to be performed due to the harsh environmental conditions and the lack of connectivity. This leads to scarcity of labelled data both showing faulty or not-faulty behaviours, and the real individual rates of the fluid flow.

MANY PROTOTYPES This instrument is not built in series but is tailor-made on the customer needs, leading to a variety of slightly different machines. They can vary in size, but also in the number and kind of modules they employ; the behaviour changes from machine to machine, therefore obtaining a unique model that works for each machine is a difficult task.

LOW COMPUTATIONAL POWER AND MEMORY RESOURCES Despite the instrument complexity, the computational power left after the computations of the physical model is quite low. Due to this fact, every model that needs to be run on board, has to cope with strict computational requirements.

Many of these practical problems are common to other instruments frequently encountered in industry like airplane engines or satellites. These problems have been addressed from a more theoretic point of view using data driven techniques that come under the name of Machine Learning and will be discussed in more detail in the following Section and Chapters.

## 2.2 TECHNIQUES TO SOLVE THE CHALLENGES

To solve the aforementioned practical issues it has been decided to rely on data driven approaches, often named *Machine Learning*. This choice has been motivated by the great flexibility these methods provide and by their ability to perform well in a variety of scenarios. The advantage of using these tools is they do not need the knowledge of complex physical equations to model the system, but they just require to have some data where to train a model. This highly reduce the modelling effort and lets to speed up its deployment.

In general, Machine Learning (ML) techniques include a variety of methods able to automatically learn from data, without the need for the user to explicitly write the model. There exist plenty of tasks that can be solved with the use of these methods like classification, regression or clustering. Two of the most important tasks in the context of this thesis are named Soft Sensing (SftS) and Anomaly Detection (AD). SftS exploits the availability of data that are already collected by the machine and use regression models to estimate expensive quantities, allowing to have an estimation when real metrology instruments are not in place or, possibly, to reduce the number of "physical" sensors. On the other hand, the goal of AD models is to look for data that are somehow different from the majority, and do not follow its pattern.

In this work SftS has been addressed in Chapter 3, paying attention to Uncertainty Quantification (UQ) techniques able to estimate not only the quantity of interest but also the associated level of uncertainty. On the other side, concerning the detection of anomalies in the sensing devices, AD has been widely studied, employed and extended in new directions.

In particular the classic Anomaly Detection task has been discussed in Chapters 4 and 5, contaminated with Tiny Machine Learning in Chapter 6, with Active Learning in Chapters 7 and 8, with explainable AI in Chapter 9 and Domain Adaptation topics in Chapter 10.

The Machine Learning with the adjective *tiny* is a new branch of ML (Tiny Machine Learning (TinyML)) that tries to develop new models that can run on computational and memory constrained devices like micro-controllers (MCU). This is very complex but allows to get huge energy savings, lower latency, lower deployment costs and increased privacy. These concepts were used to develop a compression algorithm able to fit an anomaly detector in less memory and with less time complexity.

Active Learning (AL) is the research area that studies how to make interact the model user with the algorithm and relies on the assumption that if the algorithm is let free to ask for new labels, it is able to reach better performance with less labelled data. This topic was particularly useful in relation to the scarcity of labels showing if the machine was exhibiting faulty or not faulty behaviour, indeed thanks to this work it was possible to train a better detector with less human labelling effort.

ML models are notoriously black boxes, in the sense that it is very hard to understand why the model took a particular decision. To overcome this limitation in recent years a new research area is growing, named Exlainable Artificial Intelligence (XAI), that tries to develop easier to be explained models or algorithms able to explain the predictions made by black box models. In the context of this thesis it is important to provide the user with an intuition as to why the model suggests that a particular condition might be faulty, and which is the sensor responsible for it.

Domain Adaptation (DA) on the other side is a branch of Transfer Learning, i.e. the research area that studies how to transfer knowledge between different but related tasks. It proved its usefulness when developing models able to adapt to different prototypes or machines, without the need of fully re-sampling every new machine that needs a detector.

# 3 SOFT SENSOR

The goal of this work, to make the instrument more reliable, can been reached by two paths: a first approach is to focus on the *output* i.e. the flow rate estimation model, indeed a more accurate model means better predictions and uncertainty estimations. A second and complementary approach is to focus on the *input* of the model because, to have a good estimation model, is not sufficient if the data that are fed into the model are unreliable.

The first approach, focused on the model improvement, has been reached with the use of Soft Sensing (SftS) and Uncertainty Quantification (UQ) techniques that are the subject of this Chapter. This work has been published in [88].

## 3.1 INTRODUCTION

Oil and natural gas are still one of the world most important basic goods, but the pandemic crisis started in 2020 and governments pushing towards a low-carbon future temporally collapsed the demand [176]. This is driving oil companies to look for methods and techniques that will, on one side, reduce the cost related with the overall extraction, drilling and other production activities, but also they will increase the extraction efficiency. Oil and gas companies are nowadays focusing their effort to overcome this unforeseen and sudden shift in the market request. The optimization of the exploration and production is definitely one way to deal with the actual situation.

Therefore, it is crucial to have accurate measurements equipment, since they can assist and support the reservoir and production engineers in the decision making. Moreover accurate measurements are used for many tasks like production management, tax allocation and oilfield modeling. In this context uncertainty measures can be fed into these models in order to have a more detailed understanding of the production trend and a more reliable vision of the process to take the required decisions. Besides, knowing the accuracy or the relative uncertainty of real-time measurements will help engineers and operators to define the optimal parameters to manage the wells and the whole reservoir.

The estimation of the uncertainty confidence, for a data-driven model, is becoming a common and highly requested requirement, as crucial as the prediction interpretability. This applies to many areas, not only the energy sector, but also where the prediction accuracy has a high impact on the outcome like finance and medicine. In particular, the only pointwise inference is no more sufficient to complete successfully a task, but confidence estimations for the predicted measures are needed. The most natural way to address this requirement is using the Uncertainty Quantification (UQ) approach. UQ is a set of statistical tools that determines uncertainties associated with a given model, aiming to consider all possible and reasonable uncertainty sources in order to correctly assess a comprehensive uncertainty [57]. The aim of this work is to present methods, applied to Multiphase Flow Meter (MPFM), that will return not only accurate predictions of production parameters, but also uncertainty confidence estimations.

### 3.1.1 *Previous work*

Focusing on the uncertainty estimation, the literature concerning the application of these methodologies to real life problems is quite vast, it ranges from chemistry [121, 187] to material science [240], localization [154], geology [14] and of course medical science [8] where the majority employs techniques based on bootstrap, Bayesian approaches and dropout.

### 3.1.2 *Novelty and previous works*

The novelty provided by this Chapter is the application of uncertainty estimation to the Multiphase Flow estimation problem. Usually, in the found literature, only the quality of the prediction is treated, while the uncertainty of the prediction is not reported: therefore, another goal will be to provide a confidence interval for the predictions. To the best of our knowledge this is the first described applications of these methodologies to Multiphase Flow Meters in literature, and therefore benchmarking papers are still absent. However in recent years multiple authors are approaching the *standard* multiphase problem, using a variety of tools and different settings [261].

The implementation of artificial Neural Networks (NN) is particularly trending nowadays, for their ability to determine non-linearities, which are very useful especially for WLR estimation, as well as for their various architectures that adapts to the type of measurement signals feed as input. In [208] a gamma-ray densitometer measures the attenuation of gamma rays through the fluid, and use it as an input feature for a Radial Basis Function Network (RBFN) (the innovation is using only the 4 strongest peaks instead of the entire gamma spectrum): with this configuration, they were able to estimate both flow patterns and values of volume fractions. A similar network was used also in [201], where a RBFN is implemented and validated in estimating gas flow rate. In [271], the authors tried to estimate WLR by collecting time-series data from a microwave-based measurement technology: the setup was made by dual sensor that is able to measure both the attenuation and the phase shift of the wave through the fluid. Signal information was then extracted by applying two different methods: wavelet transform and CNN, where the latter perform slightly better than the former. In [20] the authors tried to predict the flow rates of oil, gas and water in a three-phase flow: the aim was reducing the cost of data gathering by implementing an NN with only flow parameters (temperature, viscosity and pressure signals) and statistical parameters of the pressure signal (namely standard deviation, kurtosis and skewness coefficients) as inputs. In particular, they compared the ability of the network with two different inputs combinations, first using only flow parameters, then adding statistical parameters: they were able to show that the prediction improved in the latter case, with an *R-score* of at least 0.995 in all three cases. In [134], Khan implemented various types of AI techniques, besides a classical NN: an Adaptive neuro-fuzzy inference system (ANFIS) which is an hybrid version of neural network and fuzzy logic system; a Support Vector Machine (SVM), functional nets (FN), which is a network where different neurons embed different functions. The application of these techniques led to a prediction with range of accuracy of 96-99%. In many works, due to the type of data handled, a particular focus is set on the Convolutional Neural Network (CNN), which are greatly effective when using structured data, like time-series signals (1D-CNN), images (2D-CNN) or 3D signals (3D-CNN): the latter case is treated in [63], where the authors have computed gas-liquid flowrates using data collected by a wire-mesh sensor, a tool able to collect 3D flow information, as input for an NN. In this case, due to the particular nature of the signal, different type of NN architectures have been used, including 3D-CNN and Long-Short Time Memory (LSTM). In [268], instead, the authors have implemented a 1D-CNN in order to estimate the GVF: in this case, input signals were flowrate parameters (differential pressure signals and pressure and temperature signals, which provide gas density) collected from a Venturi meter in a 5 min experiment. In [117], the authors applied a CNN to predict the flow of gas-liquid multiphase flow in different regions. They also tested a modified version of a Generative Adversarial Network (GAN) that is able to improve the performance of the CNN adapting to the current flow domains. In [80] the authors implemented a method to estimate liquid and gas flow rates of two-phase air-water flow using conductance probes and neural network: in particular, the velocity of the fluid, inferred from the cross correlation of the probes signals, is used to estimate the rates, achieving a measurement accuracy error smaller than 10%.

In [123] the authors go through a description of the various Tomography technologies, like resistive (ERT), capacitive (ECT) and magnetic (EMT) tomography and describe their applications, focusing in particular on the capacitive one. The ECT is employed also in [184], where the authors used this technology to take pictures of the flow distribution, which give information on the flow regime and composition.

## 3.2 DATA-DRIVEN MODELS AND UNCERTAINTY ESTIMATION

As previously discussed in the introduction, uncertainty is a crucial issue in many real life scenarios. Providing a prediction without the associated uncertainty can be dangerous in case where the prediction is subsequently used to take important decisions [16]. The sources of uncertainty are often decomposed into two parts, the aleatoric and the epistemic [163]: the first is inherent in process at study, while the second depends on the inadequate knowledge of the model most suited to explain the data. The uncertainty is usually considered a confidence interval on the point wise inference. The confidence level associated to the confidence intervals in this work is set to 95%.

There are many tools to estimate the uncertainty like bootstrapping, quantile regression, Bayesian inference and dropout for the Neural Networks [2, 221, 247].

In the next paragraphs the data-driven models employed in this Chapter will be briefly described, namely Feed-forward Neural Network, Gaussian Process, Local Linear Forest, Random Forest. Despite the different structure, all these models have been chosen for their ability to return uncertainty estimations associated to each individual prediction.

### 3.2.1 *Feed-forward neural network*

The Neural Network (NN) is a very popular model, composed by a collection of artificial neurons linked together. Each neuron receives the input data and applies a non-linear function to the weighted sum of their inputs. In recent years many successful architectures proven to be particularly effective in the most disparate learning tasks like computer vision and natural language processing. Even though very powerful, NN suffer a variety of issues such as the data-expensive training and the difficulty to get explainable predictions. These problems tend to limit their widespread adoption in contexts where data are scarce and when decisions based on the network outcomes have a serious impact on real life. One way to mitigate and to manage the risk of taking dangerous decisions is to estimate the prediction uncertainty. There are a number of methods to estimate it from a NN like Bayesian methods and bagging [2] but for sure the most popular is by using dropout [91]. This technique was originally developed to avoid the co-adaptation of the parameters during the network training in order to reduce over-fitting and improve the generalization error. It has the great advantage of being conceptually simple, fast to implement and very cheap to compute, as a matter of fact it consists in the random shutdown of some neurons during training. In the uncertainty estimation process the dropout is activated also at inference time, leading to multiple predictions with different neurons activation for the same input datum. This allows to get a prediction distribution and therefore the prediction uncertainty. The correctness of this approach is guaranteed by Bayesian arguments [91], indeed it turns out that this procedure computes an approximation of a probabilistic deep Gaussian Process.

In this case, due to the scarcity of data, the choice relies on Feed-forward Neural Network, one of the simplest network architecture. This model is composed of multiple dense layers of decreasing size stacked one on top the other.

### 3.2.2 *Gaussian Processes*

A Gaussian Process (GP) is a *non-parametric* regression algorithm which tries to find a distribution for the target values over different possible functions that are consistent with input data [203]. Like the Bayesian Ridge Regression, the GP is a Bayesian algorithm, but while the former needs a prior information over the parameters, the latter needs a prior information over the functions it tries. This prior information is embedded in the *covariance function* or *kernel* of the function.

The most interesting advantage of GP is that, since it is a Bayesian method it returns not only the expected value of the posterior distribution, i.e. the prediction, but also the associated variance that can be used to measure the uncertainty.

As already mentioned, the choice of the kernel, when building the GP regressor, is crucial because it embeds the assumptions (*prior* information) made on the function aimed to be learned. Since a general function $f(x_1, x_2)$ that takes $x_1, x_2$ as inputs is not a kernel function, in the literature are given some covariance functions that are commonly used [75]. During the training of the best kernel, it's common use to consider the kernel as an hyper-parameter and therefore to try different combinations.

### 3.2.3 *Local Linear Forest*

Local Linear Forests (LLFs) have been introduced very recently by Friedberg in [89]: the authors point out a weakness of the RFs, that is their inability to make effective predictions in presence of smoothness in explored regression regions. In particular, a RF prediction can be expressed in Eq. 1, where the reported version is the one with adaptive weights [89]:

$$\hat{\mu}(x_0) = \frac{1}{B}\sum_{b=1}^{B}\sum_{i=1}^{n} Y_i \frac{1\{X_i \in L_b(x_0)\}}{|L_b(x_0)|} = \sum_{i=1}^{n} \alpha_i(x_0) Y_i(x_0) \tag{1}$$

where $\alpha(x_0) = \frac{1}{B}\sum_{b=1}^{B} \frac{1\{X_i \in L_b(x_0)\}}{|L_b(x_0)|}$ represents the weight given by the forest to the i-th training point when making a prediction on the new point $x_0$ and $n$ is the number of training points. From a more practical view, it is the fraction of trees where the i-th observation ends up in the same leaf as the new point $x_0$.

The improvement introduced by the LLF is to use these weights to fit a local linear regression, which is a great method to explore smooth behaviours but tends to fail in high-dimensions due to the curse of dimensionality. Therefore, LLFs are able to take the pros of both methods, predicting effectively in high dimensions with presence of smooth signals. In practice, the LLF tries to solve the minimization problem:

$$\begin{pmatrix} \hat{\mu}(x_0) \\ \hat{\theta}(x_0) \end{pmatrix} = \text{argmin}_{\mu,\theta}\left\{ \sum_{i=1}^{n} \alpha_i(x_0)\Big(Y_i - \mu(x_0) - (x_i - x_0)\theta(x_0)\Big)^2 + \lambda||\theta(x_0)||_2^2 \right\} \tag{2}$$

The general rule used by RF algorithms to split a node in two children is the so called CART split. Basically [89], naming the node to be split $P$ and a set of observations $(x_1, Y_1), \dots (x_n, Y_n)$, a candidate pair of children nodes is exploited: the mean value of $Y$ inside this nodes is computed as $\overline{Y}_1, \overline{Y}_2$. The selected children nodes among all candidate pairs are the ones that minimizes the following rule:

$$\sum_{i)X_i \in C_1} (Y_i - \overline{Y}_1)^2 + \sum_{i)X_i \in C_2} (Y_i - \overline{Y}_2)^2 \tag{3}$$

In LLF, instead, it is better to leave to the final regression step to model smooth signals: in the parent node $P$ it is run a ridge regression in order to predict the output $Y_i$:

$$\hat{Y}_i = \hat{\alpha}_P + x_i^T \hat{\beta}_P \tag{4}$$

After this step, the standard CART splitting rule is applied on the *residuals* $Y_i - \hat{Y}_i$.

### 3.2.4 *Confidence estimation for tree based methods: Infinitesimal Jackknife*

Wager developed a technique to estimate the RF and LLF prediction variance, the *Infinitesimal Jackknife*: in his paper [248] it is shown how the variance of the prediction quickly drops with the growing of the number of decision trees $B$.

In the variance estimation procedure, there are two types of noises: the sampling noise (due to randomness in data) and the *Monte Carlo noise*, due to the fact that the number of trees are not infinite but need to be approximated by a finite B, which increases the variance. In the mentioned paper it is shown that in order to reduce Monte Carlo noise at the level of sampling noise it is sufficient a $B = \Theta(n)$. Of course, the reduction holds up to a good prediction, meaning that if there is a bad prediction due to different reasons, as lots of irrelevant input features, the variance will be big despite the value chosen for B.

### 3.2.5 *Random Forests*

*Bagging* or *Bootstrap Aggregate* is a technique that allows to reduce variance when performing estimates on particularly high variance models, such as decision trees. Such trees are able to capture complex interactions in data [108], and do not suffer from high bias if the tree is deep. On the other hand, they are characterized from high prediction variance, therefore take advantage from averaging estimations coming from different trees, which helps to reduce the noise: at the same time, bias in the averaged tree is the same of the one of a single tree, since each tree is identically distributed. The variance of the trees average is:

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2 \tag{5}$$

where $\sigma^2$ is the variance of a single tree and $B$ is the number of trees in the forest. The first term comes from the fact that the trees are not necessarily independent, so a general coefficient of correlation between trees $\rho$ is employed [108]. It is therefore clear that as the number of trees B increases the second term disappears, but the first term which embeds the correlation between trees cannot be lowered.

*Random Forests* (RFs) has been introduced by Breiman in [34]: the purpose of this method is to reduce the value of the first term in Eq. 5 without increasing the second. This is done by randomly selecting a subset of total features for splitting during the tree growing.

By taking a small subset of features, the correlation between any tree couple in the forest is reduced: intuitively, if each tree uses a small set of the available features to make the prediction, it is likely that the estimation done by two different trees in the forest will be performed taking different features. Therefore according to Eq.5 this choice on $B$ helps in reducing the variance of the average.

### 3.2.6 *Metrics*

The previously discussed models will be primarily compared on their Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and 95-th percentile of the absolute errors. In case two or more models will result equivalent, they will be compared according the confidence intervals they provide. To make this comparison, two more metrics are needed: the Prediction Interval Coverage Probability (PICP) and the Mean Prediction Interval Width (MPIW) [221]. The first metric is defined as:

$$\text{PICP}_{l(x),u(x)} = \frac{1}{N}\sum_{i=1}^{N} h_i \quad \text{where } h(i) = \begin{cases} 1 \text{ if } l(x_i) \leq y_i \geq u(x_i) \\ 0 \text{ otherwise} \end{cases}$$

and it is the percentage of times the actual value is contained in the interval, while the second is the average width of the predicted interval and is defined as:

$$\text{MPIW}_{l(x),u(x)} = \frac{1}{N}\sum_{i=1}^{N} |u(x_i) - l(x_i)|$$

where $u(x)$ and $l(x)$ are respectively the upper and lower bound of the interval.

## 3.3 RESULTS

This Section shows the results of the present study and it is mainly divided into three parts: the first part describes the preprocessing phase, the second deals with the selection of the best predictive model, while the third one shows the confidence interval estimation.

As previously mentioned in Section 2, data come from an acquisition campaign that took place in ProLabNL high pressure flow loop. Data have been sampled using the experimental grid shown in Figure 3, and consist in about 300 experiments with 5 minutes of recording each and varying flow

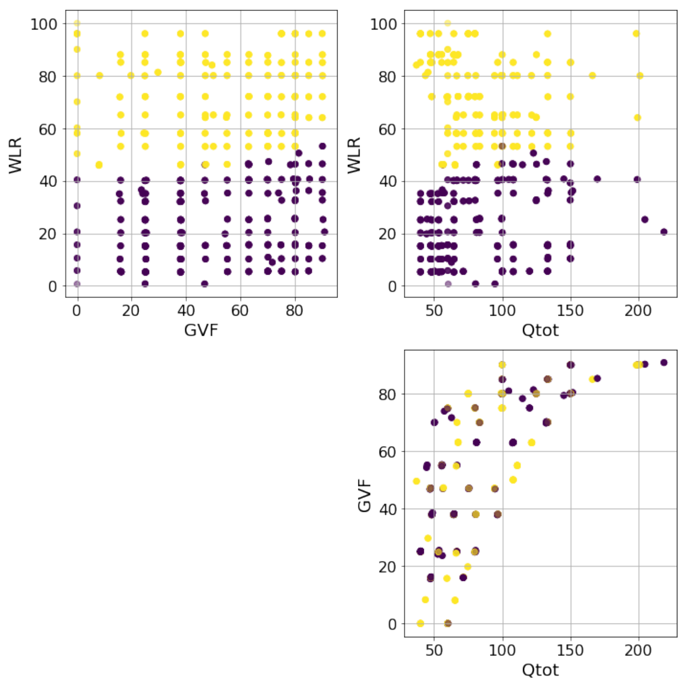
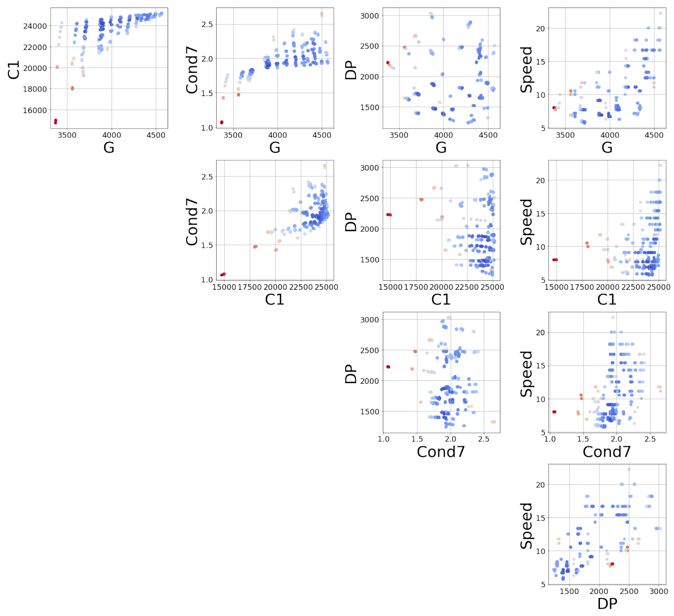


Figure 3: Experiments composing the dataset. The purple points are the *oil-continuous* experiments, while the yellow ones are the *water-continuous* experiments. WLR and GVF are percentages, while Qtot is depicted using auxiliary units.

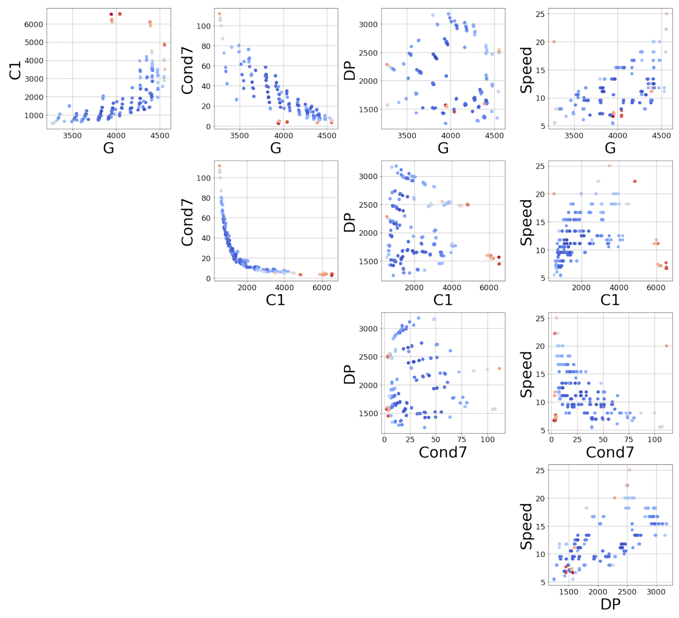
conditions. These are controlled by 3 main parameters that are *GVF*, the total flow rate $Q_{tot}$ and *WLR*, the target of the estimation model. The conditions depicted in the grid are often named *multiphase* conditions, as opposed to the *wet gas* conditions that refer to situations where the GVF exceeds 90% [104].

Depending on the fluid properties, the dataset can be divided into two parts: when the flow is mainly composed by water and the mixture is conductive the fluid is named *water-continuous*; on the contrary when the isolating fluids (oil and gas) are the majority, the fluid is named *oil-continuous* and no conductivity can be measured. The transition between these conditions can be appreciated in Figure 3, where it is clear WLR is the main parameter controlling this effect, and only weakly Qtot. Unfortunately these fluid properties need two mutually exclusive measuring instruments to be implemented, one for the conductive regime and another for the capacitive regime where no fluid conduction can measured. This leads to the development of two estimators based on the two dataset partitions, the *water* and the *oil-continuous* with respectively about 130 and 170 experiments.

The input sensors collect a stream of measurements with a high frequency rate. Features are extracted from these signals every minute and then are collected in a 5 minutes array. Once this array is full, the median value of each feature is stored and used for the flow estimation. This procedure allows to obtain robust features that are seldom corrupted by instrumentation anomalous behaviours. The extracted features are computationally efficient functions that can be easily computed on the device. An example of the stored data is depicted in Figure 4 where one feature for each sensor is shown.



(a) Oil continuous dataset.



(b) Water continuous dataset.

Figure 4: Input data and anomaly scores obtained by the Isolation Forest: the color scale goes from blue to red, where red is indicates a higher probability to be an outlier. All the above data are depicted with auxiliary units. *C1* and *Cond7* are two electrical impedance signals, *G* represents the density measured by the gamma module, *DP* is the pressure difference measured by the Venturi meter and *Speed* is the flow speed measured by cross correlating the impedance signals.

Despite the efforts to get robust fluid proprieties measures, some flow conditions are very difficult to be sampled and the resulting measures exhibits anomalous behaviours. To overcome this issue and to train and test the model on reliable data, an additional preprocessing step has been implemented. This consisted in the application of anomaly detection techniques in order to highlight experiments that may arouse the suspicious to have been measured with malfunctioning sensors. The tool employed in this context is an Isolation Forest [160], a powerful and popular method to detect in an unsupervised manner anomalous data points. The Isolation Forest is a collection of trees that randomly partition the feature space, isolating the samples. The most anomalous data tend to be isolated faster than the normal ones, therefore to be more easily detected. This algorithm have many nice properties like the fast inference time that allows this model to be implemented on board, checking if the incoming data are anomalous or not. The Figure 4 shows the application of this algorithm to the two datasets: to each data point the IF associates a score proportional to the abnormality level of the point. This procedure highlights the water continuous dataset, in particular the impedance module, exhibits severe abnormal behaviours in some experiments. To avoid unreliable results, these anomalous samples have been dropped in the last preprocessing step; on the contrary the oil continuous data are considered already sufficiently clean.

At this point the main dataset is divided into two smaller datasets containing all the features extracted by each experiment. This means in the next phase two models will be developed, one for each dataset. The selected model is the one that minimizes the Root Mean Squared Error (RMSE), the Mean Absolute Error (MAE) and the 95-th percentile of the residuals (95-th percentile) over 10 repetitions of training and testing, where the test set is composed of 30% of the randomly shuffled total dataset, paying attention not to take too close training and testing points. The proximity between data points has been defined over the output space GVF-WLR-Qtot.

As mentioned in Section 3.2, in this work 4 models that can provide confidence intervals to the estimated quantity have been tested: the Feed-forward Neural Network (FNN), the Linear Local Forest (LLF), the Gaussian Process (GP), and the Random Forest (RF). Since the GP is quite sensitive to the kernel choice, it has been tuned according a 5-fold cross validation, testing different combinations of polynomial and radial basis function kernels with different hyper-parameters. Figure 5 shows the mean results obtained by the procedure previously described, together with their standard deviation. In the oil-continuous case looking only at the mean MAE score LLF seems to be the best predictive model, however the very large error bar suggests LLF and GP might be equivalent in statistical sense. On the contrary in the water-continuous dataset the clear winner is GP with much better performances in comparison with the competitors. These results are consistent with what can be expected by the dataset size: GP has a few parameters to be tuned with respect to the other models, while RF and FNN suffer from the lack of data. Despite the similarity of LLF to RF, this method makes stronger assumptions that in this context improve its learning ability. More detailed results are shown in Table 1 and

| | FNN | GP | LLF | RF |
|---|---|---|---|---|
| 2 MAE | 14.36 | 2.94 | 7.34 | 16.89 |
| 2 RMSE | 17.60 | 3.89 | 9.66 | 19.96 |
| 95th percentile | 15.79 | 3.86 | 9.68 | 17.24 |

Table 1: Water-continuous mean results. To be consistent with the 95-percentile of the absolute error, it was reported *twice* the MAE and the RMSE.

| | FNN | GP | LLF | RF |
|---|---|---|---|---|
| 2 MAE | 5.96 | 3.02 | 2.66 | 4.97 |
| 2 RMSE | 8.13 | 5.47 | 4.27 | 7.42 |
| 95th percentile | 8.20 | 4.87 | 3.57 | 7.07 |

Table 2: Oil-continuous mean results. To be consistent with the 95-percentile of the absolute error, it was reported *twice* the MAE and the RMSE.

Table 2 where the mean results in terms of MAE, RMSE and 95-th percentile are compared. The choice to show all these scores and not just the RMSE is due to the presence of ouliers, indeed in some cases this score is not reliable and a more fair index might be MAE or 95-th percentile.

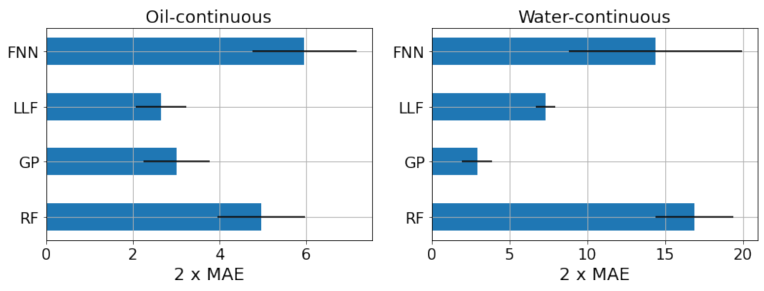


Figure 5: Results of the model selection. It is reported *twice* the MAE to be consistent with Tables 1 and 2.

The next step is to test the ability of the selected models in providing reliable confidence intervals on the two datasets. Starting from the water-continuous dataset, the GP has been trained and tested over randomly generated partitions and the results are depicted in Figures 6,7 and 8. The first Figure shows the predicted versus the actual values and the residuals versus the actual values. It confirms the good results expected in the previous test indeed the majority of the test points are close to the bisector and only few points are poorly estimated. Here it starts appearing a zone where the model struggles to predict the correct value an where its confidence intervals are wider, that is close to the transition line at 40-60% WLR. The sorted errors with their related intervals are visible in Figure 7. It is interesting here to note that on average the confidence intervals

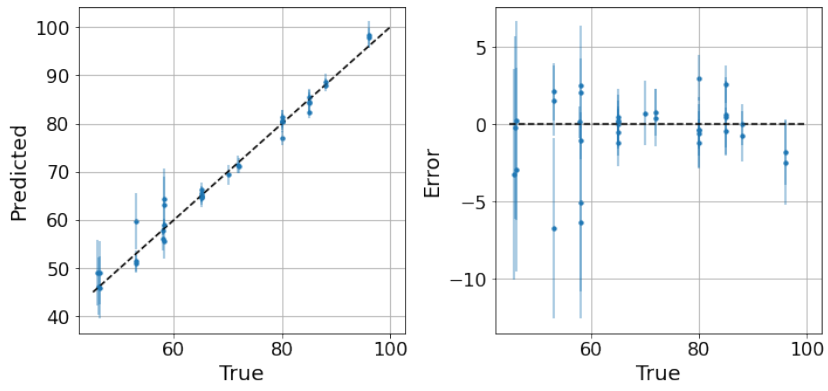


Figure 6: GP on the Water dataset. Gaussian Process predictions and related confidence intervals. 2 MAE: 3.04, 2 RMSE: 4.53, 95-th percentile: 5.41.

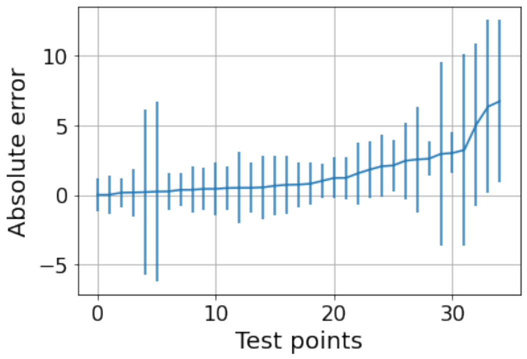


Figure 7: GP on the Water dataset. Sorted test points with their related uncertainties. The MPIW is 2.6 and the percentage of intervals crossing the zero-error line PICP is 86% compared to the expected 95%.

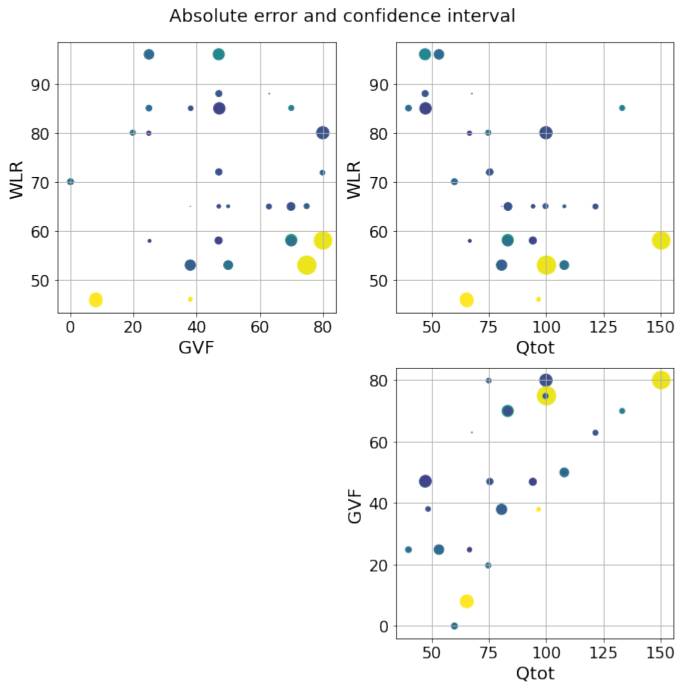


Figure 8: GP on the Water dataset. Errors and confidence intervals over the flow parameter space. The size of the dot is proportional to the error while the color is proportional to the uncertainty. The color scale goes from purple to yellow.

gets wider as the errors increase, which is an appealing effect for practitioners that may easily neglect predictions with low confidence levels. However some confidence intervals are quite large even if the prediction error is low, like in Figure 7. To this situation it is possible to find a partial explanation looking at a more complex plot in Figure 8. Here the test points are shown in the flow parameter space with size proportional to the error, and color proportional to the confidence interval. It is interesting to note the points where the prediction is less accurate are the ones close to the boundary where the model may suffer from boundary effects and the instruments are more severely challenged by the extreme working conditions. Moreover, the points close to the transition boundary around 50% WLR are the ones for which the model returns the most uncertain estimates.

Concerning the Oil dataset, as previously explained, the best model is not clear because the performance of LLF and GP are very close. To understand which model to prefer, the behaviour of the confidence intervals may be discriminating. The application of LLF to a randomly partitioned test set is shown in Figure 9, while the GP application in 10. The prediction performance of the GP are slightly worse than LLF as expected, but the great diversity is in the confidence intervals (Figures 11 and 12), indeed the LLF intervals are much shorter (1.59 on average) than the GP's (4.56 on average) but they are not proportional to the error like in the GP model; moreover the percentage of intervals crossing the zero-error line (the PICP) is quite different: 0.64 against 0.98. The choice on which model to prefer may depend on the specific

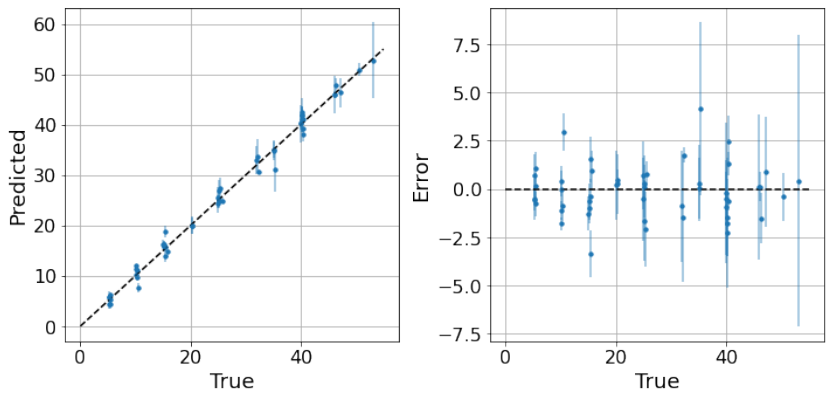


Figure 9: LLF on the Oil dataset: predictions and related confidence intervals. 2 MAE: 2.03, 2 RMSE: 2.68, 95-th percentile: 2.72.

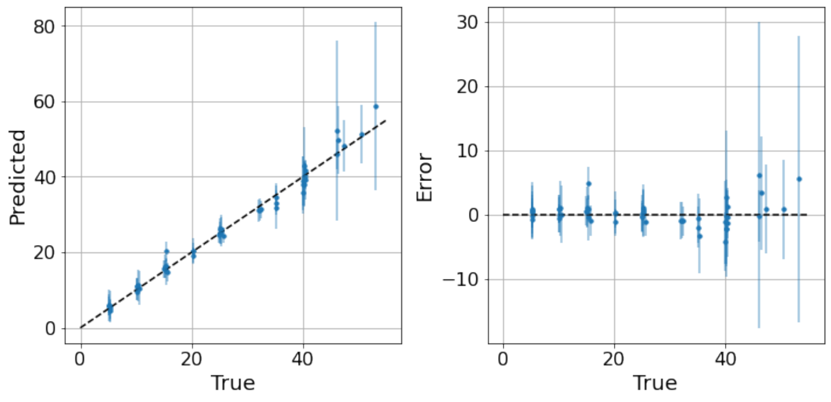


Figure 10: GP on the Oil dataset dataset: predictions and related confidence intervals. 2 MAE: 2.49, 2 RMSE: 3.74, 95-th percentile: 4.58.

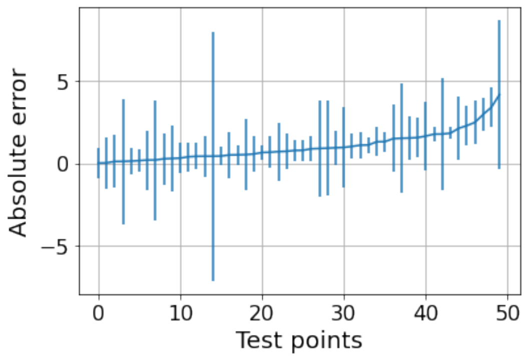


Figure 11: LLF on the Oil dataset. Sorted test points with their related confidence intervals. The MPIW is 1.59 and the percentage of intervals crossing the zero-error line PICP is 64 compared to the expected 95%.

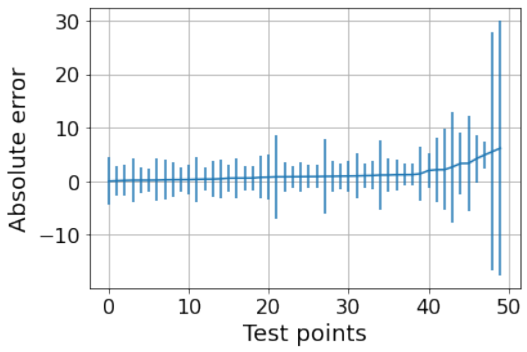


Figure 12: GP on the Oil dataset. Sorted test points with their related confidence intervals. The MPIW is 4.56 and the percentage of intervals crossing the zero-error line PICP is 98 compared to the expected 95%.

application since some applications might prefer larger confidence intervals but more certain, while others smaller intervals but less reliable. Comparing the two models in the flow parameter space it is possible to see a pattern similar to the one previously described only in the GP case (Figures 13,14). The majority of the uncertainty together with the largest errors cluster close to the transition zone. In conclusion, even if GP is less accurate than LLF it seems to have confidence intervals more in line with what can be expected from a physical model.

## 3.4 CONCLUSIONS

This study applied uncertainty quantification techniques to the estimation of extraction performance in the Oil & Gas production. The extraction performance depends on the three indicators that summarise the liquid fraction, the gas fraction and the total flow of the extracted mixture of oil, gas and water. This work mainly focused on the Water Liquid Ratio, i.e. the fraction of water inside the total volume of liquid, since it is the most important and most

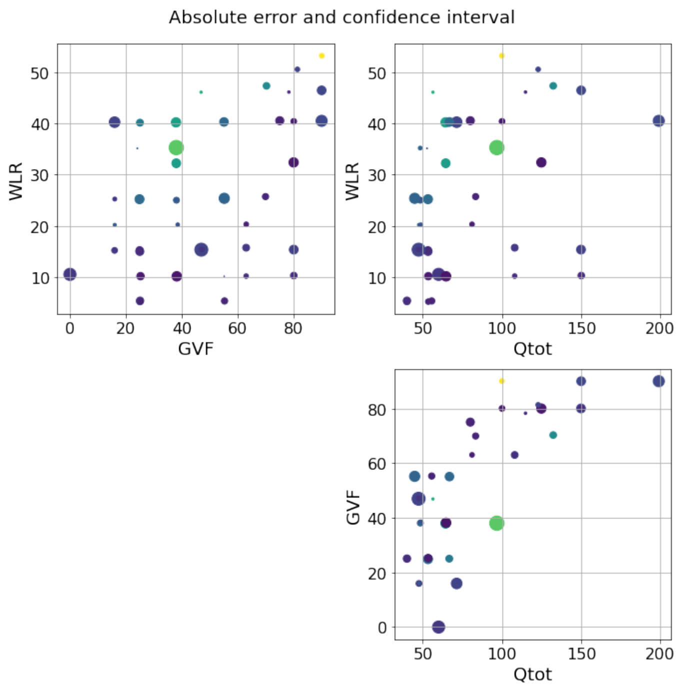


Figure 13: LLF on the Oil dataset. Errors and confidence intervals over the flow parameter space. The size of the dot is proportional to the error while the color is proportional to the uncertainty. The color scale goes from purple to yellow.

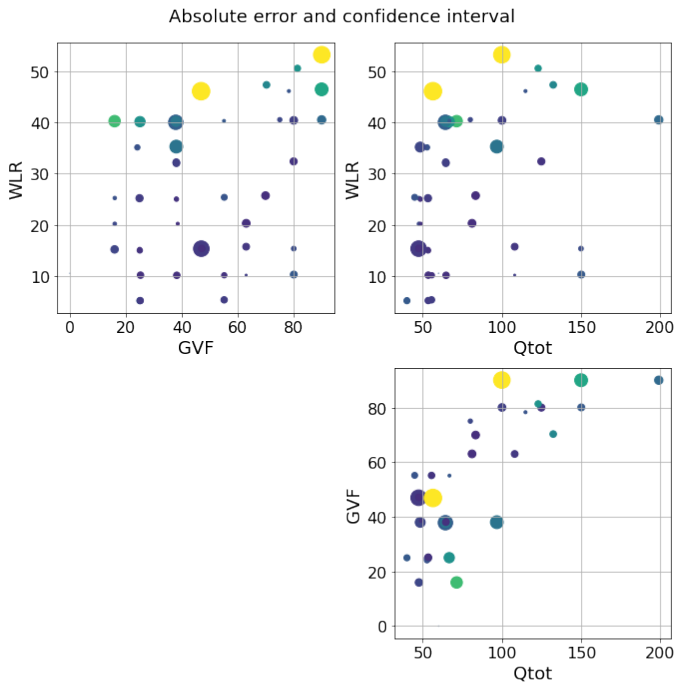


Figure 14: GP on the Oil dataset. Errors and confidence intervals over the flow parameter space. The size of the dot is proportional to the error while the color is proportional to the uncertainty. The color scale goes from purple to yellow.

complex among the discussed parameters when the final target is the extraction efficiency.

Four models have been tested to understand their behaviour on this particular context, both in terms of prediction accuracy and uncertainty estimation. These models are Feed-forward Neural Networks, Gaussian Process, Linear Local Forest and Random Forests and were applied to two complementary datasets previously pre-processed using the anomaly detection algorithm Isolation Forest. These data come from a particular instrument named Multiphase Flow Meter, that is used to extract information from the flow and to provide the extraction parameters.

The results show that most of the time the Gaussian Process is the preferable model since it has a low error compared to the other methods, and good ability in handling uncertainty. In fact in this context it returned confidence interval proportional to the prediction error. Moreover it estimated uncertainties that are in accordance with what domain experts would expect: when data are more uncertain, like when the flow is highly non-stationary in the transition regions between two flow conditions, it returns a higher uncertainty. However another model named Linear Local Forest proven its potential in making accurate predictions with few data. The main differences between these two models resulted in the confidence intervals: while Linear Local Forest has shorter confidence intervals at the cost of excluding more often the true value, Gaussian Process tends to have wider uncertainties.

Future works should focus on the data collection indeed it is likely that with more data the performances of all methods will improve, especially Random Forests and Neural Networks. In this context, one research direction that the authors are currently investigating is the application of Active Learning algorithms that are able to lessen the need for big dataset at the same time improving the predictions, like the ones exposed in Chapter 7.

# 4 ANOMALY DETECTION AND TREE-BASED METHODS

Another approach to enhance the reliability of the provided estimates is to monitor the sensor readings that are then fed into the model that estimates the production parameters. This was achieved using techniques called (sometimes interchangeably) Anomaly Detection (AD) or Fault Detection (FD), that have the goal to automatically find data not conforming to the expected behaviour. There exists many kind of anomalies depending on the data at hand: data might assume different formats like tables, streams, or images; the user might be interested on local, global, collective or context anomalies. More importantly, anomalies might assume different meanings depending on the context where they have been detected and the user expectations: what can be considered anomalous in one context, might be normal in another. In this Chapter we will describe the anomaly detection task and then we will focus on a particular class of models that originates from the Isolation Forest algorithm. This work has been published in [26].

## 4.1 INTRODUCTION

The problem of finding novel or anomalous behaviours, often referred as Anomaly or outlier Detection (AD), is common to many contexts: anomalies may be very critical in many circumstances that affect our everyday life, in contexts like cybersecurity, fraud detection and fake news [126, 165]. In science Anomaly Detection tasks can be found in many areas, from astronomy [174] to health care [179]; moreover, Anomaly Detection approaches have been even applied to knowledge discovery [60] and environmental sensor networks [110].

One of the areas that mostly benefits from the employment of AD modules is the industrial sector, where quality is a key driver of performance and success of productions and products. With the advent of the Industry 4.0 paradigm, factories and industrial equipment are generating more and more data that are hard to be fully monitored with traditional approaches; on the other hand, such availability of data can be exploited for enhanced quality assessment and monitoring [164, 225]. Moreover, products and devices, thanks to advancements in electronics and the advent of the Internet of Things, are increasingly equipped with sensors and systems that give them new capabilities, like for example the ability to check their health status thanks to embedded/cloud Anomaly Detection modules [15, 135, 252].

Despite the heterogeneous systems that may benefit from AD modules/-capabilities, there are several desiderata that are typically requested for an AD module, since obviously looking for high detection accuracy is not the only important requisite. For example, in many contexts the delay between the occurred anomaly and its detection might be critical and the low latency of the model becomes a stringent requirement. One way to mitigate the detection delay problem is typically to embed the AD model in the equipment/device: this implementation scenario directly affects both the choice of the detection model and, in some cases, the hardware. As a consequence practitioners will typically have to find a compromise between detection accuracy and computational complexity of the model: while this is true for any Machine Learning module, it is typically more critical when dealing with AD tasks. Moreover, in presence of complex processes or products equipped with different sensors, data will exhibit high dimensionality and therefore practitioners will tend to prefer models that are able to efficiently handle multiple inputs. Summarizing, a good real-world AD solution: i) has to provide high detection performances; ii) has to guarantee low latency; iii) requires low computational resources; iv) should be able to efficiently handle high dimensional data.

The former list of desiderata for AD solutions is not exhaustive, and other characteristics can increase the model appeal in front of practitioners. In recent years, model interpretability is an increasingly appreciated property. Detecting an anomaly is becoming no longer enough and providing a reason, why a point has been labelled as anomalous, is getting more and more importance. This is particularly true in manufacturing processes where the capability of quickly finding the root cause of an anomalous behaviour can lead to important savings both in terms of time and costs. In addition, interpretability enhances the trust of users in the model, leading to widespread adoption of the AD solution.

Another attractive property is the ability of the model to handle data coming from non-stationary environments. Especially in early stages of AD adoption, available data are few and restricted to a small subset of possible system configurations; a model trained on such data risks to label as anomalous all the states not covered in the training domain, even if they are perfectly normal. In order to overcome this issue, the model should detect the changes in the underling distribution and continuously learn new *normal* data. In this process, however, the AD model should not lose its ability to detect anomalies.

Given the importance and diffusion of AD approaches, we deemed relevant to review and compare an important class of algorithms particularly suited for the aforementioned requirements. The subject of this investigation is the tree-based approaches to AD, i.e. approaches that have a tree structure in their decision making evaluation; the most popular representative of this class is the famous Isolation Forest algorithm, originally proposed by Liu [160, 162], an algorithm that is receiving increasing attention and sees application in many scenarios. In this work, for the first time to our knowledge, we try to systematically review all the AD methods in this emerging class, to discuss their costs and performances in benchmarks, to report industrial applications and to guide readers through available implementations and popularity of the various approaches in the scientific literature.

This review is conceived both for researchers and practitioners. The first ones will find a comparison between the many proposed variants, while the second ones will find useful information for practical implementation. Despite this work mainly copes with AD algorithms designed for tabular datasets, it is important to note that AD can be performed on a variety of data structures like images or audio signals and algorithms dedicated for different types of data format are also present in the literature. Nevertheless, it should be remarked that any data structures can be transformed into tabular data extracting appropriated features, making AD approaches for tabular data applicable potentially to any scenario.

## 4.2 TAXONOMY AND APPROACHES TO ANOMALY DETECTION

While many Anomaly Detection approaches have been developed, a simple taxonomy divides such methods into two categories:

- *Model-based* - It is the most traditional Anomaly Detection category and employs a predefined model that describes the normal or *all* the possible anomalous operating conditions. These approaches usually rely on physics or domain-knowledge heuristics; unfortunately they are often unfeasible and costly to be developed since they require extended knowledge of the system under exam.
- *Data-driven* - The approaches examined in this thesis rely instead on data availability. More precisely, such approaches make use of two ingredients: data sampled from the analyzed system and algorithms able to automatically learn the abnormality level of those data. Such category of approaches is often referred as data-driven anomaly detection and its advantages are the great flexibility and absence of strong assumptions that limit the model applicability.

Additionally, when looking at anomalous behaviours, two problem settings can be defined: the supervised and the unsupervised one. The former consists in classifying data, based on previously tagged anomalies. It is called supervised since training data are collected from sensor measurements and have an associated label that identifies them as anomalous or not; in industrial context, the supervised scenario is typically named Fault Detection. Unfortunately, supervised settings are seldom available in reality [44]: labelling procedures are very time consuming and typically require domain experts to be involved.

On the contrary, the unsupervised scenario is the most common in real world applications. In this case, data are not equipped with labels and therefore the learning algorithm lacks a ground truth of what is anomalous and what is not. Given that, the goal of the algorithm is to highlight the most abnormal data, assigning to each one an Anomaly Score (AS). In this Chapter the focus will be on the unsupervised setting since it is the most applicable in real-world scenarios.

To be more precise, the unsupervised setting can be further divided into two sub-categories, based on the nature of the available data. The fully unsupervised one relies on training set composed of both anomalies and normal data. However in some applications obtaining training data with anomalies is quite complex, therefore in such cases data are composed only of normal instances: in this scenario semi-supervised approaches, sometimes named one-class setting, are the most natural ones to be adopted.

### 4.2.1 *Formal definition of anomaly*

The definition of anomaly is far from trivial, and depending on it, methods that try to detect anomalous data behave differently. The most widely accepted definition is quite general, and it was given by Hawkins in [109]:

> *"Observation which deviates so much from other observations as to arouse suspicion it was generated by a different mechanism".*

This statement can refer to multiple different anomalies, and does not give a clear indication on which way to follow to detect them. According to this definition it may be inferred that: i) a model needs to measure (in a not-specified way) the deviation between points; ii) each observation has an associated probability to be an outlier; iii) a different mechanism is present in the case of anomalous samples, suggesting that, on a data perspective, outliers follow a distribution that is different from the one of the inliers. The reported definition does not speak about the numerosity of outliers, but there is an hint that outliers are fewer than inliers in number. These indications give wide space to interpretations and as it will be clear later on, they encourage very different approaches.

Anomalies are traditionally divided into 4 categories even if some authors suggest different classes. Generic datasets looks like Figure 15: they are made up of normal dense and sparse clusters, surrounded by global sparse and dense anomalies. These can be defined global anomalies if they look anomalous w.r.t all the normal points, or local if their abnormality is w.r.t a single normal cluster.

### 4.2.2 *Static and dynamic problems*

Depending on the application and the available data, different problem statements can be defined. The anomaly detection problem is defined *static* if the analysis is performed on time-independent data (*static* datasets, where the order of the observations do not matter), while it is *dynamic* if it is performed on time-dependent data (time-series data or dynamic datasets). Another very basic discrimination is between univariate or multivariate anomaly detection.

A more subtle distinction concerns the way in which the algorithm training is performed. The most traditional one is the batch training where the model is trained only once, using all the available training data. This approach might be unpractical when the dataset is too large to fit into the memory, or when sampled data do not cover sufficiently well the normal operating domain: in this case the model needs to be continuously updated as new data are collected; this is even more important in situations where the normal data distribution undergoes a drift, and samples that were used in the first part of the training

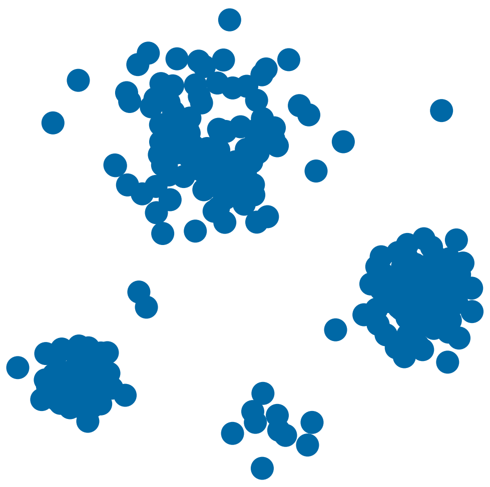


Figure 15: Example of anomalies in a simple dataset composed of dense and scattered normal clusters. Anomalies can be locally defined w.r.t a specific cluster, or global. They can be scattered in the domain or they can group together in anomalous clusters. Figure adapted from [18].

now become anomalous. Therefore in such case, the model has to adapt by learning the new configuration and by forgetting the outdated distribution.

### 4.2.3 *Classes of algorithms*

Depending on the interpretation that each author gives to the definition in Section 4.2.1, there exists different ways to measure anomaly. The only thing that brings together all the approaches is the use of an Anomaly Score (AS) assigned to each point. This score should serve as a proxy of the probability to be an outlier. Obviously, each method assigns a different anomaly score to the same point since it is based on different detection strategies.

There exists a variety of anomaly detection algorithms, but they can be categorized into 5 classes. The distinction between classes is not strongly fixed, and some methods could be categorized at the same time in different classes. The most intuitive class of algorithms is the *distance-based* one. It is based on the assumption that outliers are spatially far from the rest of the points [13]. Also the *density-based* class is quite intuitive since assumes outliers living in rarefied areas [100]. The *statistics-based* ones are conceptually simple, but often make use of heavy assumptions on the distribution that generated training data [120]. *Clustering-based* employ clustering techniques in order to find clustered data, moreover, they are strongly susceptible to hyper-parameters and often rely on density or distance measures [125].

Unfortunately, these approaches are expensive to compute or rely on too strong assumptions. Density and distance-based methods are hard to compute especially in high dimensional settings and when many data are available; such approaches are hardly applicable in scenarios where detection has to be performed online and on fast evolving data streams. Moreover, statistical methods are often restricted to ideal processes, rarely observed in practice.

Quite recently a new class of methods emerged: the *isolation-based*. This class is very different from the previous ones: it assumes that outliers are few, different and, above all, easier to be separated from the rest of data. This draws the attention from normal data to anomalies and allows to obtain much more efficient anomaly detectors. The primary goal of these methods is to quickly model the anomalies by isolating them, rather than spending resources on the modeling of the normal distribution. The seminal, and most popular, approach in this class is the Isolation Forest algorithm [160] that will be extensively discussed in Section 4.3. The original idea is based on tree-methods, but it has been recently extended to Nearest Neighbors algorithm [24].

### 4.2.4 *Tree-based methods*

Tree-based methods, as suggested by the class name, rely on tree structures where the domain of the available data is recursively split in a hierarchical way, in non-overlapping intervals named leaves. In the Anomaly Detection context, these models are seen as a tool rather than a separate detection approach. As a matter of fact, they are employed in both density-based and isolation-based approaches. The former approach is perhaps the most intuitive since at high densities one expects normal data clusters, vice-versa in low density regions one expects anomalous data. However this approach is in contrast with the simple principle of never solving a more difficult process than is needed [194]. Density estimation is a computationally expensive task since it focuses on normal data points, that are the majority. However, the ultimate goal of anomaly detection problem is to find anomalies, not to model normal data. Outliers are inferenced only at a second step. On the contrary, the isolation approach is less simple to formulate but also less computationally expensive. It directly addresses the detection of anomalies since points that are quickly isolated are more likely to be outliers.

The literature concerning anomaly detection using tree-based methods is quite vast, but in the present review we decided to focus on methods that naturally apply to the unsupervised setting, due to its relevance in industry. Not only we decided to exclude the supervised approaches, but also we excluded the ones that artificially create a second class of outliers like in [64]. These approaches often try to fit supervised models into unsupervised settings at cost of inefficient computations, especially at high dimensionality.

### 4.2.5 *Structure of the Chapter and contribution*

The aim of this Chapter will be to describe the most salient features of unsupervised tree-based algorithms for AD. Great attention will be devoted to the algorithms that primarily try to isolate anomalies, and then, as a by-product, estimate density. As stated above, to the best of our knowledge this is the first work that reviews the methods that originated from Isolation Forest, or that are closely related to it.

Throughout this Chapter, we will refer to the Isolation Forest algorithm [160] as the *original* algorithm, or by using the acronym IF. Moreover, the term *outlier* and *anomaly* will be considered synonyms. Normal data will be often named inliers and must not be confused with Gaussian data.

This Chapter is divided into 5 sections. After the first two introductory sections, the third reviews the tree-based approaches: in the first part of such section the Isolation Forest original algorithm is extensively discussed together with all the variants applied to time independent datasets; this part prepares the ground for the more complex time dependent datasets and their algorithms presented in Section 4.3.2. After this, some paragraphs are devoted to distributed and interpretable models. The fourth section compares the performances of the methods, looking at the results declared in the papers. A practical comparison between the methods fall outside the scope of this Chapter, but case studies and multiple source code repositories are listed for the interested reader. In the last Section, conclusions and ideas for future research directions are discussed.

## 4.3 ISOLATION FOREST AND TREE BASED APPROACHES

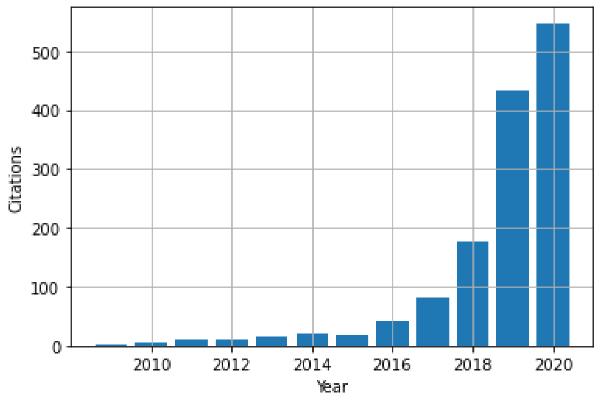


Figure 16: Combined citations of IF original paper [160] and its extended version [162] by the same authors. Source: Scopus. Retrieved on the 30th of March 2021.

Isolation Forest (IF) is the seminal algorithm in the field of isolation tree-based approaches and it was firstly described in [160]: in recent years IF has received an increasing attention from researchers and practitioners as it can be noted in Figure 16, where the evolution of citations of the algorithm in scientific papers has increased exponentially over the years.

As the name suggests, IF is an ensemble algorithm that resembles in some aspects the popular Random Forest algorithm revised in the unsupervised anomaly detection settings. Indeed, IF is a collection of binary trees: while in Random Forest (RF) [34] we are dealing with decision trees, here the ensemble model is composed by *isolation trees*, that aim at isolating a region of the space where only a data point lies. IF is based on the idea that, since anomalies are by definition few in numbers, an isolation procedure will be faster in separating an outlier from the rest of the data than when dealing with inliers.

More in details, the algorithm consists in two steps: training and testing. In the training phase, each isolation tree recursively splits data into random partitions of the domain. As said, the core idea is that anomalies on average require less partitions to be isolated. Therefore inliers live in a leaf in the deepest part of the tree, while outliers in a leaf close to the root. More formally, the anomaly score is proportional to the average depth of the leaf where each datum lies. For the sake of clarity, we report the training and testing pseudo-codes (Algorithm 1, 2 and 3 - adapted from [160]).

```
Algorithm 1: IsolationForest($X$, $n_T$, $\psi$)

Input: $X$ – data in $\mathbb{R}^d$, $n_T$ – number of trees, $\psi$ – sample size
Output: list of Isolation Trees
$forest \leftarrow$ empty list of size $n_T$;
$h_{max} \leftarrow \lceil \log_2 |X| \rceil$;
for $i = 1$ to $n_T$ do
    $\hat{X} \leftarrow sample(X, \psi)$;
    $forest[i] \leftarrow IsolationTree(\hat{X}, 0, h_{\max})$;
end
return forest
```

```
Algorithm 2: IsolationTree(X, h, h_max)

Input: $X$ – data in $\mathbb{R}^d$, $h$ – current depth of the tree, $h_{\max}$ – depth limit
Output: Isolation Tree (root node)
if $h \geq h_{max}$ or $|X| \geq 1$ then
    return Leaf{
        $size \leftarrow |X|$
    };
else
    $q \leftarrow$ randomly select a dimension from $\{1, 2, \ldots, d\}$;
    $p \leftarrow$ randomly select a threshold from $[\min X^{(q)}, \max X^{(q)}]$;
    $X_L \leftarrow filter(X, X^{(q)} \geq p)$;
    $X_R \leftarrow filter(X, X^{(q)} < p)$;
    return Node{
        $left \leftarrow IsolationTree(X_L, h+1, h_{max})$,
        $right \leftarrow IsolationTree(X_R, h+1, h_{max})$,
        $split\_dim \leftarrow q$,
        $split\_thresh \leftarrow p$
    };
end
```

The training phase starts subsampling the dataset composed of $n$ data points, in $n_T$ randomly drawn subsets of $\psi$ samples. Then, for each subset a random tree is built. At each node of the random tree a feature is uniformly drawn. The split point is uniformly sampled between the minimum and maximum value of the data along the selected feature, while the split criterion is simply the inequality w.r.t the feature split point. The splitting procedure is recursively performed until a specific number of points are isolated or when a specific tree depth is reached. In principle, the full isolation tree should grow until all points are separated, unfortunately risking to grow trees with depth close to $n-1$. However data that lie deep in the tree are the normal ones, not the target of the detector. For efficiency reasons, this is not practical and since anomalies are easy to be isolated, the tree only needs to reach its average depth $\lceil \log_2 n \rceil$.

**Algorithm 3:** PathLength(x, T, h)

```
Input: $x$ – instance in $\mathbb{R}^d$, $T$ – node of IsolationTree, $l$ – current length
        (to be initialized to 0 when first called on the root node)
Output: path length of x
if $T$ is a leaf node then
    return $h + c(T.size)$;
end
$q \leftarrow T.split\_dim$;
if $x^{(q)} < T.split\_tresh$ then
    return $PathLength(x, T.left, l+1)$;
else
    return $PathLength(x, T.right, l+1)$;
end
```

The testing phase is different and consists in checking the depth $h(\cdot)$ reached by the data point $x$ in all the isolation trees, and taking the average.

The anomaly score $s(x,n)$ is defined as:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}}$$

where $c(n)$ is a normalizing factor and $E(h(x))$ is the average of the tree depths. Note that when $x$ is an anomaly, $E(h(x)) \rightarrow 0$ and therefore $s(x) \rightarrow 1$, while when $E(h(x)) \rightarrow n-1$, $s(x) \rightarrow 0$. When $E(h(x)) \rightarrow c(n)$, $s(x) \rightarrow 0.5$.

The computational complexity of IF is $O(n_T \psi \log \psi)$ in training while $O(n n_T \psi \log \psi)$ in testing, where we recall that $\psi$ is the subsampling size of the dataset. It is interesting to note that in order to have better detection results, $\psi$ needs to be small and constant across different datasets.

Isolation forest has many advantages compared to the methods belonging to other classes. Firstly it is very intuitive and requires a small amount of computations. For this reason it is particularly suited for big datasets and for applications where low latency is a strict requirement. The use of random feature selection and bagging allows to efficiently handle high dimensional datasets. In addition, the use of tree collections makes the method highly parallelisable. Unfortunately the algorithm has some issues. The most severe is the *masking effect* created by the axis parallel partitions and anomalous clusters, that perturbs the anomaly score of some points. A closely related issue is the algorithm difficulty to detect the local anomalies.

Trying to make a summary: the standard isolation forest defines anomalies as few and different, and approaches their detection not modelling normal data but trying to separate them as fast as possible with the aid of bagged trees. The split criterion is based on randomly selected feature and split point, that create axis-parallel partitions. These characteristics will be challenged by the following methods but the structure of the algorithms will remain quite the same.

In the following subsections the focus will be on static approaches (Section 4.3.1), dynamic (Section 4.3.2), distributed (Section 4.3.3) and finally interpretable and feature selection methods (Section 4.3.4).

### 4.3.1 *Static learning*

Static learning methods can be generally divided into two sub-categories, using two approaches. The first one groups i) methods that directly originate from the seminal work [160] slightly modifying the Isolation Forest, and ii) methods that start from a different but similar algorithms like the Half-Space (HS) trees or Random Forests (RF). The former group focuses on the importance of fast isolation, while the latter on the density approximation. The second grouping approach subdivides the static methods based on how is computed the anomaly score. The majority relies on the mean leaf depth but a growing number of algorithms employs some variation of the leaf mass.

**FILTERING AND REFINEMENT: A TWO-STAGE APPROACH FOR EFFICIENT AND EFFECTIVE ANOMALY DETECTION** The general intuition that an AD algorithm should focus more on the detection of anomalies than the modelisation of normal data, has been developed also in [265]. In this case, the algorithms is based on two stages: filtering and refinement. At the first stage, the majority of normal data are filtered using a computationally cheap algorithm, while at the second stage, the remaining data are processed by a more refined but expensive tool. Filtering is performed by a tree based method quite similar to IF except for the splitting criterion: here the choice is deterministic and based on feature entropy and univariate densities computed by histograms. After that, distance-based methods are applied to the most abnormal data points and two anomaly scores are proposed to detect sparse global anomalies and clustered local anomalies.

The time complexity of the filtering stage is close to the IF, while the refinement stage is $O(s^2)$ where $s$ is the number of filtered samples. It is easy to see that the filtering stage is very important to get competitive computational performances.

**ON DETECTING CLUSTERED ANOMALIES USING SCIFOREST (SCIFOREST)** SCiForest [161] takes the assumptions of IF and tries to improve it, with special attention to clustered anomalies. Indeed Isolation Forest performs quite poorly on them. The two most important novelties that this method introduces are: i) the use of oblique hyper-planes, instead of axis-aligned, and

ii) the use of a split criterion that replaces the random split. At each partition multiple hyper-planes are generated, but only the one that maximises a certain criterion is selected.

The intuition behind this criterion is that clustered anomalies have their own distribution and the optimal split separates normal and anomalous distributions minimizing the dispersion. Therefore the split criterion is formalized as:

$$\mathrm{Sd}_{\mathrm{gain}} = \frac{\mathrm{std}(X) - \mathrm{average}\big(\mathrm{std}(X^{\mathrm{left}}), \mathrm{std}(X^{\mathrm{right}})\big)}{\mathrm{std}(X)}$$

where $\mathrm{std}(\cdot)$ computes the standard deviation.

Due to the new computations, the complexity of the SCiForest increases reaching $O(n_T\tau\psi(q\psi + \log\psi + \psi))$ in the training stage, and $O(qnn_T\psi)$ in the evaluation stage, where $t$ is the number of trees in the forest, $\tau$ the number of hyper-plane trials and $q$ the number of features composing each hyper-plane dimensions.

**MASS ESTIMATION (MASSAD)** The authors of Mass estimation (MassAD) started in their papers [241, 242] recalling the classic definition of the mass i.e. the number of points in a region. However their definition of the mass of a point is slightly more complex since they consider all the overlapping regions that cover that point. Doing that, they obtain a family of functions that accentuates the fringe points in a data cloud. Despite its usefulness, this sort of anomaly score is too computationally expensive. To overcome this issue they propose an algorithm that approximates it, employing Half Space trees. These can be thought as a simplification of isolation trees, indeed only the splitting feature is taken at random, while the splitting value is half of the range along that feature. They propose two variants of the same algorithm: one grows leaves of the same depth, while the other lets them to have different depths. The latter not only estimates the score using the leaf mass, but also improves it with a factor dependent on the tree depth.

The authors report a time complexity $O(n_T(\psi + n)h)$ that includes both training and testing. The space complexity is $O(t\psi h)$ .

**ORDINAL ISOLATION: AN EFFICIENT AND EFFECTIVE INTELLIGENT OUTLIER DETECTION ALGORITHM (KPLIST)** The method proposed in [50] eliminates the randomness of isolation forest, partitioning the space by means of successive uniform grids and at each depth the grid doubles its resolution.

The time complexity is $O(n \log n)$.

**IMPROVING IFOREST WITH RELATIVE MASS (REMASS IF)** ReMass IF [18] starts from quite similar premises to the SCiForest's, but focuses on the poor performances of IF on local anomalies. Unlike SCiForest, it does not suggest to modify the training algorithms but the anomaly score: it proposes to substitute the tree depth with a new function, the *relative mass*.

The mass of a leaf $m(\cdot)$ is defined as the number of data points inside the leaf while the relative mass of the leaf is the ratio between the mass of the parent and the mass of the leaf. More precisely, the anomaly score for each tree is defined in this way:

$$s_i(x) = \frac{1}{\psi} \frac{m(X_{\text{parent}})}{m(X_{\text{leaf}})}$$

The authors suggest to modify only the anomaly score formula, keeping the rest of the algorithm untouched. This helps improving the anomaly score, while preserving the low computational complexity.

The time and space complexities are the same as IF.

**EXTENDED ISOLATION FOREST (EIF)** In the paper [107] the authors observe the masking effect created in the IF algorithm by the axis-aligned partitions. The intersection of multiple masks can even create some fake normal areas of the domain, leading to completely wrong anomaly detection. In order to overcome the described issue, the authors suggest a very simple but effective strategy already employed in SCiForest: the use of oblique partitions. However in this case the authors do not use a repeated split criterion, loosing its benefits but also the additional computational overload.

The time complexity is similar to the SCiForest, except for the saving of the $\tau$ repetitions.

**IDENTIFYING MASS-BASED LOCAL ANOMALIES USING BINARY SPACE PARTITIONING** The approach presented in [95] is very similar to [241] and mainly differs in the way the anomaly score is computed: it does not rely only on the mass and depth of the leaf, but is weighted by the deviation between the selected split point and the corresponding feature value of the tested point.

The authors report time and space complexities similar to MassAD, i.e. a time complexity $O(n_T h(\log n + \psi))$.

**FUNCTIONAL ISOLATION FOREST (FUNCTIONAL IF)** IF was naturally born for static data but it can also be employed for functional data. In this case anomalies can be subdivided into shift, amplitude and shape anomalies. These can be transient or persistent depending on their duration. In [223] the authors formalise this approach adapting the IF algorithm to the new setting. The set of features are not given with the dataset, but are extracted projecting the functional sample over a dictionary chosen by the user. This choice is arbitrary and highly affects the resulting performances. The projection is not performed using a classical inner product because it does not account for shape anomalies; on the contrary the authors suggest to employ the Sobolev scalar product.

**ONE CLASS SPLITTING CRITERIA FOR RANDOM FORESTS (ONECLASSRF)** The Random Forest algorithm naturally applies to the supervised setting, however some attempts to adapt it to the unsupervised one has been made. As previously discussed in the former section, the most intuitive solution is to artificially sample the domain in order to create outlier data [64], but this has been shown to be inefficient. Another approach described in [99] extends the split criterion based on the Gini index to the unsupervised scenario. Intuitively, the criterion tries to generate two children: the first that isolates the minimum number of samples in the maximum volume, while the second the contrary. In practice, the authors suggest two strategies to adapt the Gini index in absence of a second class: one considers the outliers uniformly distributed, while the other at each split estimates the outliers as a fixed fraction of inliers.

**A NOVEL ISOLATION-BASED OUTLIER DETECTION METHOD (EGITREE)** Sciforest is not the only one that tries to improve the algorithm using split criteria: EgiTree [220] (a very similar approach was presented in [155]) employs heuristics based on entropy to effectively select both attribute and split value. Indeed, the goal is to take the randomness out of the algorithm. The authors start observing that in practice, looking at the features individually, two kind of anomalies exist: anomalies that are outside the normal range, and other that are inside the normal feature range but have abnormal feature combinations. In the first case anomalous data are easier to be detected since a gap is easily identifiable, and the disorder is lower. In the second, it is difficult to find a gap simply looking at the projections of data over the axes. From these observations the authors defined two heuristics. If the feature that exhibits lower entropy has an entropy value i) less than a threshold, it is partitioned along the biggest gap between the feature data ii) greater that this threshold, it is partitioned along the mean feature value, creating a balanced partition. At each splitting iteration a partition cost is computed. When the first heuristic is used, the partition cost is roughly inversely proportional to the gap, and takes the form:

$$\text{cost}(X) = 1 - \frac{\text{maxgap}(X_{\text{feature}})}{\max(X_{\text{feature}}) - \min(X_{\text{feature}})}$$

On the contrary, when the second heuristic is employed, the partition cost is maximal and equal to 1. The total partition cost of a data point is the sum on the partition costs of each node traversed by the datapoint, and the anomaly score related to a single tree is the inverse of the total partition costs.

**LSHIFOREST: A GENERIC FRAMEWORK FOR FAST TREE ISOLATION BASED ENSEMBLE ANOMALY ANALYSIS (LSHIFOREST)** The algorithm presented in [269] combines the isolation tree approach with the Locality Sensitive Hashing (LSH) forest, where given a certain distance function $d$, neighbours samples produce the same hash with high probability while samples far from each other produce the same hash with low probability. The probabilities can be tuned by concatenating different hash functions, so that an isolation tree can be constructed by concatenating a new function at each internal node. The path from the root to a leaf node is the combined key of the corresponding data instance. Since $d$ is generic, this extension allows to incorporate any similarity measure in any data space. Moreover, the authors show that their framework easily accommodates IF and SCiForest when particular hash functions are selected. They adapted the method in this way: i) the sampling size is not fixed but variable, ii) the trees are built using the LSH functions, iii) the height limit and the normalisation factor are changed consequently and iv) the individual scores are combined after the exponential rescaling. The average-case time complexity in the training stage is $\Theta(\psi \log \psi)$, while in the evaluation phase it is $\Theta(\log \psi)$.

**HYBRID ISOLATION FOREST (HIF AND HEIF)** The authors of [177] observe that IF behaves differently if the dataset has a convex or concave shape. For example they analyze the detection performances on a dataset composed of a toroidal normal cluster and some scattered anomalies that lie inside and just outside the torus. It turns out that IF struggles to detect inner anomalies, giving them a score too close to the normality. To overcome this issue the authors propose two approaches, one of which is unsupervised: at each leaf node the centroid of leaf training data is computed and recorded, then in the testing phase the distance between the point and the corresponding centroid is measured. This new score is linearly combined with the traditional leaf depth, obtaining a more robust score. Unfortunately this approach employs euclidean distance that is not scale invariant and, therefore, requires some unpractical normalizations. In [114] the approach described in the previous paper is enhanced by the using of the Extended Isolation Forest [107], obtaining a better detector.

The authors claim the time complexity of this algorithm is slightly higher than IF due to the additional computations, but anyway comparable. To verify their hypotheses they perform some simple simulations.

**A NOVEL ANOMALY DETECTION ALGORITHM BASED ON TRIDENT TREE (T-FOREST)** The method [267] is based on a very simple tree structure: the trident tree. As the name suggests, this structure is not a binary tree but it generates three children at a time. Like IF, a random feature is selected and a split criterion is applied. The split criterion is simple: data that are three standard deviations to the left of the mean are sent to the left child, the contrary for the right child, and the part in the middle of the distribution is assigned to the central child. The anomaly score is then computed in a similar way to ReMass IF, i.e. using the mass instead of the leaf depth.

The authors report a time complexity of $O(n_T \psi \log n)$ and space complexity of $O(n_T \psi n)$ for the training phase, while the time complexity of the evaluation phase is $O(m n_T \log(n\psi))$

**HYPERSPECTRAL ANOMALY DETECTION WITH KERNEL ISOLATION FOREST** In the context of computer vision, a small modification to the original algorithm has been shown in [153]. Here the goal is to find anomalous pixels inside a hyper spectral image. A kernel is employed in order to extract non linear features to be used in the IF. Then the principal components are selected, a global method is trained on the whole image and the most anomalous pixels are detected. The connected components of anomalous pixels are subjected to a local procedure, where a new model is trained and tested on the pixels. This method is applied recursively until a sufficiently small anomalous area is detected.

The authors calculate a time complexity of $O(n_T\psi(\psi + n_{\text{pixels}}))$.

**A NOVEL ANOMALY SCORE FOR ISOLATION FORESTS** The classic mean leaf depth as proxy of the data anomaly is questioned in [180]. According to the authors, the information encoded in the structure of the original isolation tree, is not fully exploited by its anomaly score. After this premise, they suggest three different alternatives that do not change the learning algorithms but only how is computed the anomaly score. Instead of the standard path length that adds a unit at each traversed node, they propose a weighted path. They suggest multiple strategies to obtain these weights. The first relies on the concept of neighborhood: more isolated points will have smaller neighbors, therefore at each node the weight will be the inverse of data passing through it. The second strategy starts from [99] where a split criterion based on Gini impurity was employed. In this context the authors suggest to weight the node using the inverse of the split criterion value, since it measures how well the split was performed. The last strategy simply takes the product between the former two.

**DISTRIBUTION FOREST: AN ANOMALY DETECTION METHOD BASED ON ISOLATION FOREST (DFOREST)** The variant proposed in [263] does not rely on axis parallel or oblique partitions but on elliptic ones. In this case, at each node multiple random features are selected and the covariance is computed. Then data are divided using the Mahalanobis distance: points that lye inside the hyper ellipsoid are sent to the left child, while the ones outside of the elliptic boundary are sent to the right leaf. The split value is chosen such that a fixed portion of data are outside the ellipse.

The time complexity differs from the IF one in the last step of covariance computing. As the subset of selected features increases, this diversity becomes more marked.

**RESEARCH AND IMPROVEMENT OF ISOLATION FOREST IN DETECTION OF LOCAL ANOMALY POINTS (CBIF)** Multiple approaches have been proposed to overcome the limitations of IF in detecting local outliers. In [92] the authors suggest the combination of clustering based algorithms and the original IF. Despite its efficacy, the choice of using a more expensive model to enhance a cheaper one, seems counter intuitive.

**K-MEANS-BASED ISOLATION FOREST (K-MEANS IF AND N-ARY IF)** In the papers [130, 131] the authors investigate the impact of the branching process in the original isolation forest algorithm. More precisely they are interested in how the algorithm behaves changing the number of children each node has to grow. They try to improve the original algorithm by means of K-means clustering over the selected splitting feature and using a score that measures the degree of membership of the point to the each traversed node.

**OPHIFOREST: ORDER PRESERVING HASHING BASED ISOLATION FOREST FOR ROBUST AND SCALABLE ANOMALY DETECTION (OPHIFOR-**

**EST)** The work shown in [258] improves on the core ideas of the LSHiForest by proposing a learning to hash (LTH) method to select the hashing function which best preserve similarities in the dataset in the projected space. The order preserving hashing algorithm (OPH) is chosen for such task as it shows excellent performances in nearest neighbour search. This algorithm is able to find the hash function which minimizes the order alignment errors between original and projected data samples. Moreover, an improved two-phases learning process for OPH is presented to enable faster computation. An isolation forest is built based on the hashing scheme, where the specific hash function to use at each node is not random, but it is fine-learnt by OPH. Finally, the evaluation phase is similar to the LSHiForest.

While this method hash higher training time complexity ($\Theta(Hb_r\psi^2 n_T a \log\psi)$) with respect to LSHiForest due to the learning process, it shows similar performance in the evaluation phase ($\Theta(n_T na \log\psi)$).

**PIDFOREST: ANOMALY DETECTION VIA PARTIAL IDENTIFICATION (PIDFOREST)** Classical IF relies on the concept of isolation susceptibility, which intuitively can be outlined as the average number of random slices that are needed to fully isolate the target data. This definition of anomaly has some great advantages, but also some pitfalls. In particular, in high dimensional data, many attributes are likely to be irrelevant and isolation may be sometimes very demanding.

PIDForest [101] is based on an alternative definition of anomaly. The authors assert that an anomalous instance requires less descriptive information to be uniquely determined from the other data. Then, they define their partial identification score (PIDScore) in a continuous setting as a function of the maximum sparsity over all the possible cubical subregions containing the evaluated data point $x$. Say $X$ full data, and $C$ a subcube of the product space and $\rho$ a sparsity measure, PIDScore can be formalized as

$$\text{PIDScore} = \max_{x\in C} \rho(X, C) = \max_{x\in C} \frac{\text{vol}(C)}{|C \cap X|}$$

PIDForest builds a heuristic that approximates the PIDscore. The strategy is to recursively choose an attribute to be splitted in $k$ intervals, similarly to $k$-ary variants of IF (authors suggest default hyperparameter $k < 5$). Intuitively, we would like to partition the space into some sparse and some dense regions. For this purpose, a possible objective is to maximize the variance over the partitions in terms of sparsity, that can be treated as a well-studied computational problem related to histograms and admits efficient algorithms for its solution. For each attribute the optimal splits are computed and the best attribute is chosen as coordinate for partitioning. Then, the iteration is repeated on each partition, until a data point is fully isolated or a maximum depth parameter is exceeded. Now the resulting leaf is labelled with the sparsity of the related subregion. In the testing phase, a data point can be evaluated on each tree of the PIDForest and the maximum score (or a robust analog, like 75% percentile) gives an estimate of the PIDScore.

In the words of its authors, the fundamental difference between IF and PIDForest is that the latter zooms in on coordinates with higher signal, being less susceptible to irrelevant attributes at the cost of more expensive computation time. Each PIDTree takes $O(k^h d\psi \log \psi)$ as training time, while testing is pretty much equivalent to IF.

**AN OPTIMIZED COMPUTATIONAL FRAMEWORK FOR ISOLATION FOREST** As many other methods, also [167] tries to improve the stability and accuracy of IF, designing new split criteria. It starts observing that the separability of two distributions, the anomalous and normal one, is proportional to two factors: the distance between peaks and the dispersion of the distributions. The authors developed a simple index, named *separability index* roughly similar to the one described in [161] but considering also the distance between distributions a feature at a time. Another difference relies in the choice of the best splitting value: instead of trying multiple random values and taking the best out of them, an optimization procedure based on the gradient of separability function is chosen.

The time complexity is $O(kn(\log n)^2)$.

**ANOMALY DETECTION FOREST (ADF)** In [224] the authors observe that IF is specifically suited for the unsupervised setting previously discussed in Section 4.2, where the training set is composed both of normal and anomalous samples. However, in case of normal-only training data, this model does not create leaves representing the anomalous feature space, and tends to give high scores to inliers. To overcome this issue the authors introduce a new structure based on two new concepts: i) the *anomaly leaves* that should model the feature values not contained in the range of training samples, and ii) the *isolation level* that is the node size below which the anomaly leaves are created. The authors also define a special kind of internal node, named *anomaly catcher*: when the node size is less than the *isolation level* threshold, it generates a generic child and an anomaly leaf. Two kinds of split criteria are used: one for the partition of generic internal nodes, and one for the partition of anomaly leaves. The first is quite similar to the uniformly random criteria of IF, with the difference it tries to guarantee a less unbalanced split. The second generates the empty (anomaly) leaf by splitting the feature between the extreme value of the dataset and the extreme value of the node space. These modifications require a small adjustment on the anomaly score since in this new settings the original normalising factor does not make sense anymore. As a consequence the *observed* average path length has been preferred.

The time complexity in the testing phase is similar to IF, but the one in the training phase is higher due to additional sortings done in the split value computation.

**USFAD: A ROBUST ANOMALY DETECTOR BASED ON UNSUPERVISED STOCHASTIC FOREST (USFAD)** The work presented in [17] addresses the issue of different units/scales in data, starting by showing some examples where different non-linear scales lead to completely different anomalies. To solve this issue the authors propose *usfAD*, a method that combines Unsupervised Stochastic Forest (USF) with IF, and naturally born for the semi-supervised task. This hybrid model recursively splits the subsample until all the samples are isolated. However it is different from the IF since it grows balanced trees with leaf of the same depth. This is accomplished using a splitting rule that uses the median value as split point. The core idea is that the median, since it relies on ordering, is more robust to changes in scale or units. After the tree growth, normal and anomalous regions are associated to each node: the former consists in the hyper-rectangle containing the training points, while the latter is the complementary region. All these modifications lead to a quite different testing phase. The anomaly score of a test point is the depth of the first node where it falls outside the normal region.

The time complexity is slightly higher than IF: the training is $O(nn_T h + n_T 2^h d)$ while the testing $O(n_T(h+d))$. Moreover, it needs $O(n_T 2^h d)$ memory space.

**FUZZY SET-BASED ISOLATION FOREST (FUZZY IF)** Attempts to improve the IF algorithm have been made also by using Fuzzy Sets approaches [129]. The anomaly score is simply measured by the so called *degree of membership*, i.e. at each node a function of the distance between the point and the centroid is incrementally added.

**INTEGRATED LEARNING METHOD FOR ANOMALY DETECTION COMBINING KLSH AND ISOLATION PRINCIPLES** In the paper [200] a method based on Kernelized Locality-Sensitive Hashing (KLSH) combined with IF is proposed, with the aim of improving the detection of local anomalies. A gaussian kernel function is used to map features to a higher dimensional feature space to map local anomalies in the original space into global anomalies, which are easy to isolate and detect. IF is then used to isolate anomalies in the kernelized dataset. Furtheremore, two improvements on IF are proposed: a random non-repeating subsampling technique and a mean optimization strategy to optimally select the segmentation attributes and values.

**RMHSFOREST: RELATIVE MASS AND HALF-SPACE TREE BASED FOREST FOR ANOMALY DETECTION (RMHSFOREST)** The algorithm presented in [171] tries to combine the advantages of the Half-Space tree described also in [242] and the anomaly score proposed in the ReMass-IF [18] algorithm. In this context the authors employ Half-Spaces for the tree construction and modify the ReMass score function adding the depth and taking a logarithmic function of the relative mass.

The time and space complexity are respectively $O(n_T(\psi + n)h)$ and $O(n_T \psi h)$.

**ANOMALY DETECTION BY USING RANDOM PROJECTION FOREST (RPF)** Many works in anomaly detection with tree-based methods still refer to Random Forests. One of them is [49] where a revised splitting rule based on the Kullback-Leibler divergence and oblique projections instead of axis aligned are used to model the dataset density distribution.

**RANDOMIZED OUTLIER DETECTION WITH TREES (GIF)** The method proposed in [38] focuses on two aspects. Firstly it proposes a theoretical framework that interprets the isolation forest variants from a distributional point of view. More precisely, it interprets isolation as a density estimation heuristic in which the algorithm reckons the weights of a mixture distribution, where the dominant component characterizes normal data, while the minor ones can be considered as anomalous. The authors conclude that any tree-based algorithm with sufficiently many fine-grained splits can guarantee some approximation quality of the underlying probability distribution.

Afterwards, starting from these premises a new method is developed, named Generalized Isolation Forest. The proposed algorithm makes use of non-binary partitions and the data are divided based on the maximization of a custom inner kernel function, in order to produce regions that are small and dense enough, as the theoretical dissertation suggests. As in the original IF, the tree is not required to be fully grown, but the partitioning process stops when a sufficient level of the distribution approximation is reached. Then, a density function, like frequency of observations, is used in testing phase, instead of path length.

Despite its name, GIF turns to be pretty much different than original IF. Nevertheless, it promises interesting performances and a solid foundation, the downside resides on the need of fine-tuning of many hyperparameters and an arguably expensive computational time for big datasets.

**PIF: ANOMALY DETECTION VIA PREFERENCE EMBEDDING (PIF)** As discussed above, anomaly definition varies across the papers: in this approach [151] the authors point out the difference between general statistical anomalies and pattern anomalies (Figure 17). The former are typically defined as samples falling in regions where the density is low, while the latter are samples that deviate from some structured pattern. Finding and fitting these structures to find the anomalies is extremely expensive, therefore the authors suggest a new method, named Preference IF, to directly tackle the problem.

The proposed method consists mainly in two steps: i) the embedding of data in a new space, named preference space and ii) the adoption of tree based isolation approach specifically suited for this new space. In particular, the adoption of a nested Voronoi tessellation and the Tanimoto distance allows much better performances than simply using the original IF in the preference space.

The complexity of this algorithm is $O(\psi n_T b \log \psi)$ in the training phase, and similarly $O(n n_T b \log \psi)$ in the testing one, where $b$ is the branching factor of the PI-tree.

**AN EXPLAINABLE OUTLIER DETECTION METHOD USING REGION-PARTITION TREES** Another tree-based approach employed in anomaly

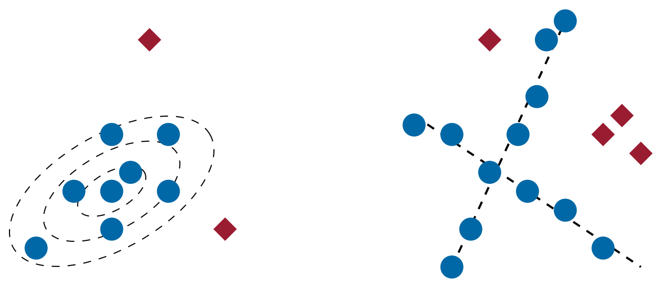


Figure 17: Statistical vs pattern anomalies. The latter are very complex since they are defined as points not belonging to a specific but unknown pattern. Figure adapted from [151].

detection is described in [191], where Region-Partition trees are employed. It is quite different from the IF. A tree with maximum height $h$ has the same amount of randomly selected features used at each depth level to split data. At each level a feature is selected and divided into $k$ intervals. This tabular structure is used to build the actual tree, starting from the empty root intervals and adding recursively a new data point, creating a new child when needed. This training procedure is performed only with normal data, therefore all nodes correspond only to regions where the distribution is expected to be normal. The detection of anomalies is quite simple since if a new sample arrives to a leaf node it is marked normal, if instead it gets stuck in a node it is labelled as outlier. In order to get an intuition about the cause that generated that anomaly, the authors suggest to count the number of times a feature is responsible to the internal node stop. To get the anomalous range, the intersection between the anomalous feature intervals for each tree can be easily performed.

### 4.3.2 *Dynamic Datasets*

Dynamic datasets are made up of infinite data streams [235]. This poses new challenges that previously described methods cannot tackle. The most simple challenge is the continuous training: since the stream is infinite, the training data may be insufficient to fully describe it. In order to do that, the model should continuously learn from the incoming data stream, and at the same moment detect anomalous points. The most complex setting is represented by the distributional drift. In this case, the stream is not stationary and its distribution experiences time-dependent variations. Here, the model must adapt to the evolving data stream, but at the same time discern anomalies from new normal data points.

The weak point of these methods is the assumption about the rarity of anomalies. If they are too numerous, they can be confused with a change in the normal distribution of data points, leading to an erroneous adaptation of the model. Doing this the model will consider them as normal and it will not raise the necessary alarm.

**an anomaly detection approach based on isolation forest algorithm for streaming data using sliding window (iforestasd)** The method proposed in [67] is an adaptation of the original algorithm to the streaming settings. It is very simple: it splits the stream in windows, and checks each window to detect anomalies. If the ratio between normal data and anomalies is too high (exceeds the expected anomaly ratio), it assumes a concept drift is happening. In this case the model is re-trained on the new window. Obviously, the threshold on the anomaly ratio and the width of the window are delicate hyper-parameters that highly depend on the specific application.

**fast anomaly detection for streaming data (streaming hs)** In the paper [235] an adaptation of [241] to data streams is presented. Unlike other tree based approaches, it is not built starting from training data but its structure is induced only by the feature space dimension. It doesn't need split point evaluation, and therefore it is fast and able to continuously learn from new data. Contrary to the basic Isolation Forest, this method employs the concept of mass to determine the anomaly score. In practice, this method works segmenting the stream in windows, and working with two of them: *reference* and the *latest* windows (despite their name, they are immediately consecutive). The reference window is used to record the mass profile, while the latest one is used for testing. When this is done, the latest becomes the reference and a new mass profile is recorded.

The authors show a time complexity of $O(n_T(h+\psi))$ in the worst case.

**rs-forest: a rapid density estimator for streaming anomaly detection (rs-forest)** The method published in [256] relies on a type of tree quite similar to the isolation one, that is let growing until a maximum depth is reached independently on the data. Before the tree construction, the feature space is enlarged in order to cope with possible feature drifts. The score is based on the density of the points in the leaf volume and the model update relies on dual node profiles similarly to the previous method.

The time complexity of the method is $O(\frac{n}{\psi}m2^{(h+1)})$ while space complexity is constant since is $O(2^h)$.

**an online anomaly detection method for stream data using isolation principle and statistic histogram (ahiforest)** Isolation Forest is a light-weighted method only designed for batch data, furthermore it suffers of slow convergence having no knowledge on the distribution of data.

In the paper [68] both these limitations are inspected, and a new method called AHIForest is proposed. This is identical to Isolation Forest, except for the selection step of the splitting point. Indeed, after a dimension is randomly picked, the choice of the threshold is not drawn from a uniform distribution, but it is based on the histogram that approximates the distribution of data projected on the given axis. The idea of histogram-based has the advantage of speeding up the convergence, at the price of adding a new critical parameter, the bin size, to be carefully chosen.

On the other hand, AHIForest deals with streaming data using a sliding window strategy. Firstly, a forest is built by data sampled from the first window, and new observations are judged in real time. Then, if anomaly rate exceeds a pre-defined threshold (as in iForestASD) or the buffer is full, the forest is updated, growing new trees from the last window and pruning old estimators.

The time complexity of this algorithm is $O(MN)$, where $N$ is the number of individual detectors and $M$ is the maximum number of leaves of each tree. Space complexity is $O(MN)$, too.

**ROBUST RANDOM CUT FOREST BASED ANOMALY DETECTION ON STREAMS (RRCF)** RRCF [102] presents exactly the same structure and anomaly scoring of isolation forest, except for the mechanism how the splitting dimension is chosen. The intuition of the authors was to pick what they called "robust random cuts" proportionally to the span of data in each dimension, instead of uniformly at random, as in the original version. In this manner, the method loses the property of scaling invariance of the Isolation Forest, but achieves some sense of self-consistency after addition or deletion of data, i.e. any tree preserves the same distributional properties independently if constructed given the whole dataset in a batch fashion and if dynamically grown from a stream of data.

Naturally, RRCF is a straightforward solution to propose the same key-ideas of Isolation Forest in a suitable way for online anomaly detection, without the need of rebuilding the model from scratch.

When a new data instance arrives, it runs across the tree, starting from the root. At each node, a candidate cut is randomly proposed in the subregion represented by the node, following the same mechanism as described above. If the new point is fully isolated by the candidate cut, this is kept and a new leaf is there inserted, otherwise the cut is discharged and the data instance moves the next node through the branch. Instance by instance, new branches grow up, making evolve the shape of trees in RRCF with respect to concept drift phenomena of data stream.

**FAST ANOMALY DETECTION IN MULTIPLE MULTI-DIMENSIONAL DATA STREAMS (STREAMING LSHIFOREST)** The model presented in [226] extends the LSHiForest algorithm for streaming data exploiting a dynamic isolation forest. The procedure can be split into three main phases: i) a dataset of historical data points is used to build a LSHiForest data structure, as presented in the original paper, then ii) the data points collected from multiple data streams are preprocessed to find "suspicious" samples, which are outlier candidates. Finally, iii) the suspicious data are updated into the LSHiForest structure and the anomaly scores of the updated data points are recalculated. To effectively extract suspicious points from the streaming data, Principal Component Analysis (PCA) and the weighted Page-Hinckley Test (PHT) are applied to a sliding window, to cope with the challenges of high dimensionality and concept drift. An update mechanism is proposed to iteratively update the LSHiForest by replacing the previous data points observed on a stream with suspicious ones.

**ISOLATION MONDRIAN FOREST FOR BATCH AND ONLINE ANOMALY DETECTION (ISOLATION MONDRIAN FOREST)** The Mondrian Forest [172] represents a family of random hierarchical binary partitions, based on the Mondrian Process, that tessellates the domain in a tree-like data structure. This random process recursively generates a series of axis-aligned slices that recall the abstract grid-based paintings by Piet Mondrian (1872 - 1944). Each slice is associated with a split time and the partitioning process can be eventually stopped after a given budget time. In the past few years, the interest upon Mondrian Forest raised up in machine learning, both for regression and classification purposes. Only recently, an application of Mondrian Forest has been proposed in anomaly detection, that exploits the similarities with the data structure from Isolation Forest, and uses the same depth-based anomaly score of the latter.

The advantage of Mondrian Forest lies on its nice self-consistency property, in particular a Mondrian Tree can be infinitely extended performing new partitions on its sub-domains, preserving the same distributional properties. Therefore a Mondrian Forest can be dynamically updated by the arrival of new data and this makes the algorithm particularly suitable for online / streaming applications.

The update mechanism is very similar to the one described for Robust Random Cut Forest. Again, each novel data point travels through the trees in the forest, and a candidate split is picked at each node. In this case, the candidate is maintained if its split time is lower than the one of the node; this can be interpreted as a split that is occurred before; otherwise it is discarded and the data instance moves to the next node until it reaches a leaf, which is associated with an infinite time.

The training and evaluation time complexities for online processing of $m$ new points are $O(n_T d(n+m)\log(n+m))$ and $O(n_T(n+m))$ respectively.

**INTERPRETABLE ANOMALY DETECTION WITH MONDRIAN POLYA FORESTS ON DATA STREAMS (MONDRIAN POLYA FOREST)** Mondrian Polya Forest (MPF) [65] is another example of a method based on the Mondrian Process, that seems one of the most promising research paths in tree-based approaches for anomaly detection.

As the previously described Isolation Mondrian Forest, Mondrian Polya Forest grows a tree partition based on Mondrian Process. The difference lies on the evaluation procedure, that estimates the density function of data instead of inferring their isolation scores. As the name suggests, MPF makes use of Polya Trees for modeling the distribution of the mass in the nested binary partition constructed by each Mondrian Tree, each cut is then associated with a beta-distributed random variable, which reflects the probability of a data point to lie in one of the sub-partitions in the hierarchical structure of the tree.

In this setting, an anomaly is identified by the fact it occurs in a region with lower density than normal data.

Alike Isolation Mondrian Forest, this method takes advantages of the properties of the Mondrian Forest, then it is able to efficiently update by nesting of new slices, instead of rebuilding the tree structure from scratch, when new data instances are available. A streaming version of the method has been proposed by the same authors of MPF with interesting results.

**ANOMALIES DETECTION USING ISOLATION IN CONCEPT-DRIFTING DATA STREAMS** In the paper [243] the authors review several isolation-based techniques in streaming anomaly detection, in particular iForestASD and Half-Space Trees, both previously introduced. The main differences are remarked, like the strong dependency by data of the first, versus the lack of knowledge in the building phase of the latter, and the different approaches for handling drift.

Moreover, a couple of new strategies are proposed, based on iForestASD. ADWIN (ADaptive WINdowing) is a well-known solution, that maintains a variable-length window of data, which increases with incoming observations. The algorithm compares any subset of the window until it detects a significant difference between data, then the new information is kept while the old one is erased. Slight variants are PADWIN (Prediction based ADWIN) and SADWIN (Scores based ADWIN), which take predictions and scores as input of ADWIN for detecting drift, respectively. KSWIN (Kolmogorov-Simirnov WINdowing) is a more innovative approach, based on Kolmogorov–Simirnov (KS) statistic test. KS is a non-parametric test, originally suitable for one-dimensional data only. The authors propose to overcome this restriction by declaring the occurrance of drift if it is detected in at least one dimension.

Empirical experiments show the inefficiency of vanilla iForestASD in real-world scenarios and the need of explicit concept drift detection methods, such as the proposed ones.

### 4.3.3 *Distributed approaches*

Wireless Senor Networks (WSNs) pose new and more challenging constraints to Anomaly detection. Indeed sensor nodes are usually quite cheap but have multiple constrains on energy consumption, communication bandwidth, memory and computational resources. Moreover they are often deployed in harsh environments that can corrupt sensor measurements and communication [66]. Despite the distributed nature of the network, Anomaly Detection on such applications should minimize the communication burden as much as possible, since data transmission is the most energy intensive process.

**DISTRIBUTED ISOLATION FOR WSN** The authors of [66] suggest the adaptation of IF to this distributed problem, considering the spatial correlation between neighbour sensor nodes in a local and global manner. They chose this base algorithm due to its already mentioned properties, that fits perfectly in this settings. However in the WSN context, data can be anomalous w.r.t. the single sensor node or w.r.t. the whole network. The local detector consists in a collection of isolation trees trained on a group of neighbouring nodes while the global one is made up of local detectors. When an anomaly is locally detected, it is marked as an error if it is not detected by neighbor sensors, otherwise it is considered an event.

The space and time complexity is $O(km)$, where $k$ is the number of trees on a local node, and $m$ the number of leaves.

**ROBUST DISTRIBUTED ANOMALY DETECTION USING OPTIMAL WEIGHTED ONE-CLASS RANDOM FORESTS (OW-OCRF)** A similar approach has been exploited in [246], where a one-class random forest has been chosen as base detector. Here each sensor node builds his own model, but it is also augmented with the models belonging to neighbouring devices. In addition, a strategy to weight the most effective neighbor models has been implemented, based on the minimization of the model uncertainty. Uniform voting is reasonable in circumstances where all the learners arise from the same distribution, but when models come from heterogeneous data distributions this strategy shows its weaknesses. Larger weights are assigned to trees that are in accordance with the majority, while trees that increase the overall uncertainty are penalized. The optimization of these weights is performed in a fully unsupervised fashion. In presence of distributional drifts, the overall model can be easily adapted to the new conditions, optimising new weights or substituting the trees with lower weight importance. The communication between the node is employed just at early stages for the sharing of the detecting models, not for the sampled data sharing.

The time and space complexity of this approach are $O(n_T h)$ and $O(n_T 2^{h+1})$ respectively.

### 4.3.4 *Interpretability and feature selection*

The detection of anomalies is an important activity in manufacturing processes but it is useless if a corresponding action does not take place. That action is expected to be proportional to the gravity of the anomaly (encoded by the anomaly score), and to the *cause* that generated it. For doing that a tool to interpret that anomaly is needed. It is easy to understand that if unsupervised anomaly detection is challenging, interpretable models face even more complex issues. In real word scenarios anomalies are unlabelled and lack of proper interpretations.

Moreover [45] observes that interpretable models enhance the trust of the user in the anomaly detection algorithms, leading to a more systematic use of these tools.

**INTERPRETABLE ANOMALY DETECTION WITH DIFFI: DEPTH-BASED FEATURE IMPORTANCE FOR THE ISOLATION FOREST (DIFFI)** IF is a highly randomised algorithm and therefore the logic behind the model predictions is very hard to grasp. In the paper [45], a model specifically designed for the interpretability of IF outcomes is presented. In particular the authors developed two variants: i) a global interpretability method able to describe the general behaviour of the IF on the training set, and ii) a local version able to explain the individual IF predictions made on test points. The central idea of this method, named DIFFI, relies on the following two intuitions: the split of an important feature should a) induce faster isolation at low levels (close to the root) and b) produce higher unbalance w.r.t splits of less important features. This is encoded in a new index named *cumulative feature importance*. With this in mind, the authors formulate the global feature importance as the weighted ratio between the cumulative feature importance computed for outliers and inliers. The local interpretation of single detected anomalies is sightly different but relies on the same intuitions. DIFFI can also be exploited for unsupervised feature selection in the context of anomaly detection.

**ANOMALY EXPLANATION WITH RANDOM FORESTS** Authors in [139] developed an algorithm able to explain the outcome of a generic anomaly detector by using sets of human understandable rules. More specifically, the proposed model consists in a special random forest trained to separate the single anomaly from the rest of the dataset. This algorithm provides two kinds of explanations: the *minimal* and the *maximal*. The first is performed isolating the anomaly using the minimal number of necessary features. On the contrary the maximal explanation looks for all the features in which the anomaly is different, employing a recursive feature reduction. Once the forest are trained and the explanations are obtained, the decision rules are extracted in a human readable manner.

The time complexity of the algorithm is $O(n_t T_{\text{sel}} T_{\text{train}})$ where $T_{\text{sel}}$ is linear with the number of normal samples in the data. For the minimal explanation $n_t$ is the number of trees trained for each anomaly and $T_{\text{train}} = O(d|T|^2)$, where $d$ is the number of features and $T$ is the size of the training set. For the maximal explanation $n_t = O(d-1)$ while $T_{\text{train}} = O(d^2|T|^2)$.

**ISOLATION-BASED FEATURE SELECTION FOR UNSUPERVISED OUTLIER DETECTION (IBFS)** In settings where the dimensionality of data is very high, even the most efficient anomaly detection algorithm may suffer. Methods of feature selection are used to the purpose of reducing the computational and memory cost. Isolation-based feature selection (IBFS) described in [262] computes an unbalanced score each time a node is split, based on the resulting entropy weighted by fraction of data samples in each leaf. Adding all the scores of the traversed nodes, it is possible to obtain a global features score that highlights the best features for anomaly detection.

## 4.4 EXPERIMENTAL COMPARISON

### 4.4.1 *Methods comparison and available implementations*

Unfortunately a small subset of authors provided an open implementation of the methods presented in the previous Section: for this reason it is hard to have a comprehensive overview of performances for the overall plethora of isolation-based and tree-based methods. Most of the authors report some performance scores (commonly ROC AUC) for their proposed methods, using as benchmarks only the original IF and few of the most popular close variants, such as Split-Criteria IF, Robust Random Cut Forest or Extended IF and other density or distance-based anomaly detection approaches.

To the best of our knowledge, an extended comparison between all the variants of tree-based AD has never been realised. To cope with this issue, we have worked to collect results available in literature on various AD benchmarks, in order to provide an easier comparison between the different approaches also in terms of accuracy.

We selected a subset of the datasets where we could have a consistent amount of outcomes, that turn out to be all from UCI Machine Learning Repository [72]. For this reason, we excluded all the methods that are intended to work on a specific scope, for instance image detection or functional-based anomaly detection. A schematic description of datasets is in Table 3.

Table 3: Description of test data sets.

| **Dataset** | **Size** | **Dim.** | **% anomalies** |
|---|---:|---:|---:|
| **HTTP** | 567497 | 3 | 0.4% |
| **SMTP** | 95156 | 3 | 0.03% |
| **Forest Cover** | 286048 | 10 | 0.9% |
| **Shuttle** | 49097 | 9 | 7% |
| **Mammography** | 11183 | 6 | 2% |
| **Satellite** | 6435 | 36 | 32% |
| **Pima** | 768 | 8 | 35% |
| **Breastw** | 683 | 9 | 35% |
| **Arrhytmia** | 452 | 274 | 15% |
| **Ionosphere** | 351 | 32 | 32% |

Moreover, we limited to the static approach, since testing for streaming algorithms allows a variety of different setups and it is hard or impossible to achieve a fair comparison with existing results.

Table 4 contains the result from our survey. In all cases it was possible, we used the ROC AUC scores from the original papers, and we suppose each method is tuned at the best of author's expertise. For many of the algorithms that were publicly available, we filled the eventually missing scores by running the tests ourselves. In this case, we consider those hyperparameters indicated in the original IF paper [160] (number of trees = 100, sample size = 256) as the most appropriate universal setup. Finally, we avoided to fill missing outcomes for such methods, like GIF, where the authors provided a fine-tuning of their algorithms, since our last assumptions would not replicate the same performances.

From the collected scores, we have not found a method consistently outperforming all the others, and it's not clear how to build a hierarchy between all the variants. We can conclude that IF is an efficient baseline that shows good performance in many cases, however each method may be the most appropriate in any specific real-life scenario and the final choice it can only be up to the practitioner.

One of the limitations of the proposed comparison is related to the choice of the Area Under Receiver Operating Characteristic Curve (ROC AUC) as main performance indicator used in most of the reviewed papers. In fact, [212] already highlights the inefficiency of ROC AUC if data are strongly unbalanced, and suggests the usage of other metrics, such as Area Under Precision-Recall Curve (PR AUC): ROC curve is drawn by plotting the true positive rate (or recall) against the the false positive rate; however, when positive labelled data are rare, ROC AUC can be misleading since even a poor skilled models can achieve high scores. For such reasons, the validity of the reported results for strongly unbalanced datasets like HTTP, SMTP or Forest Cover should be considered with some skepticism.

On the contrary, PR curve represents the precision over the recall for a binary classifier, and it would be more informative when normal instances outnumber anomalies. A slightly different alternative is Precision-Recall-Gain (PRG) curve [82]. Specifically, PRG AUC maintains the pros of the PR AUC, but allows to evaluate the model against a baseline binary classifier, i.e. the *always-positive* classifier, as ROC AUC does with the random classifier model.

There are many other metrics in the literature besides ROC AUC and PR AUC that capture different aspects of detection performance. Although in this thesis we will mainly focus on PR AUC because it is a robust and widely employed measure of detection performance, we refer the interested reader to [39].

In order to promote reproducibility, and to help practitioners in developing real-word applications, we provide a list of the available source codes about the previously discussed methods (Table 5). Unfortunately, as stated above, we were able to retrieve just a portion of the reviewed methods but we hope as anomaly detection becomes a more mature field, authors will be more used to share their code for enhancing adoption and comparisons of the proposed approaches.

Table 4: ROC AUC score of a selection of methods from literature.

| | Code available | HTTP | SMTP | Forest C. | Shuttle | Mammogr. | Satellite | Pima | Breastw | Arrhytmia | Ionosph. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **IF** | ✓ | **1.00** | 0.88 | 0.88 | **1.00** | 0.86 | 0.71 | 0.67 | **0.99** | 0.80 | 0.85 |
| **SCIForest** | ✓ | **1.00** | - | 0.74 [269] | **1.00** [269] | 0.59 [38] | 0.71 [269] | 0.65 [269] | 0.98 [269] | 0.72 [269] | 0.91 [258] |
| **EIF** | ✓ | *0.99* | 0.85 [114] | 0.92 | 0.99 [114] | 0.86 | 0.78 | 0.70 | ***0.99*** | 0.80 [114] | 0.91 |
| **RRCF** | ✓ | 0.99 [101] | 0.89 [101] | *0.91* | 0.91 [65] | 0.83 [65] | 0.68 [65] | 0.59 [65] | 0.64 [65] | 0.74 [65] | 0.90 [65] |
| **IMF** | ✓ | ***1.00*** | 0.87 | *0.90* | *0.99* | *0.74* | 0.74 | 0.64 | *0.97* | *0.80* | 0.86 |
| **MPF** | | **1.00** | 0.84 | 0.77 | 0.51 | 0.87 | 0.70 | 0.66 | 0.97 | 0.81 | 0.88 |
| **PIDForest** | ✓ | 0.99 | **0.92** | 0.84 | **0.99** | 0.84 | 0.70 | 0.70 [65] | 0.99 | - | 0.84 [65] |
| **OPHiForest** | | - | - | - | 0.99 | - | 0.77 | 0.72 | 0.96 | 0.78 | 0.93 |
| **LSHiForest** | ✓ | - | - | 0.94 | 0.97 | - | 0.77 | 0.71 | 0.98 | 0.78 | 0.91 |
| **HIF** | ✓ | - | 0.90 | - | **1.00** | **0.88** | 0.74 | 0.70 | *0.98* | 0.80 | *0.86* |
| **HEIF** | | - | 0.90 | - | 0.99 | 0.83 | 0.73 | 0.72 | - | 0.80 | - |
| **OneClassRF** | ✓ | 0.98 | **0.92** | 0.85 | 0.95 | - | - | 0.71 | - | 0.70 | 0.90 |
| **T-Forest** | | 0.99 | - | - | 0.99 | - | 0.68 | 0.71 | - | **0.84** | 0.94 |
| **EGiTree** | | - | - | **0.97** | 0.94 | - | 0.73 | - | - | - | 0.94 |
| **GIF** | ✓ | - | - | 0.94 | - | 0.87 | **0.86** | **0.84** | - | - | - |
| **dForest** | | **1.00** | - | - | **1.00** | - | - | 0.75 | **0.99** | - | **0.97** |
| **ReMass IF** | | **1.00** | 0.88 | 0.96 | **1.00** | 0.86 | 0.71 | - | **0.99** | 0.80 | 0.89 |
| **HSF** | ✓ | **1.00** | 0.90 | 0.89 | **1.00** | 0.86 | - | 0.69 | **0.99** | **0.84** | 0.80 |

Selected algorithms are: Isolation Forest (IF) [160], Split-Criteria Isolation Forest (SCIForest) [161], Extended Isolation Forest (EIF) [107], Robust Random Cut Forest (RRCF) [102], Isolation Mondrian Forest (IMF) [172], Mondrian Polya Forest (MPF) [65], Partial Identification Forest (PIDForest) [101], Order Preserving Hashing Based Isolation Forest (OPHIF) [258], Locality Sensitive Hashing Isolation Forest (LSHiForest) [269], Hybrid Isolation Forest (HIF) [177], Hybrid Extended Isolation Forest (HEIF) [114], One-class Random Forest (OneClassRF) [99], Trident Forest (T-Forest) [267], Entropy-based Greedy Isolation Tree (EGiTree) [155], Generalized Isolation Forest (GIF) [38], Distribution Forest (dForest) [263], Re-Mass Isolation Forest (ReMass IF) [18], Half-Spaces Forest (HSF) [241].
Scores are referenced when are provided by a different source than the original paper for the method. If scores were not available in the original paper, we have performed the missing experiments when an implementation of the algorithm was available or by using our own implementation: such cases were reported by using Italic entries; in these circumstances we always performed testing with 100 trees and sample size equals to 256. If scores were not available in the original paper and no implementation of the algorithm were available, the entries were left blank '-'. Finally, we highlighted by bold entries the best performances for each dataset.

Table 5: Source code repositories.

| Model | Repository | Language |
|---|---|---|
| DIFFI [45] | github.com/mattiacarletti/DIFFI | Python |
| EIF [107] | github.com/sahandha/eif | Python |
| Functional IF [223] | github.com/GuillaumeStaermanML/FIF | Python |
| GIF [38] | github.com/philippjh/genif | Python |
| HIF [202] | github.com/pfmarteau/HIF | Python |
| IF [160] | scikit-learn.org | Python |
| iForestASD [67, 243] | github.com/Elmecio/IForestASD_based_methods_in_scikit_Multiflow | Python |
| Isolation Mondrian Forest [172] | github.com/bghojogh/iMondrian | Python |
| LSHiForest [269] | github.com/xuyun-zhang/LSHiForest | Python |
| MassAD [241] | sourceforge.net/projects/mass-estimation | MATLAB |
| OneClassRF [99] | github.com/ngoix/OCRF | Python |
| PIDForest [101] | github.com/vatsalsharan/pidforest | Python |
| RRCF [102] | github.com/kLabUM/rrcf | Python |
| SCiForest [161] | github.com/david-cortes/isotree | Python |

### 4.4.2 *Industrial Case Studies*

Tree based AD approaches have been extensively employed in industry because of their nice properties. Some of the relevant industrial applications of tree based methods are summarized in Table 6. Despite the existence of multiple tree-based algorithms, the large majority of applications concerns the original Isolation Forest and a big part of them are applications in the power industry. Fraud detection and cybersecurity examples, while being really popular in the literature, were not considered in this list since they are not strictly industrial applications.

In some of the reported cases, authors used the reported anomaly detection method as part of a more complex pipeline that typically involve a feature extraction procedure when dealing with non-tabular data for example: for the sake of simplicity, we didn't report such 'evolutions' of the methods in our classification.

Table 6: Industrial applications of tree-based approaches for Anomaly Detection.

| Work | Year | Sector/Equipment Type | Method |
|---|---|---|---|
| Ahmed et at. [6] | 2019 | Smart Grid | IF |
| Alsini et al. [9] | 2021 | Construction Industry | IF |
| Antonini et al. [15] | 2018 | IoT audio sensors | IF |
| Barbariol et al. [27] | 2020 | Multi-phase Flow Meters | IF |
| Brito et al. [37] | 2021 | Rotating Machinery | DIFFI |
| Carletti et al. [44] | 2019 | Home Appliances Manufacturing | DIFFI |
| De Santis et al. [213] | 2020 | Power Plants | EIF |
| Du et al. [71] | 2020 | Sensor Networks | IF |
| Hara et al. [106] | 2020 | Hydroelectric Generators | IF |
| Hofmockel et al. [113] | 2018 | Vehicle Sensors | IF |
| Li et al. [152] | 2021 | Machine Tools | IF |
| Lin et al. [156] | 2020 | Power Plants | IF |
| Luo et al. [170] | 2019 | Eletricity Consuption | IF |
| Kim et al. [136] | 2017 | Energy & Smart Grids | IF |
| Maggipinto et al. [173] | 2019 | Semiconductor Manufacturing | IF |
| Mao et al. [175] | 2018 | Power Consumption | IF |
| Puggini et al. [199] | 2018 | Semiconductor Manufacturing | IF |
| Riazi et al. [206] | 2019 | Robotic Arm | IF |
| Susto et al. [228] | 2017 | Semiconductor Manufacturing | IF |
| Tan et al. [236] | 2020 | Marine Gas Turbines | IF |
| Tran et al. [245] | 2020 | Fashion Industry | IF |
| Wang et al. [249] | 2019 | Power Transformers & Gas-insulated Swithchgear | IF |
| Wetzig et al. [254] | 2019 | IoT-Gateway | Streaming HS |
| Wu et al. [257] | 2018 | Energy & Smart Grid | IF |
| Zhang et al. [270] | 2019 | Cigarette Production | IF |
| Zhong et al. [274] | 2019 | Gas Turbine | IF |

In this review, many approaches have been listed and it might be hard to get a feeling on their actual importance for the research community, also given the fact that many approaches have been only recently submitted. To mitigate such issue, some statistic related to the method citations have been collected in Table 7 and 8. Given that citations are only a proxy of the relevance of an AD method and that it is somehow unfair to compare citation of recently introduced methods versus established ones, the proposed list should be taken as a loose reference for listed methods importance.

Table 7: Static methods. The '*' highlights methods published in 2020 or 2021 for which the reported statistics at the time of the writing of this work (March 2021) is of course not reliable.

| Paper | Acronym | Citations in 2020 | Total citations | Annual rate |
|---|---|---|---|---|
| Liu et al. (2010) [161] | SCIForest | 10 | 49 | 4.1 |
| Aryal et al. (2014) [18] | ReMass | 7 | 20 | 2.5 |
| Li et al. (2020) [153] | | *7 | *7 | *3.5 |
| Ting et al. (2010) [241] | MassAD | 6 | 53 | 4.4 |
| Zhang et al. (2017) [269] | | 6 | 26 | 5.2 |
| Karczmarek et al. (2020) [131] | K-Means IF | *5 | *7 | *3.5 |
| Liu et al. (2018) [167] | | 4 | 9 | 2.2 |
| Marteau et al. (2017) [177] | HIF | 4 | 6 | 1.2 |
| Staerman et al. (2019) [223] | Functional IF | 3 | 5 | 1.7 |
| Goix et al. (2017) [99] | OneClassRF | 2 | 4 | 0.8 |
| Yu et al. (2009) [265] | | 1 | 26 | 2.0 |
| Zhang et al. (2018) [267] | T-Forest | 1 | 4 | 1.0 |
| Chen et al. (2015) [49] | RPF | 1 | 3 | 0.4 |
| Shen et al. (2016) [220] | EGiTree | 1 | 2 | 0.3 |
| Hariri et al. (2021) [107] | EIF | *1 | *1 | *1.0 |
| Mensi et al. (2019) [180] | | 1 | 1 | 0.3 |
| Gopalan et al. (2019) [101] | PIDForest | 1 | 1 | 0.3 |
| Liao et al. (2019) [155] | E-iForest | 0 | 5 | 1.7 |
| Chen et al. (2011) [50] | kpList | 0 | 4 | 0.4 |
| Aryal et al. (2021) [17] | usfAD | *0 | *1 | *1.0 |
| Buschjager et al. (2020) [38] | GIF | *0 | *1 | *0.5 |
| Park et al. (2021) [191] | | *0 | *0 | *0.0 |
| Holmer et al. (2019) [114] | HEIF | 0 | 0 | 0.0 |
| Ghaddar et al. (2019) [95] | | 0 | 0 | 0.0 |
| Yao et al. (2019) [263] | dForest | 0 | 0 | 0.0 |
| Karczmarek et al. (2020) [130] | n-ary IF | *0 | *0 | *0.0 |
| Sternby et al. (2020) [224] | ADF | *0 | *0 | *0.0 |
| Xiang et al. (2020) [258] | OPHIForest | *0 | *0 | *0.0 |
| Gao et al. (2019) [92] | CBIF | 0 | 0 | 0.0 |
| Leveni et al. (2021) [151] | PIF | *0 | *0 | *0.0 |
| Lyu et al. (2020) [171] | RMSHForest | *0 | *0 | *0.0 |
| Karczmarek et al. (2020) [129] | Fuzzy IF | *0 | *0 | *0.0 |
| Qu et al. (2020) [200] | | *0 | *0 | *0.0 |

MassAD gathers the citations from Ting et al. (2010) [241] and Ting et al. (2013) [242].

Table 8: Dynamic methods. The '*' highlights methods published in 2020 or 2021 for which the reported statistics at the time of the writing of this work (March 2021) is of course not reliable.

| Paper | Acronym | Citations in 2020 | Total citations | Annual rate |
|---|---|---|---|---|
| Ding et al. (2013) [67] | iForestASD | 25 | 69 | 7.7 |
| Tan et al. (2011) [235] | Streaming HS | 17 | 82 | 7.5 |
| Guha et al. (2016) [102] | RRCF | 9 | 23 | 3.8 |
| Wu et al. (2014) [256] | RS-Forest | 7 | 39 | 4.9 |
| Ding et al. (2015) [68] | AHIForest | 1 | 3 | 0.4 |
| Sun et al. (2019) [226] | Streaming LSHiForest | 0 | 2 | 0.7 |
| Ma et al. (2020) [172] | Isolation Mondrian Forest | *0 | *0 | *0.0 |
| Dickens et al. (2020) [65] | Mondrian Polya Forest | *0 | *0 | *0.0 |
| Togbe et al. (2021) [243] | | *0 | *0 | *0.0 |

## 4.5 CONCLUSION AND FUTURE WORK

In this work we focused on anomaly detection, a practical problem that many times arises in industrial applications. Indeed, the detection of product defects or production instruments faults can be quickly addressed by this kind of techniques.

This review dealt with a particular type of algorithms based on tree structure. These have many advantages, like fast computations, low latency, low memory requirements, parallelism and high detection performances. Moreover, they can cope with the streaming data scenario where the model has to adapt to new incoming data. Moreover, recent efforts have been made by the scientific community to equip such methods with interpretable traits, making them particularly appealing in real-world contexts where root cause analysis is also of paramount importance.

The main procedural differences between the different methods have been discussed and the performances declared by their authors have been compared.

Use cases and a list of ready to use implementations has been made in order to provide practitioners an effective review. The methods performances over different datasets have been grouped together in a unique table.

This Chapter has highlighted the many advantages of tree-based approaches over competing alternatives, and the different strategies proposed by the authors.

Some of them are very similar but others introduced very interesting novelties. Just to name a few, the authors found very promising the isolation principle, the anomaly score based on the mass in addition to tree depth, the weighted trees, the pattern anomalies, the continuous training made on data streams and the split criterions that try to accelerate the isolation.

On the other side, criteria that rely on distances other than $L1$-distance, or that try to directly estimate the density, risk to quickly lose the advantage over more traditional methods.

The present study has some limitations, mainly due to the fact that many methods are recent or do not have public implementations provided by the proposing authors, nevertheless this work is intended to be a stating point for future investigations. The most important one lies in the performance comparison; moreover the provided tables have been assembled using results declared in the reviewed papers, so caution must be taken when looking at this comparison and the time complexities.

# 5 WINDOWING ANOMALY DETECTION

The first anomaly detection approach on time-series is by applying static models on intervals (windows) of the evolving signal. This approach, even if conceptually simple, has some issue that will be discussed in this Chapter. The goal of this work is to find a computationally simple approach in order to filter the behaviour of the measurement system, from the behaviour of the measured process. In fact the measured flow is non-stationary and might evolve in a multitude of possible flow patterns that are perfectly normal but might not be seen during the training phase. An anomalous measurement might be due to a possible fault, or might be due to a flow condition that was not observed before. This motivates the work presented in this Chapter and published in [27] that tries to decouple the flow evolution from the instrument in order to apply AD model on the measurement process.

## 5.1 INTRODUCTION

In recent years, many efforts have been done in order to improve the MPFM performances, increasing its complexity: many sensors have been implemented on-board, improving the number of available physical variables. Unfortunately, the increased complexity is associated with a bigger set of failure types that the system can experience. The reliability of this system is crucial for every customer that consumes the supplied data for monitoring, decision making and control the oil production. To achieve the required reliability, the MPFM must be able to self-diagnose its sensors in an autonomous fashion.

Besides the energetic sector, data reliability is fundamental in many fields such as: finance, IT, security, medical, e-commerce, agriculture, and social media. With the advent of "Industry 4.0", the interest in accurate metrology tools has pushed many companies to develop complex measuring instruments. These sophisticated systems, that we call Complex Measuring Systems (CMSs), estimate quantities difficult to be measured combining different sensors measurements and data fusion techniques [253]. The MPFM is one example of CMSs and therefore in the following pages, the acronyms will be treated as synonyms. Although specifically developed for the MPFM, the proposed methodology could be applied to other CMSs.

Self-diagnosis capabilities can be reached in two ways: by leveraging a priori/physical knowledge of the metrology tool and the underlying measured phenomena, or by exploiting historical data and statistics/Machine Learning. In the latter case (that is the focus of this work), several algorithms can be employed, both in the realm of classification approaches and in the category of unsupervised anomaly detection methodologies. The choice mainly depends on the type of available data. In the context of MPFM, these techniques cannot be directly applied and many constrains have to be taken into account; such challenges can be summarized as follows:

- *Data structure and multi-dimensionality*: MPFMs are usually made up of different *metrology modules* that measure quantities related to distinct physical variables and each module can have a different sampling rate. As a result, data to be monitored are a collection of multivariate and heterogeneous streaming of time-series.

- *Non-stationarity*: The measured process can be non-stationary, i.e. the mean and the variance of the underlying process can vary over time; in this context, historical data can be not representative of the possible system states. To overcome this issue, the monitoring agent has to find some quantities that describe the behaviour of the instrument, independently on the process.

- *Root cause analysis*: Due to the changing environment and the complex interaction between the modules, identifying the faulty sensor/module can be a quite difficult task.

- *Edge application*: For most MPFMs, self-diagnosis capabilities have to be equipped on the edge, since: (i) many systems have to provide data in real-time, (ii) Internet-of-Things scenario [103] may not be feasible or cost-effective for many customers.

- *Limited Resources*: Many MPFMs have unfortunately limited computational power and memory capacity.

Given the problems previously discussed, there is a great need for developing robust algorithmic techniques for MPFM self-diagnosis that are able to: (i) provide confidence intervals for the measures they produce and to (ii) detect sensors/measuring modules that are proving anomalous measures. Such targets may be achieved by the usage of *Anomaly Detection* techniques [127]. More specifically, given the above issues and following the guidelines in [4], we define here the general requirements for real-world anomaly detection algorithm applied to the MPFMs:

- predictions must be made in real time and on-board;

- algorithms should not rely on historical data to be trained, but it must learn from data collected on the field;

- outcomes of the anomaly detection module need to be accurate and robust, i.e. they shouldn't be affected by changes in the underlying process;

- anomaly detection algorithms should minimize an *optimal* trade-off between false positives and false negatives; the optimality is defined given the application at hand;

- the computations must be as light as possible;

- the anomaly detection algorithm must handle different sampling frequencies;
- the anomaly detection algorithm must be independent on the meter calibration settings.

The focus of this work will be to present an Anomaly Detection approach for MPFM that considers the aforementioned constraints and allows to equip the instrument with self-diagnosis capabilities. In particular, a preprocessing pipeline specifically designed for this type of meter will be shown. This will allow to: i) effectively employ computationally efficient Anomaly Detection tools; ii) develop a powerful Root Cause Analysis algorithm. Moreover, we will test such approach exploiting real data, coming from specialized testing facilities, called flow loops. In the following we will refer to the proposed approach with the name *AD4MPFM* that stands for: Anomaly Detection for Multiphase Flow Meters.

DATA-DRIVEN ANOMALY DETECTION Classically, a model-based strategy is the most natural approach to tackle the Anomaly Detection problem. However, the performances of these strategies are strongly dependent on the quality of the model they rely on. Finding a model for a MPFM can be particularly challenging, since it means: i) modelling all the measuring modules, their interaction and the underlying process; ii) due to the system non-stationarity, the usage of fixed models without customization and periodic updates is hardly viable.

By exploiting the availability of historical data, a data-driven strategy is typically preferable. Data-driven approaches [42, 51, 52, 251] aims at finding a model that best fits data without physical a priori knowledge of the process. They are usually divided into two main categories: supervised and unsupervised. Supervised methods are very powerful but need data where each sample is labelled *Faulty/Non-Faulty*. Unfortunately labelled datasets are hardly available: the labelling procedure is very expensive and time-consuming. A human domain expert has to manually tag the dataset with a posteriori visual inspection. However the large number of module combinations and operating conditions makes these methods typically hardly viable.

Unsupervised techniques allow to overcome these issues. Their main benefits are [3]: (i) the capacity of handling multivariate data and multi-modal distributions and (ii) the fact that they do not need labels. Unsupervised Anomaly Detection (AD) algorithms are fundamental tools to detect anomalous behaviour in metrology instruments. Working in an unsupervised way, they are flexible, accurate and highly suited for fault detection in complex systems. They are able to perceive abnormal instrument behaviours that even domain experts struggle to detect. Given the advantages described above, AD approaches have been applied in various contexts: automotive [237], biomedical instruments [179], decision support systems [43], fraud detection [232] and industrial processes [231].

In literature there are many definitions of anomaly depending on the application of interest. Hawkins in [109] describes an outlier in a way that seems the more appropriate to our problem:

> *"An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism."*

Therefore outliers are *far* and, hopefully, *less* than inliers. The choice on how far and how few the observations must be to be outliers is not trivial. The outlier numerosity can vary: it is reasonable to assume that anomalous observations will increase as the environment ruins the sensor. When the fault is fully developed the outliers can be a significant percentage of the total samples.

LITERATURE REVIEW Despite the importance of detecting malfunctions in MPFMs, the literature on self-diagnosis and AD applied to CMS is not very broad. Traditionally, CMS are monitored employing univariate control charts [7, 86] that fail to capture complex data behaviours and multidimensional anomalies. Moreover they are generally more effective in monitoring the underlying process instead of the performance of the CMS. More refined techniques that manage multivariate data, are scattered on a lot of different applications such as automotive [40], aerospace [21], chemical industry [238], Heating, Ventilating and Air Conditioning [31] and other industrial/manufacturing applications [12, 32, 124, 141, 168, 227]. Other approaches that are similar to the one set by the CMS, can be found in [190, 264] and [122] where AD techniques are applied to wireless sensor networks in a non-stationary environment. Unsupervised methods able to detect the root cause of the CMS fault are, to the best of our knowledge, still missing. As stated above, the literature concerning the self-diagnosis and fault detection applied to MPFMs is at its early stages.

In general, AD detection on multivariate time series is still a quite unexplored research field. The existing literature is mainly divided into techniques applied to multivariate static datasets [229], and methods applied to univariate time-series [230].

Some of the most important multivariate static emerging techniques will be briefly described in Section 5.2. Concerning AD on univariate time series we cite the following approaches. [5] employs a neural network to model the time-series and detects the anomaly looking at the residue: an anomaly occurs when the actual value is too far from the network prediction. In [112] the authors look for contextual anomalies in univariate time-series that show a strong seasonality. They decompose the series in median, seasonal and residue and use robust univariate AD methods on signal windows. Two interesting approaches that consider the correlation between the measures can be found in [145] and [119]. The last one faces a problem very similar to ours since they manage highly dynamic, correlated, and heterogeneous sensor data.

In this Chapter we try to combine and to take the best from these approaches, enlarging the few unsupervised tools applied to multivariate time series. The proposed algorithm is designed to fit the MPFM requirements and

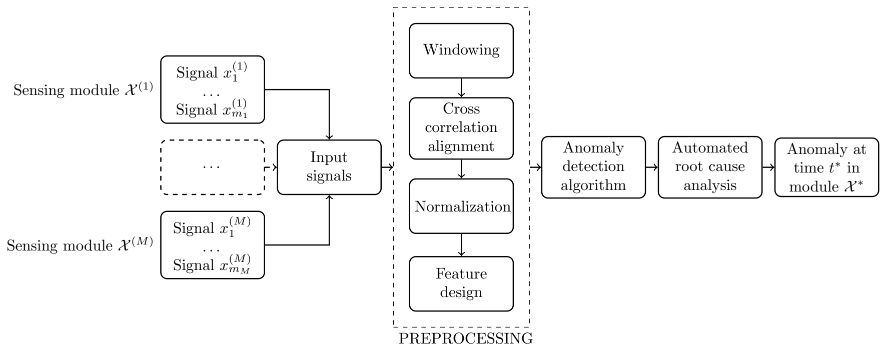

Sensing module $\mathcal{X}^{(1)}$
Signal $x_1^{(1)}$
...
Signal $x_{m_1}^{(1)}$
...
Sensing module $\mathcal{X}^{(M)}$
Signal $x_1^{(M)}$
...
Signal $x_{m_M}^{(M)}$
Input signals
Windowing
Cross correlation alignment
Normalization
Feature design
PREPROCESSING
Anomaly detection algorithm
Automated root cause analysis
Anomaly at time $t^*$ in module $\mathcal{X}^*$

Figure 18: AD4MPFM: Flowchart of the proposed algorithm for anomaly detection of Multiphase Flow Meters.

characteristics described in Section 2. Moreover the AD4MPFM is able not only to detect the fault, but also to detect the root cause in an unsupervised fashion. In Chapter 9 more sophisticate approaches will be developed to address the same problem.

## 5.2 PROPOSED APPROACH

The key points that led us to the development of AD4MPFM were basically three:

- non-stationarity of the underlying process;
- high correlation between the measuring modules;
- need for light computations to enable edge computation.

As already mentioned, the MPFM is employed in the estimation of physical quantities that are challenging to be measured. For doing this, it might combine many measurements coming from different modules. The physical quantities to be measured might vary widely and frequently over time: this makes the use of historical data very hard. Indeed, when available, these data might not describe all the possible operating points of the MPFM. In addition the same type of MPFM can behave differently due to different sensors calibration. Therefore we need a monitoring algorithm that is:

- *independent on the process* - it is able to decouple the dynamics of the measured process from the dynamics of the instrumentation;
- *independent on the calibration settings* - i.e. able to auto-tune its parameters.

In MPFM it is frequent to have highly correlated sensing modules: some sensing modules might measure the same physical property in order to get robust measurement or to measure derived quantities (e.g. velocity). A more interesting case is the correlation between modules that measure different physical properties: they are usually at least *locally correlated* because they see the same physical process (e.g. the appearance of a bubble, or the development of a slug flow) by different perspectives. In other words, all sensors at any given moment measure a different property of the *same* physical phenomenon. We stress that by locally correlated we mean correlated on a sufficiently small time interval.

A common strategy in AD [5, 93] is to model the underlying process and looking at the residue between the prediction and the actual value; when the residue becomes bigger than the confidence interval of the predictive model, the anomaly is detected. However modelling a non stationary process is very complex from many points of view and, as explained above, in this context we need a computationally light model that does not need to be trained on historical data, therefore this kind of strategy is not viable in AD for MPFM.

Although we cannot afford to model and forecast the evolution signal $x_i$, we develop a different approach based on the following intuition: since the correlation between $x_i$ and $x_j$ is sufficiently high, we can think of $x_j$ as an approximate model for $x_i$, and vice versa. Looking at the difference between correlated modules, we filter the process dynamics and we are able to analyze the instrumentation behaviour. The basic assumption is that an anomaly is likely to be present when the modules disagree: it occurs when the informative content of one signal differs from all the others. This approach does not need historical data but it only needs to find a suitable way to get the difference between different sensor measurements. Obviously, the higher the correlation between the modules is, the better this kind of filter will work. If the instrumentation works in ideal way, the differences will be gaussians.

ALGORITHM The algorithm depicted in the block diagram in Figure 18 is our proposal for AD for MPFM, namely AD4MPFM, and it expresses the fundamental intuition detailed in the previous sub-section. In this sub-section every step of this flowchart will be analyzed in detail.

DATA COLLECTION The first step is the collection of signals belonging to the locally correlated modules. Since these modules might have different acquisition rates, a homogenisation of the signals time frequencies has to be done. This can be achieved taking the mean of all the signals over the lowest sampling rate.

WINDOWING The AD4MPFM algorithm requires fixed size batches; for this reason, the signals need to be windowed. The windowing procedure splits the original signals into small overlapping intervals of size $\tau$ and time-overlap $\alpha$.

ALIGNMENT The sensing modules might have a time delay between them. However the efficacy of the following feature design is obtained only when the modules measure the same event at the same time, namely when the signals are well aligned in time and the correlation between them is sufficiently high. To ensure such alignment a cross correlation procedure is employed; such choice over more sophisticated time alignment algorithms like Dynamic Time Warping [133] was motivated by the computational resources constrains.

NORMALISATION An important and simple preprocessing procedure for comparing two signals belonging to sensors that measure different physical properties, is by normalising them. This step is very simple but quite delicate:

- subtracting the mean and dividing by standard deviation can be dangerous, due to the possible presence of outliers. To overcome this issue, we decided to normalise each signal window using the median and median absolute deviation;
- in order to be consistent between different windows of the same signal, we have defined a *reference* batch, whose median and median absolute deviation $mad$ have been used to normalize the other batches;
- despite the outlier robustness of the median and mad, the choice of the reference batch has to be done carefully. This batch should be collected in controlled conditions, keeping attention to the possible outliers.

FEATURE DESIGN The next step in the AD4MPFM is to encode the previously explained insight: the features should express the difference between the informative content of the signals. Given two normalised and windowed signal $x_i$ and $x_j$, the feature $z_{ij}$ has been defined as:

$$z_{ij} = x_i - \text{sign}(K_{ij})x_j \qquad z_{ij} = -z_{ji}$$

where $K$ is the correlation matrix between all the signals and $K_{ij}$ is the correlation between $x_i$ and $x_j$. The presence of the $\text{sign}(\cdot)$ is necessary when the considered data are negatively correlated. While other choices for making signals comparable can be employed (for example allowing a softer difference or by weighting the difference with the correlation value), we decided to adopt the procedure described above for simplicity in the edge deployment.

UNSUPERVISED ANOMALY DETECTION: METHODS Unsupervised AD algorithms can be divided into many categories depending on their working strategy. In this Chapter the authors have chosen to compare six different AD algorithms belonging to the most important families: Proximity-Based; Linear Model; Outlier Ensembles; Statistical methods.

Probabilistic and statistical methods are a class of very general techniques. They usually assume an underlying data distribution. After the parameter training, the model becomes a generative model able to compute the probability of a sample to be drawn from the underlying distribution. The method that we named MAD is a multidimensional extension of the *well-known control chart* based on the median absolute deviation $mad$. Since mean and standard deviation are very sensitive to outliers, it is common practice to set control limits on the basis of $mad$ and $median$ [209] since they are much more robust. Given a window of the signal $x_i$:

$$\text{mad}(x_i) = b \ \text{median}(|x_i - \text{median}(x_i)|), \quad b = 1.4826$$

the anomaly score (AS) for each sample is defined by the MAD algorithm as:

$$MAD(t) = \max_{i \in \{1,\dots,d\}} \left| \frac{x_i(t) - \text{median}(x_i)}{\text{mad}(x_i)} \right|$$

Note that in ideal conditions, *mad* gives a robust estimate of the Gaussian standard deviation. Since the MAD algorithm is based on the median absolute deviation, its AS can be easily interpreted as the relative distance of the samples from the distribution center.

UNSUPERVISED ANOMALY DETECTION: USAGE AND METRICS After the computations, a general AD algorithm returns an index that measures the abnormality level of data, namely the anomaly score of each sample. Data with higher AS are more likely to be outliers. In order to assess which method performs better at detecting anomalies, the AS must be converted in a binary *anomalous/not anomalous* notation with the aid of a threshold. Note that in the real implementation of the algorithm, this threshold is the value above which the CMS raises a malfunction alarm. This threshold setting can be quite challenging, and it is usually a compromise between false positive and false negative alarms.

In presence of an unbalanced dataset, as in the anomaly detection settings where outliers are much less than normal data, the precision (PREC) and recall (REC) metrics are more meaningful than true positive and false positive rate [212]. The precision is the number of true anomalies (i.e. the number of items correctly labelled as belonging to anomalies) divided by the total number of items labelled by the algorithm as anomalies. The recall is defined as the number of true anomalies divided by the total number of measures that are actually anomalies. A common way to make a comparison between the performance of multiple classifiers is by using the F-score or the AUC score: both of them summarize precision and recall measures. The F-score is defined as the harmonic mean between PREC and REC: finding the classification threshold that maximises F-score means finding an optimal compromise between precision and recall; more precisely, according to the F-score, the best method is the one the has PREC and REC values closest to the (1,1). AUC stands for Area Under the Curve and indeed it is the measure of the area enclosed by the curve in the PREC and REC diagram. The bigger the AUC for a specific method is, the better this method will perform on average. The AUC score is a global performance measure that does not depend on a specific threshold, like the one that maximizes F-score. For this reason we have chosen to compare the unsupervised AD methods using this metric instead of the F-score.

ROOT CAUSE ANALYSIS In many applications, interpretability and explainability are fundamental to ensure that relevant actions are enabled in association with the outcome of an AD module [43]. The AD4MPFM can also enable interpretability: the feature design process is able not only to greatly simplify the anomaly detection, but also the Root Cause Analysis (RCA). In this sub-section we will show a new RCA that takes advantage of the preprocessing procedure. The intuition behind this method is the following: assuming that at most one sensor can fail at a time, there exists a direction in the $n$-dimension features space, where anomalies distribute, which identifies the faulty sensor. This technique can be applied to any number of features but can be easily understood in 3 dimensions. If a fault generates on $x_1$, the $n$-dimensional features space $z_{1,2}, z_{1,3}, z_{2,3}$ will show outliers that will propagate along the bisector of the plane $z_{1,2}, z_{1,3}$. On the contrary, the magnitude of the projection along the axis $z_{2,3}$ will be negligible. The *guilty direction* $d$, relative to fault on $x_1$ is $[1/\sqrt{2}, 1/\sqrt{2}, 0]^T$. In the simple case of three signals, the $x_2$-guilty direction is given by $[0, -1/\sqrt{2}, 1/\sqrt{2}]^T$ while the $d$ for $x_3$ is $[0, -1/\sqrt{2}, -1/\sqrt{2}]^T$. In the general case there are as many $d$ as the number of employed signals. If a generic sample $p(t^*) = [x_1(t^*), x_2(t^*), x_3(t^*)]^T$ is labelled as anomaly, the corresponding features $[z_{1,2}(t^*), z_{1,3}(t^*), z_{2,3}(t^*)]^T$ are projected onto the these special directions. The bigger is the projection, the bigger the suspected contribution is to the fault. In general the *root cause* $r$ is defined as:

$$r(t) = \underset{i \in \{1,\dots,d\}}{\arg\max}\ g_i(t)$$

where $g$ is the *guilty score vector* obtained as:

$$g_i(t) = \frac{|d_i \cdot z(t)|}{\sum_j |d_j \cdot z(t)|}$$

The guilty directions matrix $D$ can be obtained by the Algorithm 4.

**Algorithm 4:** Algorithm for the creation of the *guilty directions* matrix.

```
Data: Number of employed signals $n_d$
Result: Guilty directions matrix $D$
Number of features $n_f$ equal to $\frac{n_d^2 - n_d}{2}$;
Matrix initialization with size $n_f \times n_d$;
Find row and column indices of a strictly upper triangular matrix with
  size $n_d$;
foreach Feature indices $i \in \{0, 1, \dots n_f - 1\}$ do
    Take the $i$-th couple of row and column indices $(r, c)$;
    foreach Direction indices $j \in \{0, 1, \dots n_d - 1\}$ do
        if $j$ equal to $r$ then
            $D[i, j] = 1$;
        else if $j$ equal to $c$ then
            $D[i, j] = -1$;
        else
            $D[i, j] = 0$;
        end
    end
end
Normalize the columns of $D$ by the euclidean norm;
```

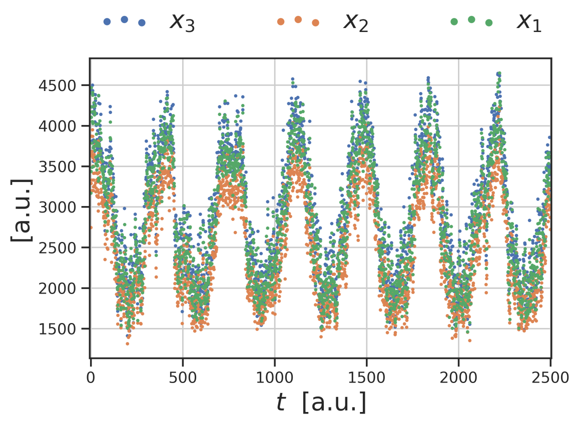


Figure 19: Small interval of the three employed signals. Their high correlation is used by the method to detect possible faults.

CASE STUDY AND EXPERIMENTAL SETTINGS As stated above, the AD4MPFM approach has been developed and tested on a particular CMS that is the MPFM. Real and synthetic datasets have been used to test the proposed algorithm.

In this analysis we have decided to use the three highly correlated signals shown in Figure 19. These signals are close to sinusoidal because they were collected during a *slug* flow: great bubbles of low and high density fluid were alternating [55] [35]. We have chosen these data because here, global and local anomalies are easily identifiable by visual inspection.

### 5.2.1 *Fault Types*

Measuring instruments can incur many types of fault. [73] has isolated four elementary faults: the *bias*, the *drift*, the *precision degradation* and the *complete failure* fault (Figure 20). Although a general event can be a combination of these faults, we have decided to rely on this classification in order to provide a more robust benchmark.

While bias, complete failure and precision degradation do not evolve in time, the drift fault has a linear deteriorating behaviour. This is why in the following paragraphs we will make different analysis depending on the fault type.

### 5.2.2 *Synthetic fault generation*

One of the main challenges in evaluating AD approaches is that tagged data are required, even though typically these are not available. This is the reason why unsupervised approaches are chosen in the first place. In this case study, a number of faults have been added over real MPFM data in order to have a sure evaluation of the effectiveness of the AD4MPFM approach.

The synthetic faults were obtained adding anomalies inside these data, in particular the faults were generated on $x_3$, the earliest signal (Figure 19 and 20). In order to be consistent between different signals and datasets, the anomaly amplitude is proportional to the median absolute deviation mad(·) of the anomalous signal.

Given a signal $x$ belonging to the module $X$ and indexed by the discrete time $t \in [0, N]$, the *bias* fault of module $X$ has been obtained in the following way (Figure 20c):

$$x_{\text{bias}}(t) = x(t) + f_{\text{bias}} \qquad t \sim \mathit{Bernoulli}\{c\}$$

where *Bernoulli* is the Bernoulli distribution, $c$ is the *outlier contamination* and

$$f_{\text{bias}} = A \ \text{mad}(x)$$

$A$ has been named *anomaly amplitude* and takes values in $[0, +)$.

The *complete failure* fault translates to a constant value (Figure 20b):

$$x_{\text{complete failure}}(t) = f_{\text{complete failure}} \qquad t \sim \mathit{Bernoulli}\{c\}$$

where :

$$f_{\text{complete failure}} = \text{median}(x) + A \ \text{mad}(x)$$

Given a normal distribution $\mathcal{N}$, the *precision degradation* fault is defined as (Figure 20a):

$$x_{\text{precision degradation}}(t) = x(t) + f_{\text{precision degradation}} \qquad t \sim \mathit{Bernoulli}\{c\}$$

where:

$$f_{\text{precision degradation}} = k \ A \ \text{mad}(x) \qquad k \sim \mathcal{N}\{0, 1\}$$

Unlike other faults, the *drift* evolves in time:

$$x_{\text{drift}}(t) = x(t) + f_{\text{drift}} \qquad t \sim \mathit{Bernoulli}\{c\}$$

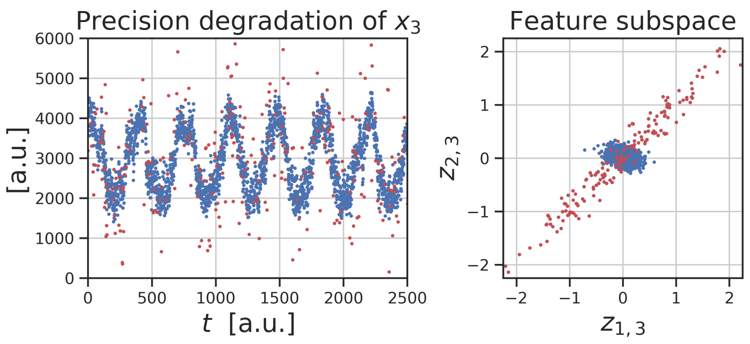
where:

$$f_{\text{drift}} = A \ \text{mad}(x) \ \frac{t}{N}$$

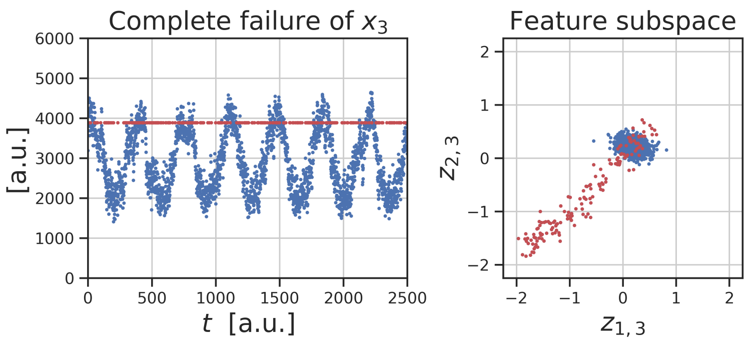
Together with fault type and anomaly detection models, we vary the following quantities in order to provide an extended evaluation of the AD4MPFM efficiency: contamination, anomaly amplitude, window size $\tau$ and window overlap $\alpha$.

### 5.2.3 *Research Questions*

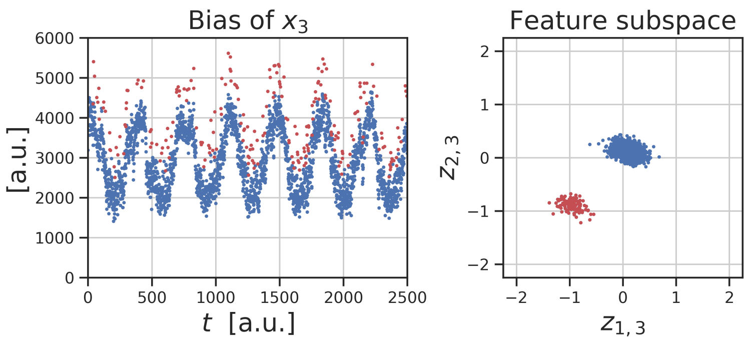
In the design of the experiments we have decided to define some guidelines for investigation. To prove the effectiveness of the AD4MPFM approach and to show the effect of different design choices, we have developed various experiments aiming at exposing the impact of each building block of the AD4MPFM pipeline depicted in Figure 18.



(a) *Sensor precision degradation.*



(b) *Sensor complete failure.*



(c) *Sensor bias.*

Figure 20: (sx) Synthetic anomalous signal obtained starting from the real signal $x_3$ of Figure 19. Contamination 10%, anomaly amplitude 1. (dx) Corresponding feature subspace. Synthetic anomalies are red colored. We highlight the effectiveness of the preprocessing block of AD4MPFM that makes anomalous samples lie on the subspace diagonal.

W.r.t. the *preprocessing* block we will show in the next Section by visual inspection how resulting features do have a discriminative quality.

Regarding *anomaly detection* block, since unsupervised AD methods are not widely used in CMS, we will focus our investigation in showing the performance of various methods, the effect of parameter tuning and the impact on various fault types. In the wide context of Unsupervised Anomaly Detection methods, we will compare 6 different approaches, namely CBLOF, HBOS, IF, MAD, MCD and PCA that are representative of the 4 families described in Section 5.2. In particular, we have chosen approaches that can be implemented in typical CMS scenarios: this will be discussed in more details in Section 5.3. In particular, concerning the *static* faults, namely the ones that do not evolve in time (complete failure, bias, precision degradation), we will try to answer the following questions:

1. Given an anomaly amplitude, how does the contamination affect the performances of each model?
2. Given a contamination, how does the anomaly amplitude affect the performances of each model?
3. In the general case where both the anomaly amplitude and the contamination can vary, which is the model that behaves better?

On the contrary, for the drift case we will address other questions that will be answered in paragraph 5.3.5:

1. Given a contamination value and some windowing settings, which is the model that performs better along the windows?
2. Are there any differences if the contamination changes?
3. How is the optimal threshold affected by the contamination?

Later, in Section 5.3, we will wonder which is the best AD method amongst the ones shown in Section 5.2, given the computational complexity.

Finally, w.r.t. the *Root Cause Analysis* block, we will perform analysis by visual inspection that show the effectiveness of the guilty direction procedure.

## 5.3 RESULTS

### 5.3.1 *Preprocessing*

We remark that, in the AD4MPFM pipeline, the goal of the preprocessing block is to decouple the observed dynamics of the underlying process from the behaviour of the metrology instrument; in order to demonstrate such capacity, we report in Fig. 21 an example of the time series evolution of the MPFM raw data. Moreover, in Figure 22 scatterplot matrices of the same data are reported, by also showing the effect of the preprocessing steps. In particular, in panel (a) of Figure 22, impedance, gamma and Venturi module measurements are reported. It can be easily appreciated that the 2 dimensional distributions

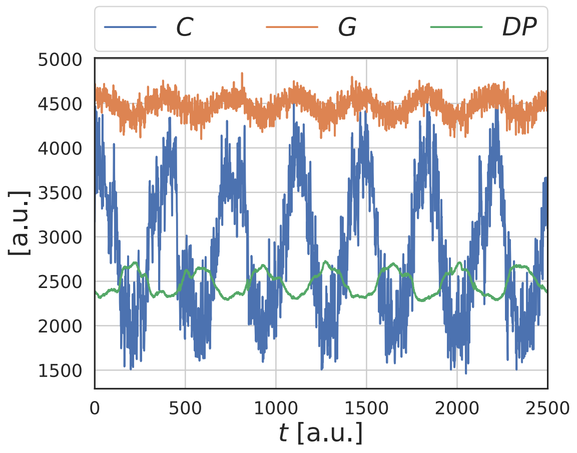
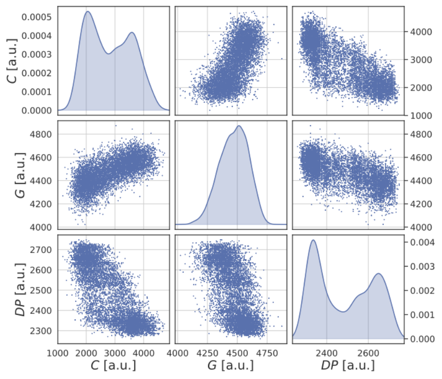
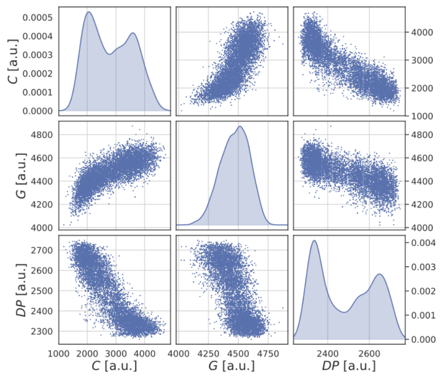


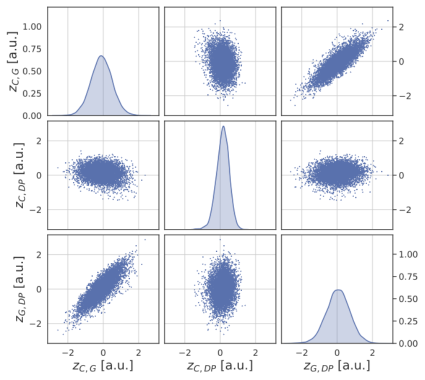
Figure 21: Real Data: raw signal example. Signals from different modules are not aligned in time and have different scales.



(a)



(b)



(c)

Figure 22: Real data: *physics dynamics* and *meter behaviour* decoupling of signals shown in Figure 21. *(a)* Scatter plot of the signals. They can draw very different patterns depending on their value, misalignment and correlation. *(b)* Signal alignment. It always improves the correlation between the signals. *(c)* Data after the process filtering. The measured dynamics is filtered and only the instrumentation behaviour is shown.

between two variables are not gaussian, making it difficult to implement self-diagnosis procedures. In panel (b), showing the data after the alignment step, it can be seen how crosscorrelations have been enhanced. Finally, in panel (c) it is evident the benefit of the feature design step: features now exhibit gaussian distributions that will ease the detection of CMS anomalies as expected.

The effectiveness of the proposed preprocessing box can be appreciated even more in the semi-synthetic data reported in Figure 20. The simulated faults are shown on the left panel while the corresponding data, after the feature design process, are shown on the right. The corrupted data are colored in red. It is clear the advantage given by the proposed manipulation: local, global and context anomalies can be more easily identified. Note that the few corrupted points that cannot be distinguished, are the ones were the added noise is almost equal to zero.

While the signals shown in Figure 20 represent only some of the many trends that the flow can develop depending on the operating point, we have seen that the preprocessing block of AD4MPFM is effective in any of the observed conditions: unfortunately, we cannot report here for conciseness other examples, but such effective behaviour is demonstrated by the AD results reported in the following experiments that are based on this preprocessing step.

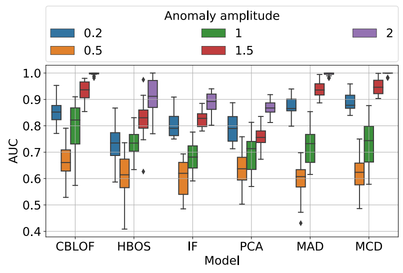
We stress the fact that the preprocessing phase greatly simplifies the task of detecting outliers: instead of monitoring a continuously changing complex signal pattern, the unsupervised AD methods has only to detect the outliers that lie outside the unique central cluster.

### 5.3.2 *AD module: Complete failure*

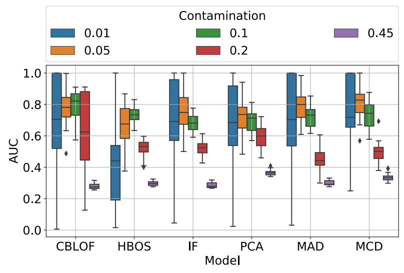
During a complete failure fault, the anomalous signal keeps a constant trend. An example of feature subspace $(z_{1,3}, z_{2,3})$ containing this type of anomalies is shown in Figure 20b. The fault was generated on $x_3$: the correct samples gather in a central cluster, while the anomalous samples distribute on the diagonal. The majority of them can be easily distinguished by the normal cluster, indeed the only anomalies that cannot be detected are those within the normal process dynamics. Beware that the normal cluster width and position can be affected by the normalization phase: if the *reference window* is seriously corrupted, the reference median and mad can be biased. For this reason, care must be taken on the choice of the scaling references.

QUESTION 1 Given an anomaly amplitude and for small values of contamination, the best methods seems to be MCD and MAD (Figure 23b). Increasing the contamination, the performances of almost every method get worse: only the CBLOF seems not to be affected by the anomaly concentration. At very high contamination all the methods have a very low AUC.

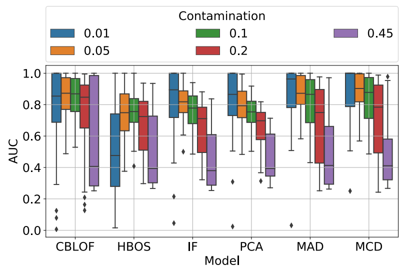
QUESTION 2 Fixing the contamination, the minimum AUC value is obtained for anomaly amplitude between 0.5 and 1. CBLOF performs better, followed by MAD and MCD. PCA is among the worst (Figure 23a).



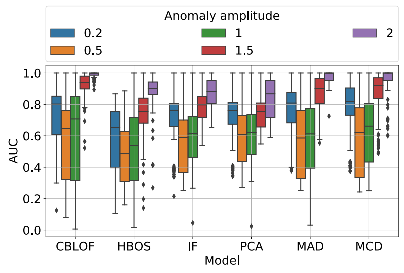
(a) *Fixed contamination %10. The anomaly amplitude varies.*



(b) *Fixed anomaly amplitude 1. The contamination varies.*



(c) *General case, where both the anomaly amplitude and the contamination vary. Contamination view.*



(d) *General case, where both the anomaly amplitude and the contamination vary. Anomaly amplitude view.*

Figure 23: Complete failure.

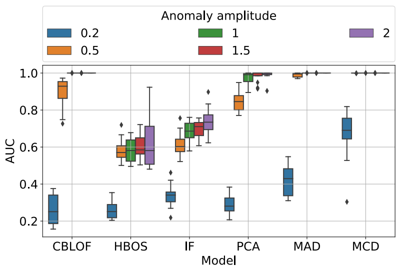
QUESTION 3 In the more general case (Figures 23c and 23d) both the amplitude and the contamination vary. There, CBLOF performs substantially better and has less deteriorating capabilities; it seems not to be robust only at high contamination level.

### 5.3.3 *AD Module: Bias*

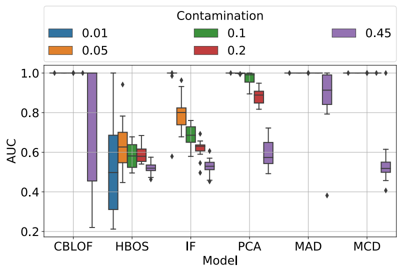
In the bias fault of Figure 20c both contextual and global anomalies are present. The surprisingly good ability of the feature design process to distinguish both types of anomalies is shown in the feature subspace. The normal central cluster is still there, but another anomalous cluster is present on the diagonal.

QUESTION 1 At fixed $A$, MAD has less deteriorating performances. CBLOF is very good but, as expected it is not robust at high contamination. IF deteriorates as $c$ increases (Figure 24b).

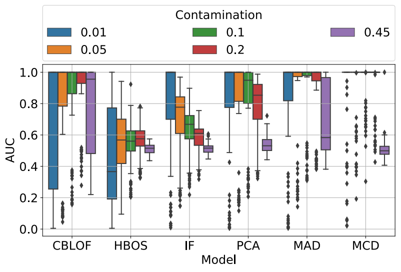
QUESTION 2 Fixing $c$ it is possible to observe many interesting facts: MCD is able to detect the smallest anomalies. MAD performs well but CBLOF recovers quickly. HBOS and IF perform almost the same way, independently on the anomaly amplitude (Figure 24a).



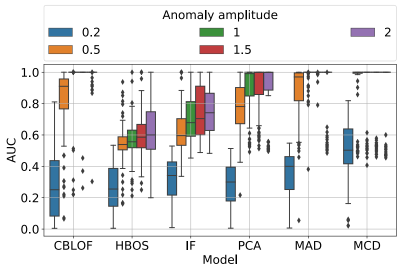
(a) *Fixed contamination %10. The anomaly amplitude varies.*



(b) *Fixed anomaly amplitude 1. The contamination varies.*



(c) *General case, where both the anomaly amplitude and the contamination vary. Contamination view.*



(d) *General case, where both the anomaly amplitude and the contamination vary. Anomaly amplitude view.*

Figure 24: Bias.

QUESTION 3 It is not simple to get conclusions on the general case from Figures 24c and 24d. MCD works better excluding at high $c$. CBLOF and MAD outperform the other methods when the anomalies are quite evident.

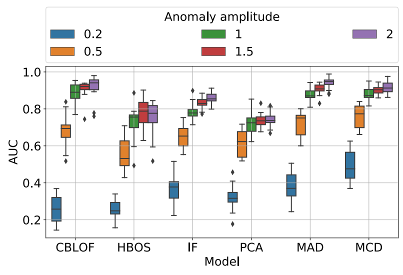
### 5.3.4 *AD Module: Precision degradation*

The feature design process distributes the anomalies on an ellipse centered in the normal cluster (Figure 20a). As before, too small anomalies cannot be detected, but it's not important since they are within the normal signal noise. Note that the central cluster is more centered because these anomalies do not bias the reference median and mad.

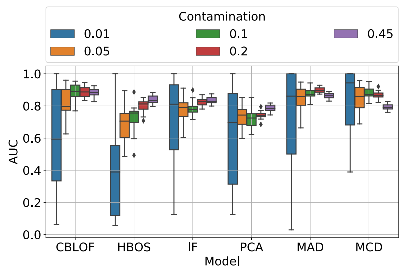
QUESTION 1 While MAD and CBLOF have better performance at high $c$, MCD has the best performance at low contamination level. Increasing the number of outliers, the other methods improve their AUC (Figure 25b).

QUESTION 2 All but PCA and HBOS increase their AUC with the changing anomaly amplitude. As expected MCD detects the anomalies much earlier (Figure 25a).

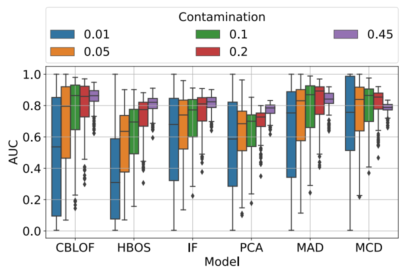
QUESTION 3 When both the anomaly amplitude and the contamination can change, as the contamination increases, CBLOF loses its advantage and is ouperformed by MAD and MCD. When the anomalies are small and few



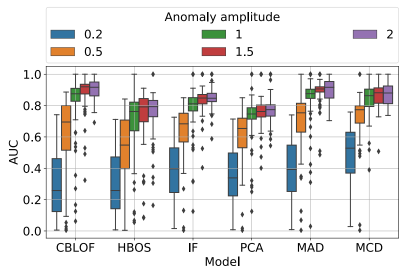
(a) *Fixed contamination %10. The anomaly amplitude varies.*



(b) *Fixed anomaly amplitude 1. The contamination varies.*



(c) *General case, where both the anomaly amplitude and the contamination vary. Contamination view.*



(d) *General case, where both the anomaly amplitude and the contamination vary. Anomaly amplitude view.*

Figure 25: Precision degradation.

MCD outperforms the other methods, otherwise MAD and CBLOF are better (Figures 25c and 25d).

### 5.3.5 *AD Module: Drift*

As drift evolves linearly in time, every window has its own feature subspace (Figure 26). Given a sufficiently small window size, these subspaces can be thought as a sequence of bias faults with increasing anomaly amplitude. In the first window the anomalous cluster is completely inside the normal cluster, but slowly it comes out.

QUESTION 1 As expected by the previous experiments MCD detects the fault earlier and MAD has good performance too. In question 5.3.3 we have already observed that CBLOF recovers quickly and aligns to MCD and MAD. PCA arrives at good AUC values but too slowly. IF and HBOS saturate quickly at the same low value (Figure 27b).

QUESTION 2 Lowering the contamination, the variance of the methods explodes and the AUC score becomes very noisy (Figure 27a). On the contrary increasing the number of anomalies, the AUC of the first window improves and the variance reduces.

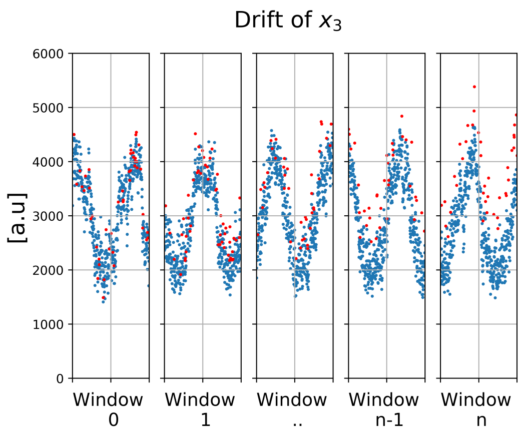
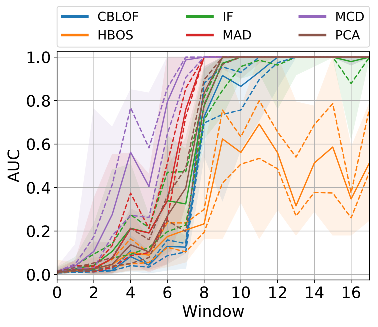
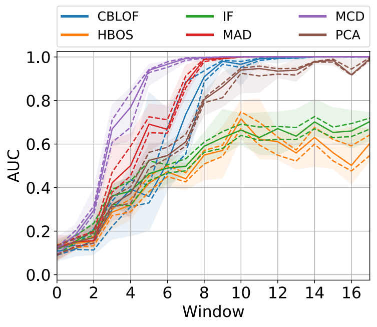


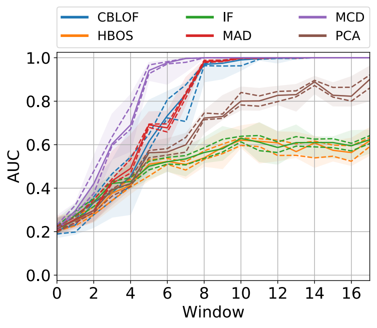
Figure 26: Sensor drift. The signal has been windowed in $n$ segments in order to highlight the $n$ different anomaly settings. The representation of the data in the feature subspace is not reported here for conciseness, but for obvious reasons it would be similar to the one reported in panel $(c)$ of Figure 20 with various anomaly amplitudes.



(a) *1% of contamination.*



(b) *10% of contamination.*



(c) *20% of contamination.*

Figure 27: Drift fault: evolution of the AUC score along the windows. The performances are strongly sensitive to the outliers contamination. The shade extends from the minimum and maximum values of the data, the dashed lines represents the first and third quartiles, while the solid line is the median (second quartile).

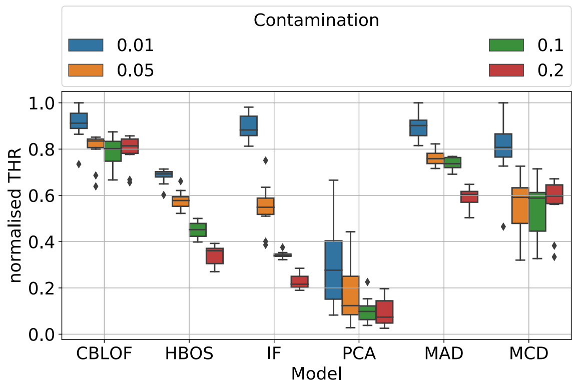


Figure 28: Normalised F1-max-threshold of the last window. Not every method starts from the max threshold because the normalisation has been done on all the windows and sometimes a larger threshold has been used in previous windows.

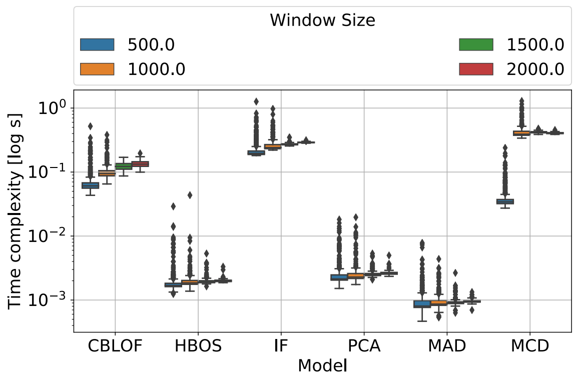


Figure 29: Plot of the computational time employed by each unsupervised method, for different window sizes. Two groups can be distinguished: CBLOF, IF and MCD are very slow while HBOS, PCA and MAD are very fast methods.

QUESTION 3 With *optimal threshold* we mean the one that maximizes the F1-score. In order to make model and window comparisons, this has to be normalized. Fixing the window, the answer to the question 3 is quite simple when looking at Figure 28: if the contamination increases, the threshold has to be lowered. But not all the methods behave in the same way: MCD needs a severe adjustment of the threshold, while CBLOF is less affected by the contamination. In this case PCA performs poorly because it has a huge threshold variation, even inside the contamination classes.

### 5.3.6 *Root Cause Analysis*

The algorithm usually applies the RCA only to the anomalous samples, but in order to understand how this block works we have applied it on every sample. The results are shown in Figure 30. The first fact that can be noticed is the great ability to correctly classify the faulty module; in these examples the anomalies are assigned to the faulty signal $x_3$. The second fact is the great simplicity of

Table 9: Time Complexity [ms].

| Model / Window Size | 500 | 1000 | 1500 | 2000 |
|---|---|---|---|---|
| CBLOF | 60.9 | 96.3 | 121.2 | 140.7 |
| HBOS | 1.7 | 1.9 | 1.9 | 2.0 |
| IF | 195.0 | 243.9 | 270.9 | 297.8 |
| MAD | 0.8 | 0.9 | 0.9 | 1.0 |
| MCD | 33.9 | 399.2 | 417.2 | 422.0 |
| PCA | 2.1 | 2.3 | 2.5 | 2.6 |

the method obtained thanks the proposed feature design: the guilty directions are clearly visible inside the central cluster.

### 5.3.7 *Implementation Discussion*

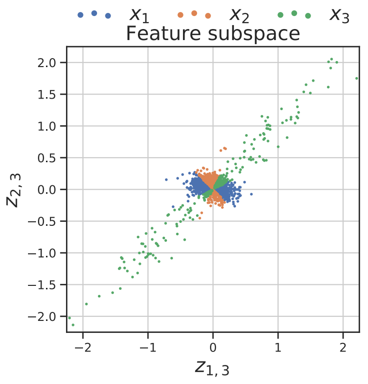
The results obtained on semi-synthetic datasets are promising, as a matter of fact the AD4MPFM algorithm is able to detect with high performances (accordingly to process experts) the generated fault and the faulty module as shown in the previous experiments. The promising results were the basis for proceeding with edge implementation of the AD4MPFM approach in real MPFMs. While the approach has been designed in order to satisfy real-world constrains of CMSs, several aspects should be taken into account before implementation.

*Robustness to outlier severity:* robustness is a key property for applications where the CMS is placed in remote or 'costly' locations. In fact, the MPFM is a typical example, since such CMS is usually placed in harsh and remote sites. In Sections 5.3.2,5.3.3 and 5.3.4 we have tested many AD algorithms changing the type of fault and studying their robustness to different fault contaminations and anomaly amplitudes. MAD seems to be the one that in general is less affected by the changing fault conditions.

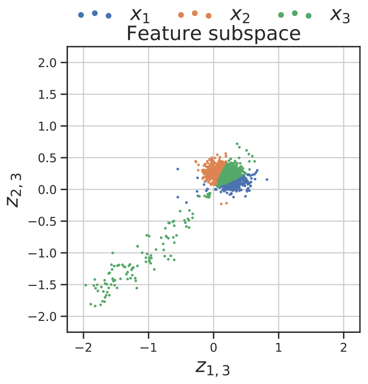
*Threshold setting:* from the implementation point of view, the optimal threshold must be easy to be set and has to be stable; indeed, on edge applications (especially with installations in remote sites) correcting the threshold could be hardly viable. Studying the drift fault we have observed how the optimal threshold behaves and which method exhibit the more stable set up. In our context, CBLOF has the most robust threshold.

*Trade-off between detection abilities and time complexity:* depending on the application at hand, the choice of the optimal AD method can be done by investigating the trade-off between detection accuracy and time complexity. Table 5.3.5 and Figure 29 show the mean computational time[1] needed by the AD algorithm to compute the AS for one window. CBLOF, IF and MCD can be considered *heavy* methods since they are slower by two degrees of magnitude

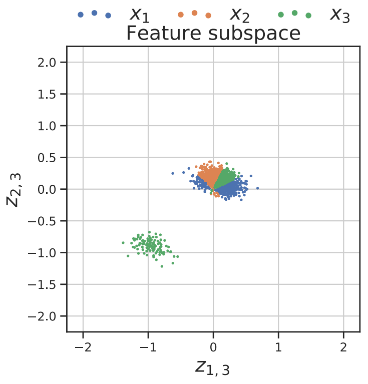
[1]The time complexity of the algorithms was measured on a 2.7 GHz Intel Core 5 Processor and refers to the PyOD implementation available in [273].



(a) *Precision degradation fault.*



(b) *Complete failure fault.*



(c) *Bias fault.*

Figure 30: Root Cause Analysis: The samples in the feature subspace are colored by guilty index. It is clear the simplicity and the effectiveness of the proposed method.

than MAD, PCA and HBOS. Unfortunately MCD was one of the most effective methods but is the worst compared in terms of time complexity. Increasing the window amplitude, there is not a significant increase in computational time. Since MAD is much faster than the other methods and has good detecting abilities, it can be considered as a good candidate for maximizing the trade-off between the mentioned properties in many application scenarios.

*Simplicity of the coding implementation:* Simplicity of the coding implementation is a key factor in applications that do not use high level programming languages. MAD is not only the fastest algorithm, but it is also the simplest to be implemented: MAD does not need complex computations and can be easily written in any coding language.

*Interpretability:* as stated before, interpretability is a key property for monitoring and maintenance-related intelligent tool. For some applications, where interpretability is more relevant than other qualities, MAD is the recommended amongst the considered AD approaches. MAD threshold is easily interpretable (see Section 5.2). Nevertheless, other AD methods can be equipped with interpretability procedure [43], however this will require additional complexity that may not be feasible in some application scenarios.

## 5.4 CONCLUSION

In this Chapter, we have proposed a novel approach for the self-diagnosis and anomaly detection of Multiphase Flow Meters, named AD4MPFM. The approach is specifically designed for handling the typical constrains of complex metrology instruments like the MPFM that are based on multiple sensing modules, time-series streams and data fusion. The building blocks of AD4MPFM are modern unsupervised anomaly detection techniques and a preprocessing pipeline able to decouple the complex dynamics of the underlying process with the quality of the metrology tool behaviour. Moreover, given the importance of interpretability and explainability in monitoring and maintenance contexts, we have equipped AD4MPFM with Root Cause Analysis capabilities, thanks to a so-called guilty direction that points out the module responsible for an anomaly. The main advantage of AD4MPFM is the fact that it does not need historical data to be trained and it is ready for Plug & Play implementations.

The effectiveness of the proposed algorithm has been tested both on semi-synthetic and real datasets. Many types of faults have been simulated, changing the severity of the faulty conditions. As discussed, MPFMs can be very different: therefore, the various desired properties (robustness, accuracy, complexity, etc.) for the self-diagnosis system can vary in order of importance. In this regards, as shown by the experiments reported in this work, the AD4MPFM procedure allows to adopt various design choices in order to make some properties better satisfied than other. As future work, a research direction could be to adopt an *ensemble* approach on the employed Anomaly Detection methods in order to maintain all the good properties of various unsupervised techniques.

A further remark is that, as shown by the reported experiments, a simple approach like MAD is still very accurate in detection, proving the effectiveness of the preprocessing procedure and on the whole AD4MPFM pipeline.

Additional research investigations will be focused on other four aspects: (i) validation of the proposed approach on new Complex Measuring Systems real case studies; (ii) the employment of deep learning approaches for anomaly detection that will fit the IoT scenario, where the edge implementation constrain can be relaxed, iii) exploration of new approaches in order to reduce the dependence of the algorithm from normalization and alignment, iv) analysis of models that do not need to explicitly extract features from the signals but rely on distances and similarities like the one exposed in [182]. Indeed, the feature extraction process is a delicate step: in some scenarios the extraction of significant features could be very complex or the extraction process could be so coarse that the information encoded in the signal is lost.

# 6 ISOLATION FOREST ON RESOURCE CONSTRAINED DEVICES

This work tries to address the limited computational capabilities of industrial devices where it is important to detect possible anomalies. An example is again the MPFM because, even if provided with discrete computational resources, only a small part of them can be devoted to Anomaly Detection tasks. The ideas developed in this Chapter were published in [28].

## 6.1 INTRODUCTION

Over the last few years, the cost of sensors and microprocessors (MCUs) have significantly decreased; moreover, the new technological scenarios brought by the Internet of Things (IoT) and Industry 4.0, are pushing to the embedding of such sensors and MCUs in an increasing number of systems and devices with *edge computing* capabilities. On one side, the combined availability of sensing and computational capabilities into local devices paves the way for new applications [196], like for example the automatic monitoring of the data sensed by edge computing devices [74] by means of Machine Learning (ML) approaches; on the other hand, such new scenario inspires the research community towards the development of algorithms able to run ML models onto these ultra-constrained devices [23]. Low resources ML models need to be as light as possible in order to fit the available memory and to be computed on MCUs, moreover they need to efficiently handle multidimensional data that come from a variety of sensors that might be linked to the board.

Concerning the computing paradigm, at the moment we are witnessing a change in the considered architectures: traditional IoT-designed ML models usually heavily rely on cloud computations with resulting latency, bandwidth and privacy concerns that are nowadays representing in many cases an obstacle to the adoption of such solutions [215]. We now instead see the increase of a new paradigm that comes under the name of *TinyML*, where computations are done on the edge tiny devices like MCUs [118]. This change allows to drastically reduce the latency and the energy consumption caused by the transmission process, and to send to the cloud only the necessary data packet, enhancing the system security. Unfortunately these improvements come at the cost of stricter constraints on the memory and complexity a model can handle to run on these devices: the memory capacity goes from some *gigabytes* (cloud GPUs) to *kilobytes* (MCUs), with coherent computational speed scaling [22].

In the context of this work, we focus on a particular ML application, Anomaly Detection (AD), that is gaining increasing attention in past recent years. AD algorithms are particularly useful in order to monitor large amount of data [76], and to provide efficient feedback on the data reliability; these models are specifically designed to find unusual patterns inside data that are generated by complex multidimensional processes. This task is usually *unsupervised*, meaning the labels describing if a sample is anomalous, are few or totally absent. In this context the Isolation Forest (IF) is a very appealing algorithm due to its good detecting performances compared to its algorithmic complexity [210]. However, in the case of ultra-constrained devices, even small improvements in the algorithm can make a difference: the detection performance is only one of the factors that might be considered in the choice of an algorithm to be run on an edge device; other factor might be memory, latency, computational power and energy cost to run on battery power [210].

The goal of this work is to show an algorithm like Isolation Forest that is very cheap to train, could be shrunken in a way that simultaneously reduces its hardware requirements and increases its performances. This can be achieved adding a *weak* supervision in the form of few labels, allowing the forest to be rearranged *without* a proper retraining; indeed the proposed methodology might be used to train a new forest from scratch or to retrofit a previously trained forest with newly obtained labels, similarly to an Online Learning scenario [62, 205].

The relaxation of the unsupervised settings is typically reasonable in the context of Decision Support Systems (DSS). In recent years DSSs became pervasive and today are applied to all fields where complex and delicate decision have to be made to assist human operators in the decision-making process like in medicine [233], precision agriculture [143], energy [189], environment [56] and security [147, 250]. These systems are equipped with anomaly detection functionalities that allow to automatically monitor the process, giving feedback and alerting the human user to make a possible action; in this context end-users can provide feedback on anomaly detection module suggestions [217], making the weakly-supervised scenario that is considered in this work, reasonable.

Moreover, it has to be taken into account that the development of effective AD model is typically a collaborative and *iterative* process between data scientists (AD developers) and end-users (AD users): the firsts choose the algorithm that best fits the given requirements, while the latter ones evaluate if the model ranks the anomalies according to the their expectations. This is necessary since outliers are not uniquely defined but they depend on the user and the context [217], therefore requiring different detection strategies [26]; for a more detailed dissertation we refer the interested reader to [217].

In this context, we propose here the TiWS-Isolation Forest (TiWS-IF) algorithm that is intended to dialogue with the DSS, following the TinyML paradigm: the DSS system shares with the model the available annotations provided by the end-user, in a weakly supervised fashion [41, 150], that can be exploited to enhance performance and reduce complexity. Indeed, by assuming that a first model is trained on a fully unlabelled dataset and put in operation on a DSS, during its lifetime it is not unlikely to collect some weak supervision in the form of few labels that can be used to improve the existing detection algorithm. This allows the model to adapt to the user definition of *true* outliers, giving a more domain-specific prediction of outlierliness [61, 62]. To the best of our knowledge, TiWS-IF is one of the first approaches in the Isolation Forest literature designed to work in a weakly-supervised scenario and to enforce the user-definition of anomaly in an iterative way. Moreover it is the first that, exploiting the available knowledge, reduces the algorithm complexity in view of edge implementations.

This Chapter is organized as follows: in Section 6.2 the Isolation Forest algorithm is described focusing on the algorithmic complexity and the ensemble strategy; the datasets employed to test the proposed strategy is described in the same Section. Then Section 6.3 goes deeper in the analysis of the algorithm describing some issues related to the standard training procedure and showing many examples. Starting from this point, Section 6.4 describes a weakly-supervised algorithm whose goal is to overcome the previously mentioned obstacles. Here a number of results concerning the application of this new algorithm to real world datasets are shown, proving the effectiveness of the proposed approach. In the final Section the work is summarized and future improvements are discussed.

## 6.2 ISOLATION FOREST ALGORITHM AND PERFORMANCE METRICS

Isolation Forest [162] is an ensemble of binary trees named *isolation trees* since their goal is to isolate data points. As already stated in Chapter 4 this algorithm relies on the assumption that anomalies are few and different from normal points, and that recursive space partitioning should isolate anomalous data points (outliers) in an easier way w.r.t. *normal* data points (inliers). This is done by means of an isolation tree that recursively splits the space, choosing randomly with uniform probability the feature and the threshold where to split the space [244]; this process is repeated until every point is isolated in a leaf, or the isolation tree reaches a maximum depth. Since it is reasonable to expect that anomalies are isolated faster than inliers, their path length along the tree should be smaller when compared to normal data points. This allows to define an anomaly score $s(\cdot)$ as:

$$s(x) = 2^{-\frac{E[h(x)]}{c}}$$

where $x$ is the input data point, $c$ is a normalising factor representing the average depth of a binary tree and $E[h(x)]$ is the expected isolation tree depth reached by the point $x$. Then, in many applications, the anomaly score is transformed in a binary label by means of a threshold $\tau$ (usually 0.5): if the anomaly score associated to a point is higher than the threshold, the point is flagged as an anomaly, otherwise is considered normal.

As previously mentioned, IF is an ensemble of different isolation trees that are constructed in a way to guarantee robustness also in the presence of random choices that are present in the isolation procedure: each isolation tree is trained using a bagging strategy, i.e. by using different sub-samples of the same dataset. Experimentally, the IF authors suggested as a guideline to use $t = 100$ trees with sub-samples of $\psi = 256$ points for obtaining a stable estimate $E[h(x)]$ using the sampling mean:

$$\overline{h}(x) = \frac{1}{n_T} \sum_{t}^{n_T} w_t h_t(x)$$

where every isolation tree is implicitly assumed to be equally informative, and therefore weighted by the same constant value $w_t = 1$. The aforementioned choices for $n_T$ and $\psi$ are typically adopted by many authors in the literature and they are the default choice in many libraries implementing IF.

Bagging allows IF to achieve linear time-complexity together with small memory requirements [162], that are very interesting properties when considering *tiny* implementations on MCUs. Indeed the time complexities in the training stage is $O(n_T\psi \log \psi)$ while in the testing is only $O(nn_T \log \psi)$ where $n$ is the number of tested instances [160]; the memory requirements are bounded by the number of nodes of each isolation tree, that is $2\psi - 1$ and the number of trees in the forest $n_T$ [162], therefore are $O(n_T\psi)$.

In the rest of the Chapter, the original Isolation Forest algorithm with 100 randomly grown trees is referred as the *standard* or *original* one.

Any anomaly detector returns a continuous score associated to each point that should reflect its degree of anomaly and it is called *anomaly score (AS)*. To convert the AS into binary labels (anomaly/inlier) a threshold is set: points with an AS above the threshold are predicted as anomalies, and vice versa. By varying this classification threshold it is possible to measure a different true positive rate (TPR), false positive rate (FPR), precision and recall. However, to understand the global performances of a model it is necessary to summarize these quantities into an unique value: the area under the curve TPR-FPR curve (ROC AUC score), the average precision $\overline{p}$, and the F1-score are some of the most popular choices. The preferred metric employed in this work is the average precision:

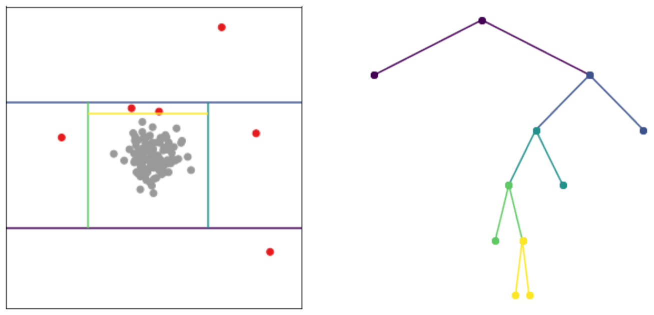
$$\overline{p} = \sum_{i}^{n_\tau} p_i(r_i - r_{i-1}),$$

that summarises the precision $p_i$ and recall scores $r_i$ and it is better than the area under the ROC curve when the dataset is highly unbalanced [212]. The F1-score is a valuable metric that summarizes the precision and recall by taking their harmonic mean.

## 6.3 ANALYSIS OF THE STANDARD ISOLATION FOREST ALGORITHM

In the previous Section, Isolation Forest is described as an ensemble i.e. a collection of *weak* learners (isolation trees) that are trained following a completely *random* procedure. This might induce to believe that all of the learners have similar impact and averaging their contribution is the best possible approach, but is it true? Are the isolation trees similarly informative?

A first evidence that this is not the case can be seen in Figure 31a and 31b where two isolation trees are constructed on the same toy dataset composed of


(a) Good random isolation: the splitting values lie in between normal and anomalous values.

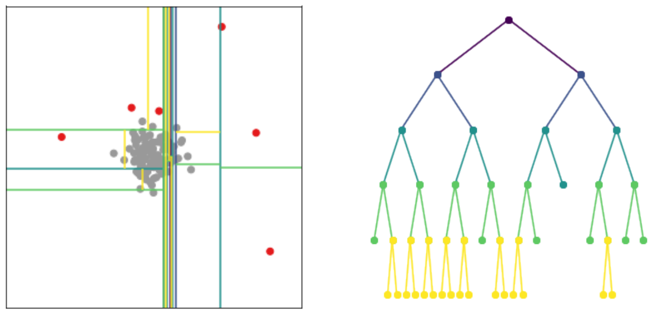

(b) Bad random isolation: the splitting process focuses on normal data and ignores anomalous ones. This leads to not effective isolation.

Figure 31: Two different isolation trees grown on the same dataset. The anomalies included in the dataset are depicted in red.

a unique central normal cluster and some quite simple anomalies. It is clear that while the first isolation tree makes use of its partitions in an effective way isolating quickly the anomalies, the second focuses too much on normal points and therefore it looses many chances to isolate data.

From the previous qualitative example it seems that not all the isolation trees have the same role in the IF and their equal weighting might not be the best choice. To get a quantitative feeling of the previous intuition it was settled a more rigorous experiment: 100 isolation trees were randomly grown following the *standard* procedure, creating the forest named $F_{100}$. Then each tree was individually tested using the available labels together with the average precision $\overline{p}(\cdot)$ and the histogram of their average precision plotted in Figure 32b. From this picture it easy to see that not all the isolation trees behave in the same way: the range between the best and the worst is quite large and the majority lies in between.

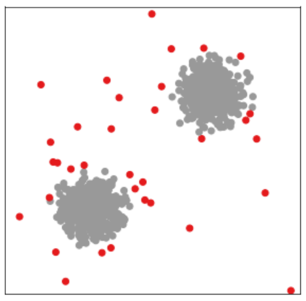
This analysis led us to the following idea, which is at the core of the proposed TiWS-IF algorithm: why not to exploit the available weak supervision in order to sort the trees and possibly to get only the best performing isolation trees? This might reduce the model complexity, following the TinyML paradigm, and fine-tune the detection algorithm towards the outlier definition expected by the end-user of the DSS.

With this idea in mind, we made additional analysis by sorting the isolation trees according to three ordering strategies: i) the *best* strategy, the *worst* strategy and the *random* strategy that are described in the next few lines. The *best* consists in sorting the isolation trees according to their $\overline{p}$ score in descending order i.e. from the best tree to the worst; the *worst* strategy is the opposite and sorts the trees from the worst to the best. The *random* instead, chooses a random permutation of the trees and therefore it simply shuffles them.

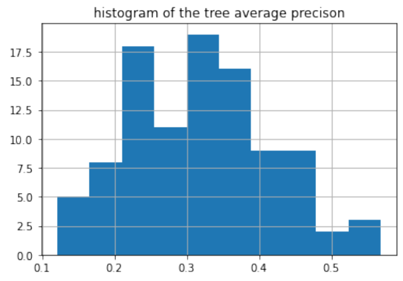
Using the *best* strategy, 100 different isolation forests were built using an increasing number of trees: the first forest contained only the best isolation tree, the second one only the *two* best trees and so on, until the 100-th forest contained all the 100 trees like the *standard* Isolation Forest. At each iteration the average precision of the forest was measured, leading to the blue line in Figure 32c. The same was performed with the *worst* strategy (orange line) and with the *random* (green dashed line); however, since the *random* permutation is a non-deterministic strategy, it was repeated 100 times in order to get stable results and to draw the green area.

Figure 32c shows quite interesting results that anticipate some aspects that are also visible in real world datasets: the first and most evident is that good performances can be reached with just 5-40 isolation trees instead of 100. This means that many isolation trees are just overabundant or little informative. The three lines obviously terminate with the same value (the *standard* Isolation Forest performances) but follow quite different paths: the blue line starts very well and rapidly reaches $\overline{p}_{100}$ while the others require many isolation trees to reach appropriate average precision.

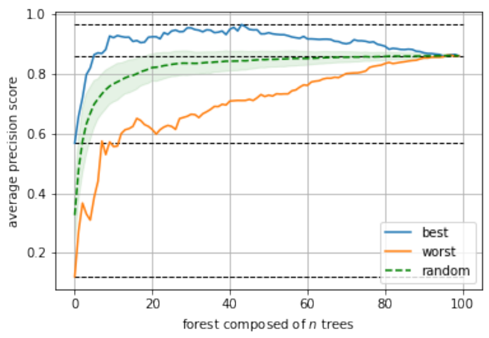
Other interesting results can be observed in Figure 47 where a second toy dataset is considered: a toroidal dataset with some anomalies in its center; also in this case, the same type of experiments, previously performed on the first toy dataset, were considered, but the previously discussed aspects became even more evident. First of all, the random training procedure generates a lot of very ineffective isolation trees, and very few good ones are present in Figure 33b. This behaviour directly reflects on the IF construction: not only the *standard* performances were reached very quickly like in the previous toy example, but the best *achievable* performances are much higher than the standard ones. In this example, the best isolation tree alone is better than the whole forest, and with few more trees the forest can reach even better results; indeed on a scale between 0 and 1 with very poor *standard* performances around 0.2, the best achievable performance exceeds 0.7. Unfortunately, due to the corrupting effect introduced by the bad isolation trees on the left of Figure 33b, adding more trees means poisoning the solution leading to quite bad *standard* results (Figure 33c). Actually this effect is visible in Figure 32c too, but is less evident and the gap between best performances (blue line) and the mean performances (green line) is smaller. This suggests that in many datasets the IF results may be improved, depending also on the structure of the dataset itself.



(a) Simple dataset made up of two clusters and some scattered anomalies.

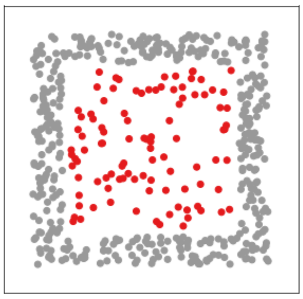


(b) Histogram of the average precision scores obtained measuring the performances of the isolation trees.

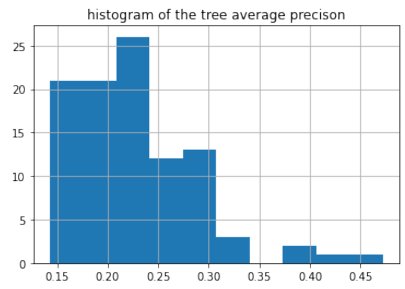


(c) Average precision of different forests built with different strategies.

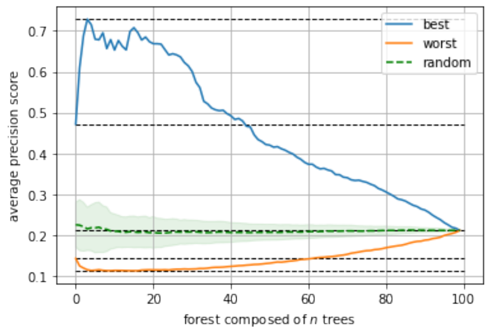
Figure 32: Toy example: double cluster dataset.



(a) Normal data organized in a square toroid and anomalous points inside.



(b) Histogram of the average precision scores obtained measuring the performances of the isolation trees. There are many very bad trees and some few good ones.



(c) Average precision of different forests built with different strategies.

Figure 33: Toy dataset: square cluster. A more complex example where the advantages of carefully choosing the best isolation trees makes a huge difference in performances.

### 6.3.1 *Real world datasets*

We applied an experimental procedure similar to the one previously described for toy datasets, on real-world data: such data were retrieved in [204] and are adaptations of the UCI Machine Learning datasets [72] for the Anomaly Detection task. Such datasets consists of labelled data coming from different domains:

- biomedical (*annthyroid*, *arrhythmia*, *breastw*, *cardio*, *mammography*, *pima*, *thyroid*, *vertebral*);
- environmental (*cover*, *ionosphere*, *satellite*, *satimage-2*);
- human language (*letter*, *mnist*, *optdigits*, *pendigits*, *pendigits*, *speech*, *vowels*);
- others (*musk*, *shuttle*).

| | # data | # features | # anomalies | contamination % |
|---|---|---|---|---|
| annthyroid | 7200 | 6 | 534 | 7.42 |
| arrhythmia | 452 | 274 | 66 | 14.60 |
| breastw | 683 | 9 | 239 | 34.99 |
| cardio | 1831 | 21 | 176 | 9.61 |
| cover | 286048 | 10 | 2747 | 0.96 |
| ionosphere | 351 | 33 | 126 | 35.90 |
| letter | 1600 | 32 | 100 | 6.25 |
| mammography | 11183 | 6 | 260 | 2.32 |
| mnist | 7603 | 100 | 700 | 9.21 |
| musk | 3062 | 166 | 97 | 3.17 |
| optdigits | 5216 | 64 | 150 | 2.88 |
| pendigits | 6870 | 16 | 156 | 2.27 |
| pima | 768 | 8 | 268 | 34.90 |
| satellite | 6435 | 36 | 2036 | 31.64 |
| satimage-2 | 5803 | 36 | 71 | 1.22 |
| shuttle | 49097 | 9 | 3511 | 7.15 |
| speech | 3686 | 400 | 61 | 1.65 |
| thyroid | 3772 | 6 | 93 | 2.47 |
| vertebral | 240 | 6 | 30 | 12.50 |
| vowels | 1456 | 12 | 50 | 3.43 |

Table 10: Summary of the main characteristics of the real word dataset employed in this Chapter.

In Table 10, the datasets are all summarized by providing the number of samples, features, anomalies and the contamination, i.e. the percentage of anomalies inside the datasets. The number of samples varies from few hundreds (*vertebral*) to hundreds of thousands (*cover*), while the number of features starts from 6 to 400 in the *speech* dataset. However, the most important characteristic in this context is the contamination: some datasets have less than 1% of anomalies (*cover*) and reach above the 35% (*ionosphere*). All of them have a number of anomalies that exceeds 30 in order to obtain reliable results in the following analysis.

In Figures 34 and 35 the experimental results with the real world datasets are reported, showing similar traits with the ones obtained with toy datasets: there seems to be two peculiar behaviours, one described by a logarithmic-shape curve (for example *breastw* and *satimage-2*) and the other by a bell-shape (like *cardio* or *vowels*). Both of them reach the best performance very quickly, i.e. with 5-20 isolation trees, but the bell-shaped starts with higher performances than the standard, reaching very fast the best *achievable* scores and then degrading. In this case, the gap between best and average results (blue and green lines) is very large, suggesting the *standard* IF algorithm might have a lot of room for improvements. It is not clear the underlying motivation of these behaviours: we expect them to be dependent to the structure of the dataset and the definition of outlier that, as explained in the Introduction, it is very dependent on the domain and the end-user expectations.

## 6.4 WEAKLY-SUPERVISED ALGORITHM

One may ask how the analysis reported in the previous Section can be exploited in order to improve the original algorithm, and if these results are just over-fitted, meaning the best isolation trees here obtained are valid only for the employed data, or if they can be generalized to new data points. In other words, can the procedure that selects the best isolation trees lead to over-fitting results? Or, on the contrary, might such procedure be used to learn the best performing trees on a portion of the dataset and to apply those trees on other data coming from the same dataset distribution? This procedure might help in many ways: it can be used to reduce the number of trees and therefore the memory and power consumption of the algorithm, but it might also be used to increase the average performance of the forest.

Based on these questions, a new weakly-supervised algorithm, called TiWS-IF, was designed and tested on multiple real word datasets. Starting from an unsupervised training of the Isolation Forest, the TiWS-IF algorithm use the available weak supervision provided by the domain expert to choose the best performing trees and consequently the best performing forest. Defining $\overline{p}(\cdot)$ as the average precision scoring function, $\text{argsort}(v)$ the sequence of indexes able to sort the vector $v$ in descending order and $\text{argmax}(v)$ the index associated to the maximum entry of $v$, the procedure is defined in Algorithm 5. This takes as input the standard Isolation Forest $\text{IF}_{n_T}$, i.e. the collection of $n_T$ trees $T_t$ grown with the original unsupervised procedure, and the small set of supervised data

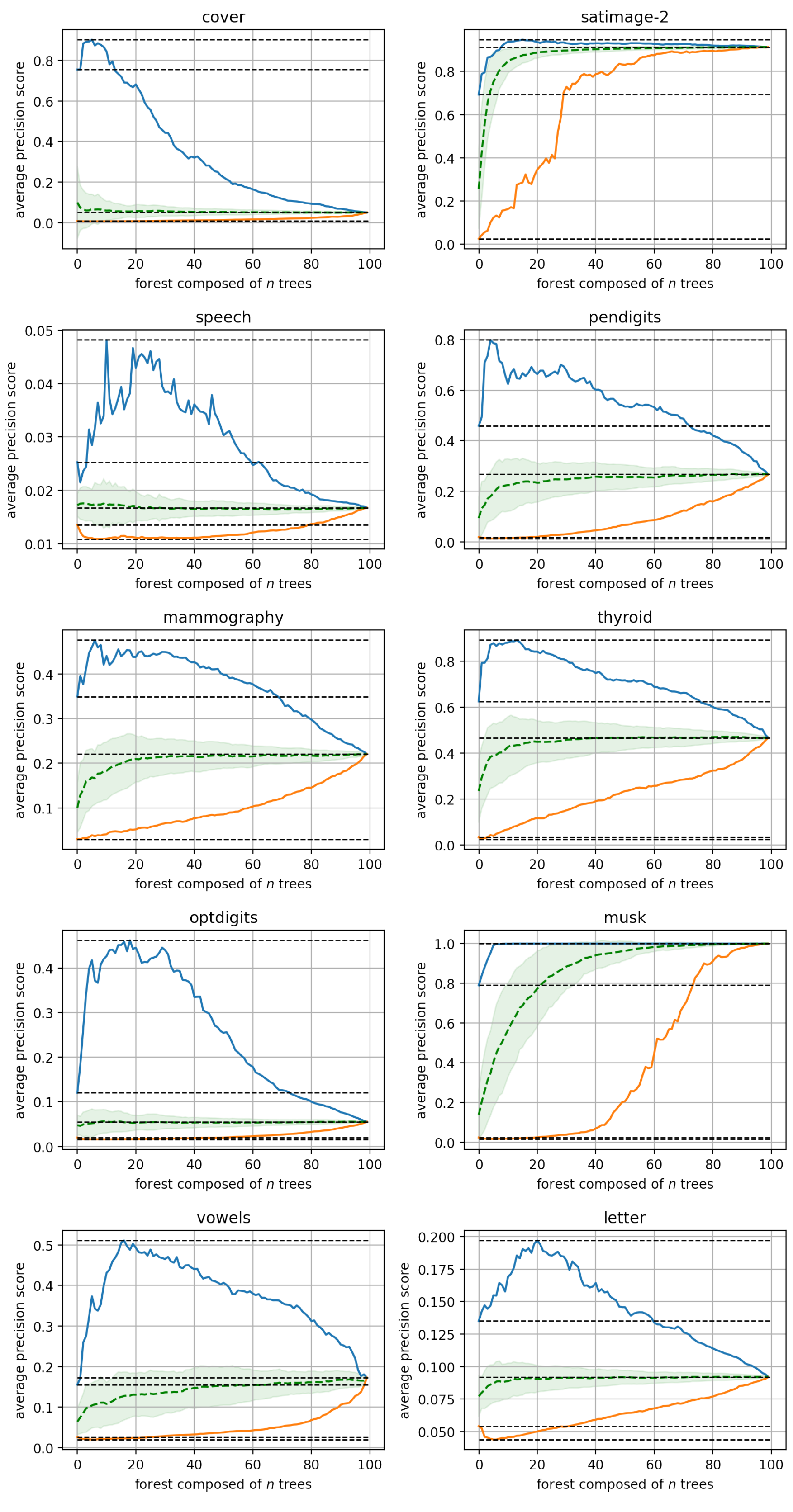


Figure 34: Real-world examples: datasets with smallest contamination. It can be seen how in some cases the gap between the random results (green distribution) and best achievable (blue line) is quite large, meaning that there is a lot of room to improve the original algorithm.

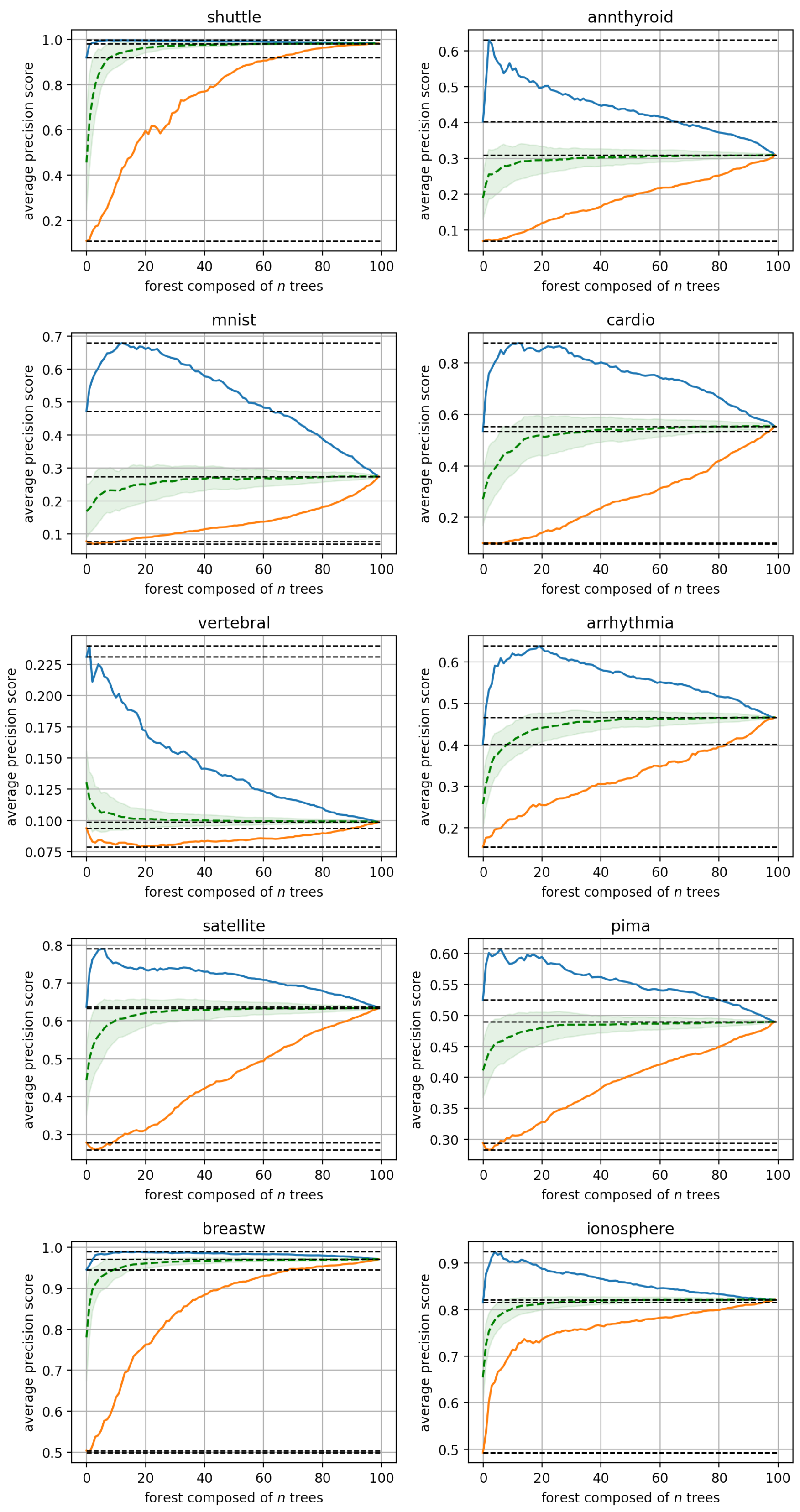


Figure 35: Real-world examples: datasets with highest contamination. It can be seen how in some cases the gap between the random results (green distribution) and best achievable (blue line) is quite large, meaning that there is a lot of room to improve the original algorithm.

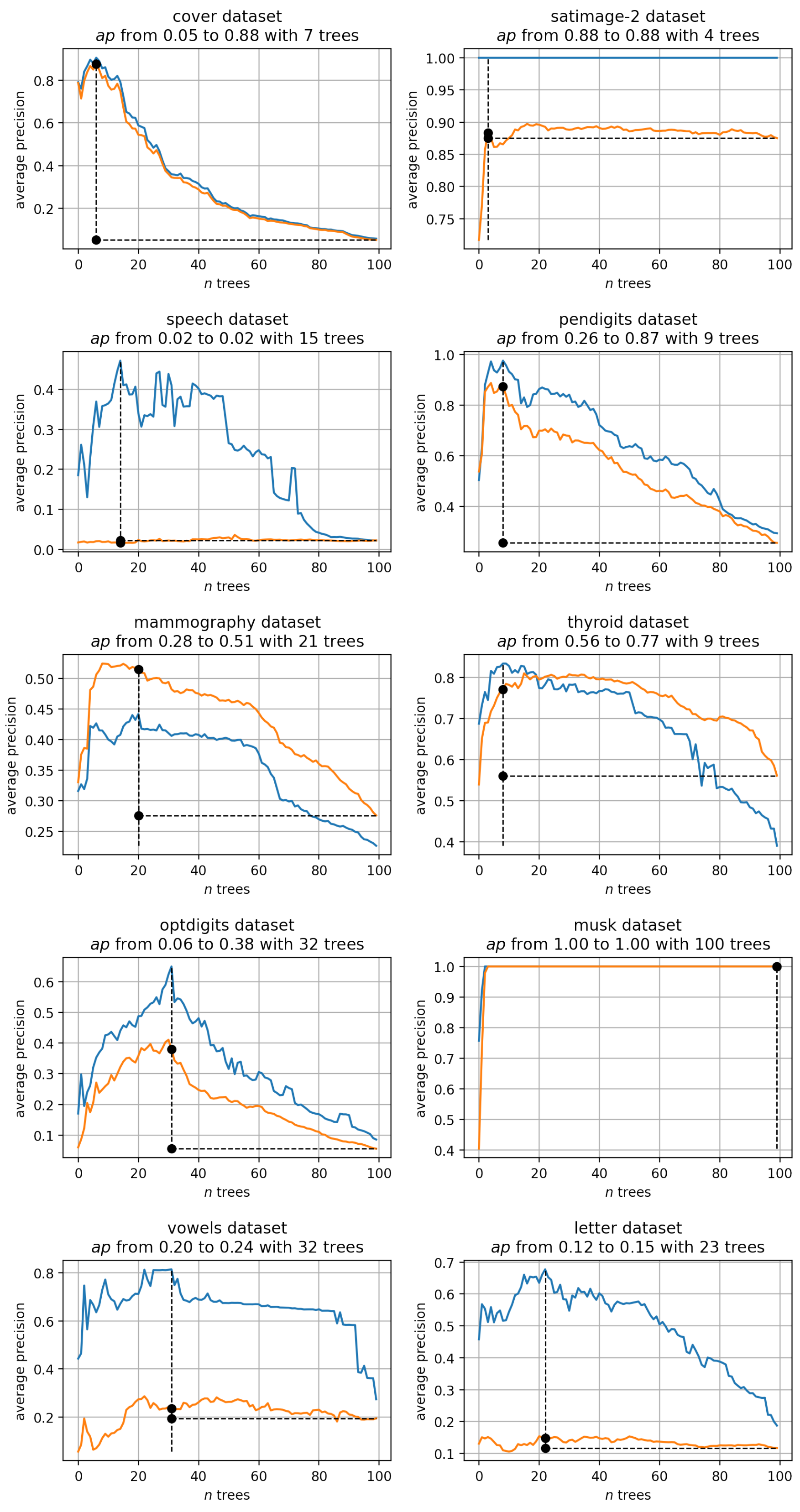


Figure 36: Application of the proposed algorithm to the real word datasets with the smallest contamination, with supervision applied to 20% of the training dataset. The blue curve shows the performance of the algorithm on the supervised part of the training dataset, while the orange are obtained on the test dataset (50% of the full dataset). The two black dots highlight: the average precision obtained on the test set obtained with the forest that maximises the average precision during training, the number of trees needed to get these performances, and the average precision of the *standard* algorithm during test.

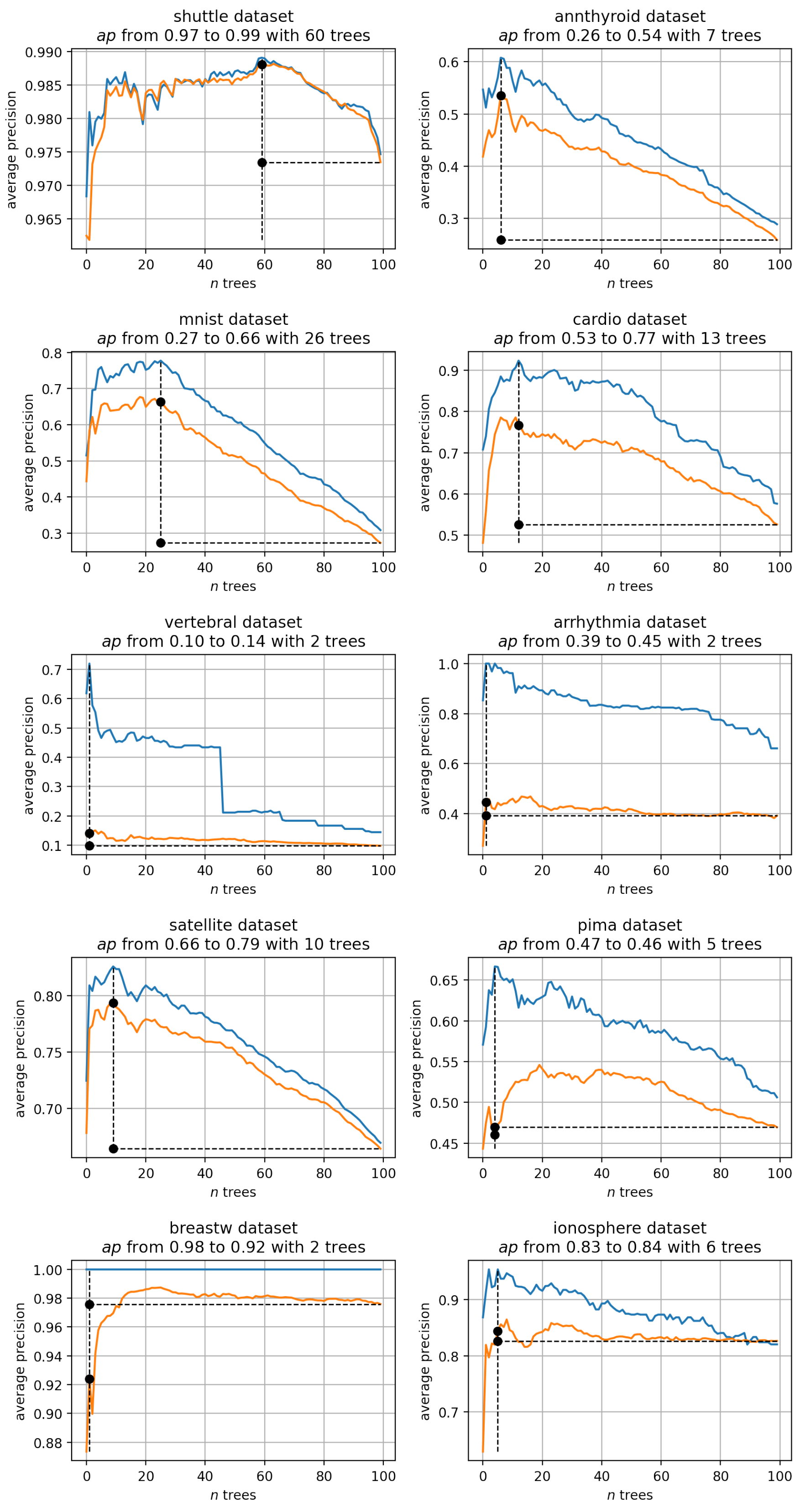


Figure 37: Application of the proposed algorithm to the real word datasets with the highest contamination, with supervision applied to 20% of the training dataset. The blue curve shows the performance of the algorithm on the supervised part of the training dataset, while the orange are obtained on the test dataset (50% of the full dataset). The two black dots highlight: the average precision obtained on the test set obtained with the forest that maximises the average precision during training, the number of trees needed to get these performances, and the average precision of the *standard* algorithm during test.

**Algorithm 5:** TiWS-IF training

```
Data: $\text{IF}_{n_T} = \{T_t\}$ with $t = 1, \dots, n_T$ , $D^s = (X^s, y^s)$
Result: TiWS-IF = $\{T_t\}, t \in \{t = 1, \dots, n_T\}$
$s^T$ and $s^F$ initialization;
for $t = 1, \dots, n_T$ do
    $s^T_t \leftarrow \overline{p}(T_t(X^s), y^s)$;
$a \leftarrow$argsort$(s^T)$
for $f = 1, \dots, n_T$ do
    $\text{IF}_f \leftarrow \{T_k\},\ k = a_1, \dots, a_f$ ;
    $s^F_f \leftarrow \overline{p}(\text{IF}_f(X^s), y^s)$;
$b \leftarrow$argmax$(s^F)$
TiWS-IF $\leftarrow \text{IF}_f$ with $f = a_1, \dots, a_b$
```

$D^s = (X^s, y^s)$. Note that $T_t(X)$ (and $\text{IF}_t(X)$) returns the anomaly score given by $T_t$ ($\text{IF}_t$) associated to every sample in $X$. Using $D^s$ each tree is measured according to the average precision metric $\overline{p}$ and the sorting order $a$ is computed. Given this ordering, $n_T$ new forests are created in an incremental way, adding one tree at a time: the first contains only the best tree, the second contains the two bests and so on until the $n_T$-th forest that contains the original forest. At this point each forest is again tested using the available dataset, and the best performing forest is saved and becomes the TiWS-IF. If more than two forests reach the highest precision, the forest with less trees is discarded in favour of the largest; this choice to select the most robust forest among the best ones, mitigating possible over-fitting behaviours.

Given $n_s$ the number of labeled points, the average precision scoring is $O(n_T n_s)$ while the sorting $O(n_T \log n_T)$. Excluding the anomaly score estimation $T_t(X^s)$ that is anyway performed by the IF, this means that for small number of labels the training time complexity is controlled by the number of trees (leading to $O(n_T \log n_T)$), otherwise is $O(n_T n_s)$. Moreover by selecting a smaller amount of trees the new forest reduce its testing time complexity to $O(nr \log \psi)$ and the memory requirements to $O(r\psi)$ where $r \leq n_T$.

Intuitively a cross validation phase might be added to the previous algorithm to achieve a more robust ranking of the best trees. Unfortunately not only increases the computational cost that needs to be maintained as low as possible, but from numerical experiments it also seems not to bring the expected benefit. This might be due to the strong unbalancement between labelled anomalies and labelled inliers, indeed doing a stratified validation shows the model the same anomalies many times, decreasing the effect of cross validation.

The proposed algorithm was tested over multiple datasets, splitting the full dataset into two equally large sets keeping constant the outlier contamination. Then, to simulate the role of the domain expert, a fraction of the training set with the same anomaly contamination was labelled and used in the supervised part of the algorithm. In all experiment, $n_T$ was set to be equal to 100, following the guideline provided by the original Isolation Forest paper and the common practice in the community.

An example of the results is visible in Figures 36 and 37 where the algorithm is tested on the same datasets shown in Figures 34 and 35, but with a supervised fraction of the training set of about 20%. As expected, the reported results resemble the ones shown in Figures 34 and 35, but they are different since now the training algorithm does not see the full dataset (and its anomalies) but a very small portion of those. In this picture the blue line shows the performances obtained from each forest in the supervised portion of the dataset, and the orange line represents the average precision that is measured using the same forests but over the testing set. The vertical line lies in correspondence with the maximal point reached by the blue curve, while the horizontal highlights the performances obtained by the standard forest on the test set. Larger is the interval between the two black dots, better is performing the proposed algorithm. The first aspect that needs to be discussed is the performances of the *last* forest, i.e. the last point of the curves in Figures 36 and 37, that depicts the average performances of the *standard* algorithm in the training (blue) and testing phase (orange); sometimes they perfectly overlap but unfortunately this not always happen due to the presence of different kind of anomalies inside the two datasets, since they are very few and different with respect to the whole dataset. This therefore justifies the different shapes of the training and testing curves that are very often, but not always, similar. The value of the proposed approach lies in the fact that choosing the best trees in the supervised set leads to maximise the chances to get a forest that outperforms the *standard* one even in the test set, and this is clearly visible in Figures 36 and 37. The only dataset where TiWS-IF fails is the *speech* dataset that has very high dimensionality (400 features) compared with the contamination (1.65%). On the contrary, the algorithm very often is able to choose a IF that reaches better performances with respect to the standard ones and even in many cases it selects a point that is close to optimal.

Obviously, the previously mentioned Figures 36 and 37 proved the TiWS-IF validity, but it needs to be repeated to get robust results and to better quantify the improvements lead by this choice of trees: since anomalies are few and different a particular split of the data may affect the results; to filter out the randomness introduced by this aspects, the algorithm is tested with 25 repetitions by shuffling the datasets before partitioning, with varying number of labels but by keeping constant the contamination: the results are reported in Figures 38-39 and 40-41. During the experiments, three values are collected: the baseline i.e. the value of the standard forest on the test set, the number of best trees learnt during training and the average precision measured in the test phase using this subset of trees. Moreover to assess the quality of the proposed solution, three popular approaches have been compared to TiWS-IF and to the standard Isolation Forest: namely the Local Outlier Factor (LOF) [36], the One-Class SVM (OC-SVM) [216] and the Histogram-based Outlier Score (HBOS) [100].

The detection results are depicted in Figures 38-39, while in Figures 40-41 the memory savings due to the forest reduction can be noticed by means of the tree cardinality of the selected forest. In Figures 38-39 the value of the standard forest on the test set is drawn using the blue color while the red represents the improvement due to the proposed strategy, the green is the OC-SVM, the purple LOF and in yellow the HBOS. From the Figures 38-39, Isolation Forest and TiWS-IF are, overall, the best approaches on average, however there is not an absolute winner (as typically happen when comparing several anomaly detection approaches over different problems), but it depends on the dataset structure: in some cases IF and TiWS-IF are the best methods, but in other cases OC-SVM or LOF performs better. The biggest *absolute* improvement of TiWS-IF over IF is measured on the *cover* dataset, where the average precision goes from about 0.1 to 0.8, probably due to the huge quantity of available data and repeatable anomalies. However, the second biggest *relative* improvement is on the *optdigits* dataset that does not have any special property with respect to the other datasets. In most of the cases TiWS-IF is able to improve over IF when has just one labelled anomaly on the supervised set and with more labels it shows a rapid improvement.

Looking at Figures 40-41 instead it is possible to see the number of trees that the forest needs to get the results shown in Figures 38-39. The model reduction is most of the times very large, indeed the bars seldom exceed 20-40, saving a lot of memory and computational power since, as previously explained, both memory requirements and time complexity scales linearly with the number of trees. It is interesting to note that as the fraction of labelled data increases, the algorithm is able to reduce more the size of the selected forest. Moreover the algorithm avoids to remove trees in cases like musk and shuttle where the IF performs already in a very good way.

One of the goal of TiWS-IF was to improve the average precision of the anomaly detector, providing a small supervision. Additional experiments were conducted to study the behaviour of other metrics describing the performance of the detector, like the ROC AUC score, and the maximum F1 score obtainable varying the classification threshold. In Figure 42 the improvements in these metrics before and after the TiWS-IF algorithm are depicted with different colors depending on the dataset. As expected, both of the pictures as positively correlated meaning that improving the average precision leads to improve the ROC AUC and the F1-score too. However the F1-score and the average precision are more correlated as they rely on the same basic quantities, i.e. precision and recall.

## 6.5 CONCLUSIONS

The detection of anomalies is a critical task in many real life scenarios, however the majority of algorithms are not designed to run on edge devices, learning from unsupervised data and getting the most out of few labelled data, when available. This Chapter tried to cope with these challenges, primarily observing that even one of the most popular and effective algorithm like the Isolation Forest can be improved considering this scenario. This is due to the creation of randomly grown isolation trees that, even tough they are the key aspect of the original algorithm being very cheap to train, some of them risk to damage the

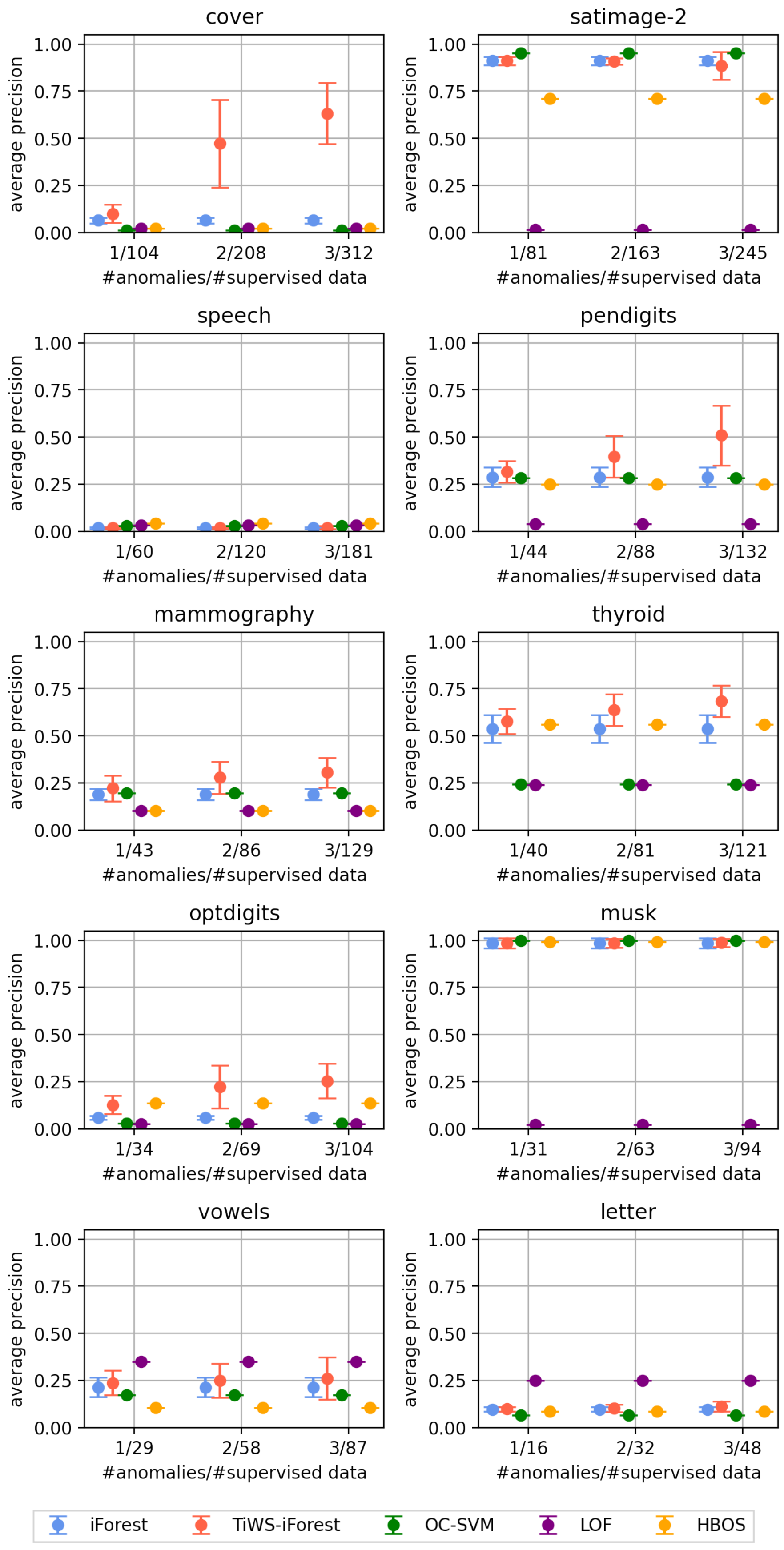


Figure 38: Detection performances: results obtained on 20 real word datasets; for each dataset, three different training sizes are analyzed, keeping the same dataset contamination but increasing the number of anomalies from 1 to 3; note that the number of labelled inliers changes accordingly. The algorithm seems to be robust even with very small and unbalanced training sets.

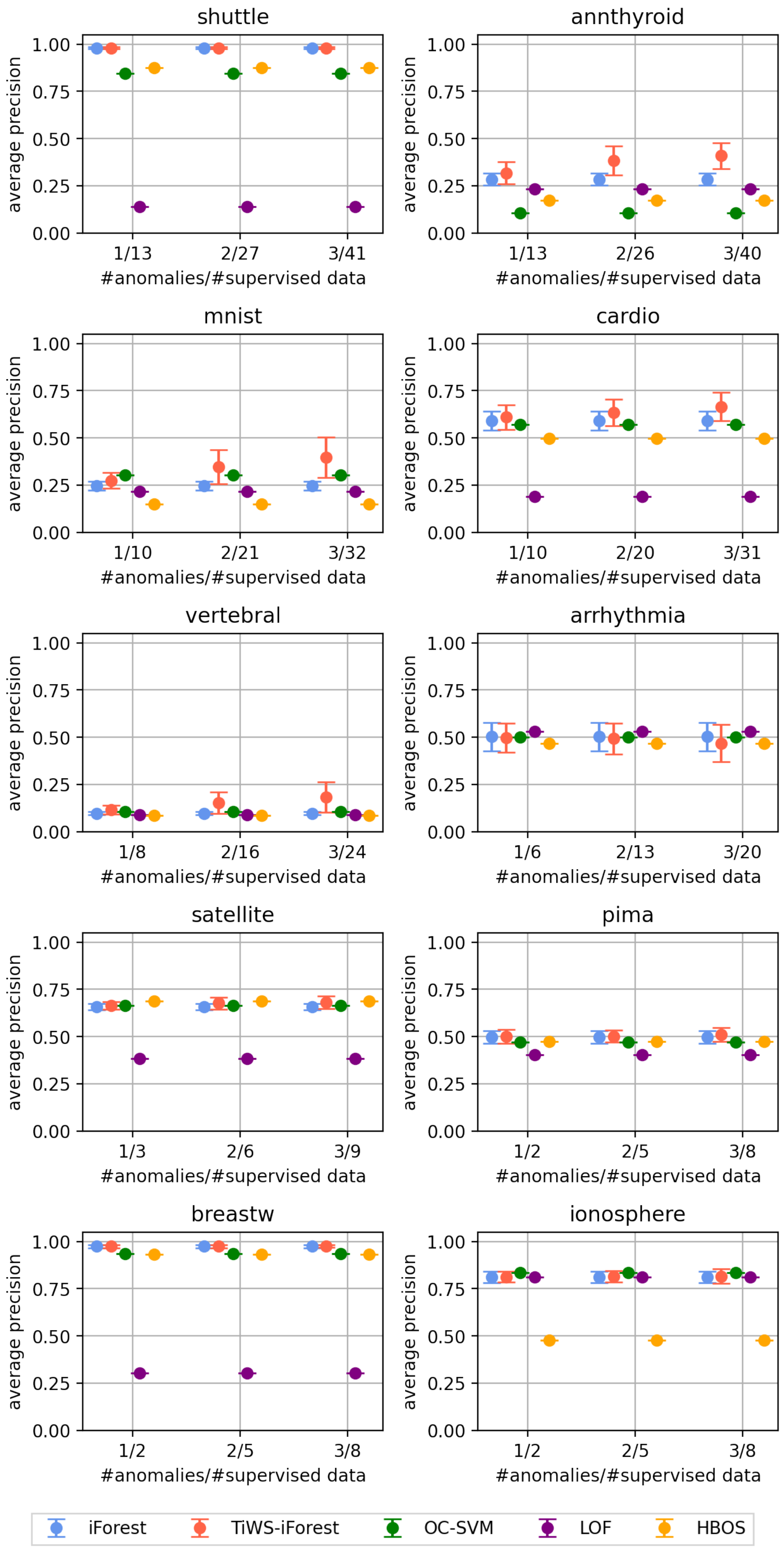


Figure 39: Detection performances: results obtained on 20 real word datasets; for each dataset, three different training sizes are analyzed, keeping the same dataset contamination but increasing the number of anomalies from 1 to 3; note that the number of labelled inliers changes accordingly. The algorithm seems to be robust even with very small and unbalanced training sets.

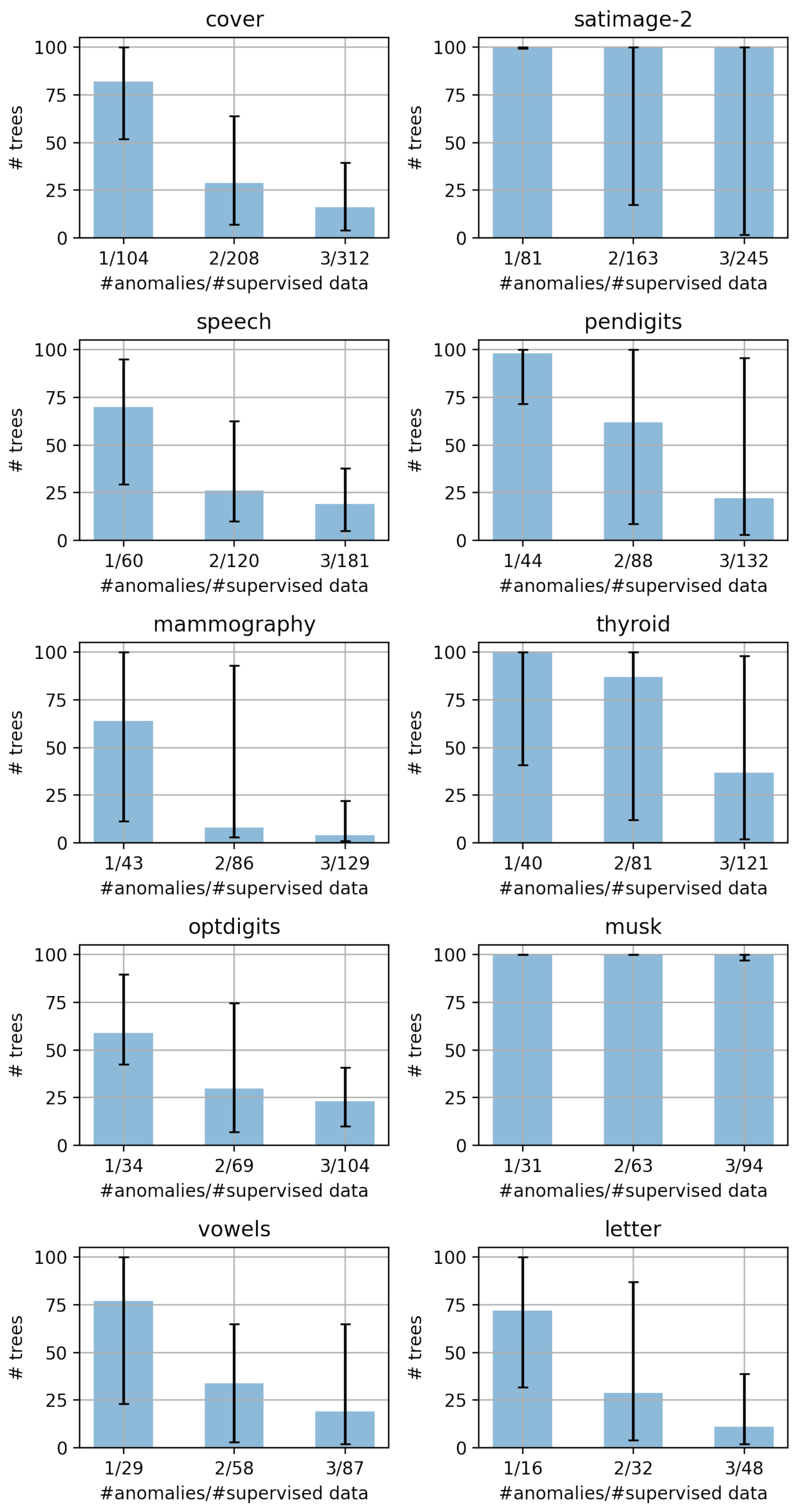


Figure 40: Reduction performances: results obtained on 20 real word datasets; for each dataset, three different training sizes are analyzed, keeping the same dataset contamination but increasing the number of anomalies from 1 to 3; note that the number of labelled inliers changes accordingly. The blue bars represent the median number of trees that the proposed strategy needs to reach its performances, while the black interval shows the 5th and 95th percentile. As opposed to the standard algorithms, TiWS-IF needs a fraction of memory and computations.

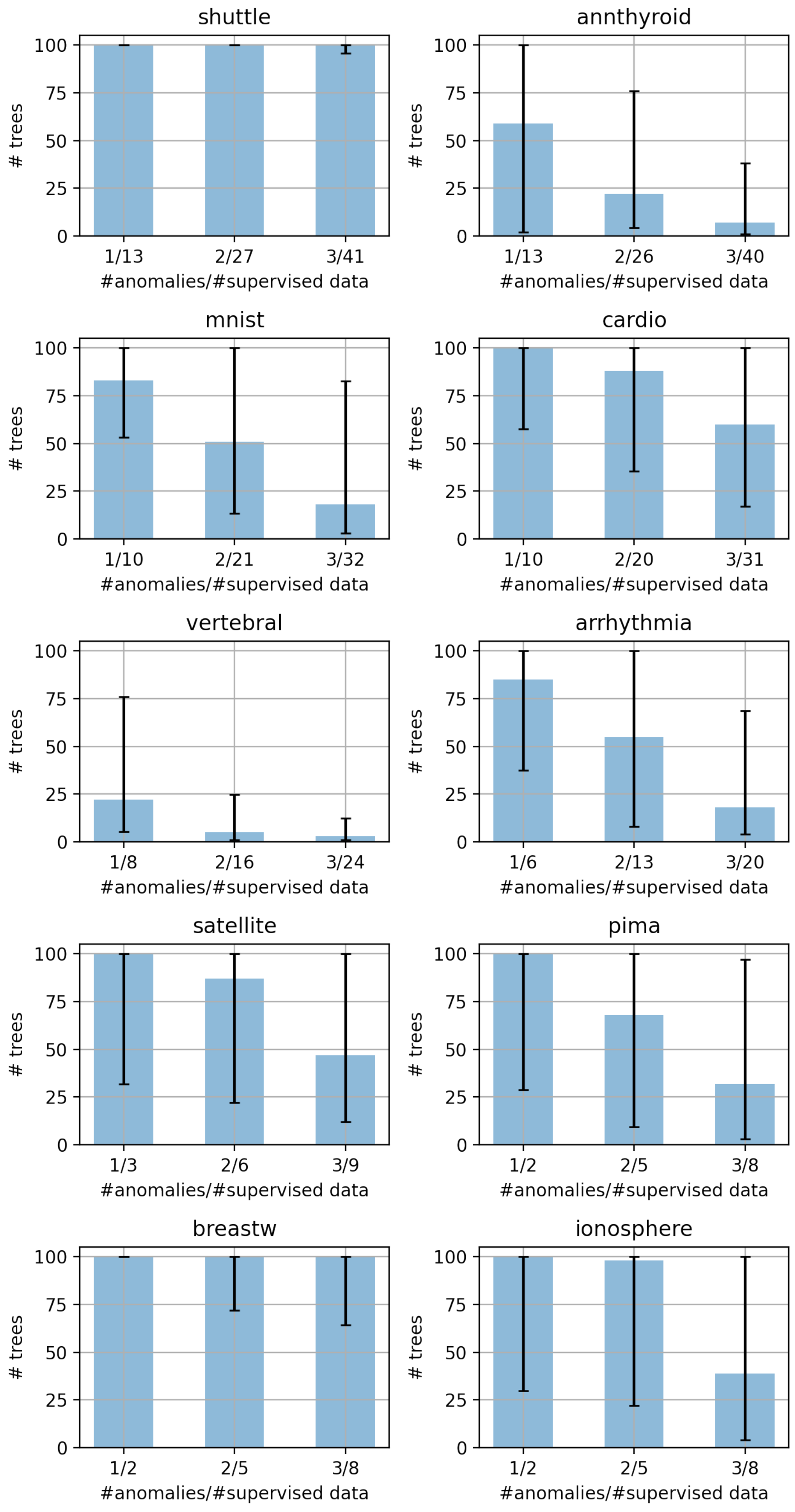


Figure 41: Reduction performances: results obtained on 20 real word datasets; for each dataset, three different training sizes are analyzed, keeping the same dataset contamination but increasing the number of anomalies from 1 to 3; note that the number of labelled inliers changes accordingly. The blue bars represent the median number of trees that the proposed strategy needs to reach its performances, while the black interval shows the 5th and 95th percentile. As opposed to the standard algorithms, TiWS-IF needs a fraction of memory and computations.

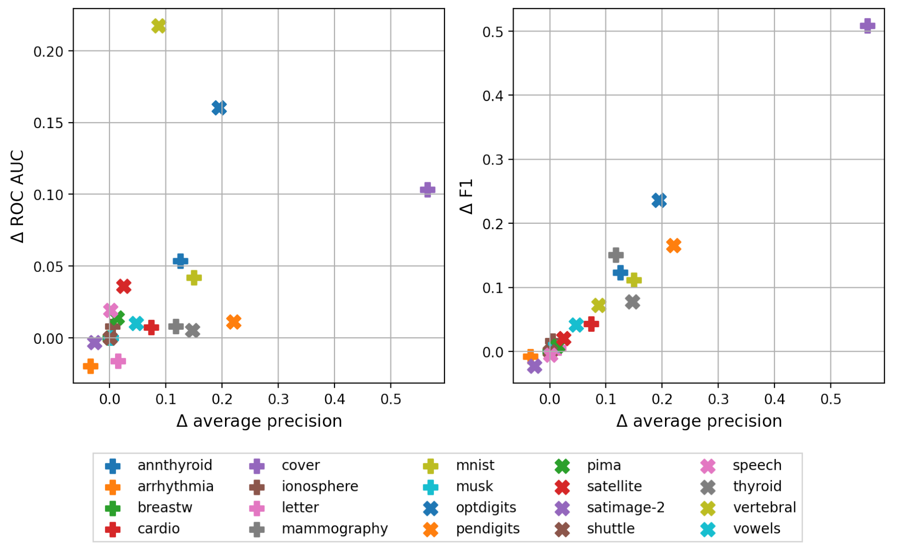


Figure 42: Performance improvement measured with different metrics: average precision, ROC AUC curve and F1-score. The improvements are valued as the difference between the score of the TiWS-IF and the corresponding IF.

solution accidentally. This work tries to solve this problem, designing a simple solution to remove the unnecessary trees, keeping only the most informative with the aid of few labels, named TiWS-IF. As shown in experiments on real world datasets, the forest highly benefits from this procedure and allows the practitioner to include some information in the unsupervised algorithm without a retraining procedure. Not only the detection performances increase, but also the memory and computational cost highly decreases, allowing the implementation of even more constrained devices.

A similar approach might be used vice-versa to generate the maximum number of isolation trees that fits into a given memory, that are the best with respect to the available supervised data, or it can be applied to other variants of Isolation Forest, like Extended Isolation Forest [107] and Isolation Mondrian Forests [172].

Another very promising approach, which we reserve as future work, is to exploit the insights described in [181], where the authors test alternative ways to compute the anomaly score. Indeed, in the cited article, the authors show that there may be other clever strategies to group the anomaly scores of the ensemble of trees to obtain the final anomaly score. This could give more robust results, inducing other strategies to select the best compressed forest.

As our work was inspired by DSS and dynamic settings, where data availability change over time, as future works, we will investigate the capability of handling the so-called *concept drift*, i.e. when the data distribution drifts and the new labels that are collected might refer to related but different concepts. Regarding this, we underline that the original Isolation Forest algorithm do not naturally cope with concept drift, making such research direction relevant not only in a TinyML scenario, but, more in general, in the application of Anomaly Detection solutions in real world scenario.

Furthermore, in a weakly supervised scenario, the risk of overfitting is real, as the number of labels used for training is very low and the number of labeled anomalies even more so. That said, experimental results show that the proposed approach rarely goes into overfit, either because the anomalies are somewhat similar to each other, or because the compression procedure saved the trees that generalize best. However, two strategies may be used to mitigate this possible effect: i) the simplest would be to choose more trees than the ones proposed by the best forest, where best is referred to the small set of available labels. This would give a less compressed forest, more similar to the original Isolation Forest. Another strategy that the authors reserve as a future work, might consist in using different labelled samples to rank the trees, and to rank the forests. This indeed might increase the generalisation capabilities of the algorithm.

# 7 ACTIVE ANOMALY DETECTION

In many fields, the application of ML techniques is severely limited by difficulties in the labelling process [96, 276]. Large scale unlabelled data sets might be available, but obtaining accurate labels for that data is costly, requiring expert intervention, or in some case absolutely impossible. Anomaly detection is one of such domains. Anomalies, by their nature are generally hard to identify, and therefore obtaining accurate labels is usually a difficult task. For this reason most approaches rely on unsupervised algorithms, where labels are not needed.

Active Learning (AL) is a subset of ML where models are assumed to be interacting with an oracle that can provide labels for a specific data point. There are many variants of AL setups, but in its simplest formulation, a model is given access to a relatively large unlabelled dataset, and it can subsequently query the oracle for the correct label of specific data points [219]. This might represent an user giving targeted feedback to the model, in order to improve performance with as little information as possible. The hope is that doing so, the model is able to direct the labelling efforts and minimize the costs associated with the expensive labelling process.

To convince the reader about the advantages of performing active learning strategies while learning a model here it is proposed a simple example of a binary classification problem on a set of linearly separable points uniformly distributed in the unit interval. We assume to have access to a large unlabelled dataset, but since the labelling "budget" is scarse, at most $k$ labels can be obtained. We know the data are linearly separable, so we want to identify the boundary between the two classes. We will try to do so by training a simple Support Vector Machine (SVM) on the labelled points. If we were to randomly label $k$ points from the data, the resulting model's decision boundary will be halfway between the most extreme cases we were able to acquired from both classes. If we assume the data to be linearly separable, the true decision boundary might be anywhere in the interval between the maximum of one class and the minimum of the other. We can analytically derive the expected size of this region by considering that the expectation of the minimum of $k$ i.i.d random variables $x_i \sim U(a, b)$ is:

$$E\left[\min\{x_1, x_2, \ldots, x_k\}\right] = \frac{b-a}{k+1} + a \ . \tag{6}$$

From this we can derive the expected distance of the closest example of each class to the real decision boundary to be proportional to $\frac{1}{k}$. This means that the model is quite slow in identifying the real boundary between the two classes, since the error shrinks linearly with k. The main reason why the error has a poor dependency on $k$ is that, if we label a new data point that is not in the uncertainty region, the new label is essentially useless to the model. As the error becomes smaller, so is the probability of randomly sampling an useful label. This consideration gives an intuition to why the AL approach can be useful. In fact, if we actively query points that are as closely as possible to the midpoint of the uncertain region, the maximum error is cut in half with new each label obtained. Therefore, the error under this approach shrinks proportionally to $2^{-k}$, which is an exponential decrease. This algorithm is essentially performing a binary search for the optimal solution. In [149] the authors show that the reasoning for this simple one dimensional case can be extended to binary classification with SVM in higher dimensions.

It is important to note that, in order to sample the optimal point, we need to evaluate each candidate separately. This might be very expensive to do, since it might require to evaluate the model's output for the whole dataset.
Also, identifying the model's "uncertain" region might require the solution of complex non linear optimization problems. In general, for most applications, rather than querying the "optimal" point with computationally expensive operations, one can resort to simple heuristic rules to simplify this step. In later Sections, we will rely on the second approach, since we are not particularly interested in exact bounds on the model's error, and good simple heuristic rules can significantly reduce the overhead introduced by the AL procedure.

The toy problem of binary classification is closely related to the AD task. In the AD case, the binary classification is usually very unbalanced. In practical application, things get more complicated, and we are unlikely to see the same exponential relations obtained for this simple case. However, most observations made remain very relevant, and will be crucial in understanding the motivations behind the model introduced in later Sections.

The popular definition of anomalies, i.e. *observations which deviate significantly from the majority of the data and do not conform to a well defined notion of normal behaviour* is too general and might not be applicable in every context. Anomalies are very domain dependent and samples considered anomalous in a context are normal in another and vice versa. Unfortunately to train a detector on a specific context requires labelled data that are often missing, therefore it is common practice to rely on unsupervised models: even if they are based on general notions of anomaly, they can return an anomaly score without the need of expensive labelled data. In this Chapter the goal is to develop a model that starting from an initial unsupervised solution, is able to tune it towards the user definition of anomaly, by interacting recursively with the user. This, following the Active Learning approach, allows to achieve the same or even better results w.r.t. a supervised model, with less labelled samples.

## 7.1 INTRODUCTION

Anomaly detection is commonly tackled in many industrial scenarios, such as credit card fraud detection [97], insurance fraud detection [81], insider trading detection [69], medical anomaly detection [255]. In these dynamic and often complex contexts, the problem of detecting anomalies is crucial in order to predict and avoid failures as well as to perform fault detection. In many industrial processes in fact, data-driven approaches for smart monitoring (for example predictive maintenance) have a key role, allowing to identify and isolate faults and to prevent future sudden failures. To solve this problem, anomaly detection represents an efficient solution. Generally, in this scenario a great amount of collected data are available but, since labelling is an expensive and time consuming process, there is a lack of ground truth labels, undoubtedly stating whether or not a point is anomalous. The learning problem therefore is unsupervised and the algorithm can just blindly look at the structure of the dataset, without a clear definition of what is an anomaly from the user perspective. Therefore unsupervised algorithms can only detect samples that exhibit some general property different to the rest of the dataset, for example some approaches look for points far from the majority, or detect points living in low density areas.

Unfortunately, anomalies are strongly domain specific [85]: as stated above, since no official definition is given, the concept of what an anomaly is entirely relies on the application in question. Specifically, it may happen that a set of data has different anomalies based on the given application domain and that the same data may be considered anomalous in one domain but normal in another [217]. For instance looking at data acquired by a measuring instrument, the manufacturer might define anomalies as events where the instrument has a faulty behaviour while the end-user might be more interested in events where the measured process behaves in a previously unseen way [27]. As a direct consequence, training a domain specific anomaly detector would require a full set of labeled data to capture the user definition of anomaly.

In real world applications, assigning labels to input data pose a considerable challenge to take into account [276]. In order to train reliable models, a large amount of labeled data is needed but, in practical scenarios, labeled examples are limited or often too expensive and time-consuming to collect, leading to a huge issue to face. Obtaining labels requires an often too expensive cost to take care of since the labeling procedure is usually carried on by a human domain expert who manually labels each point with a time-consuming and demanding routine. Moreover, by definition anomalous points are rare and difficult to spot, making the problem a difficult challenge to be solved.

Due to the difficulty of finding labeled points, in practical contexts anomaly detection is often treated as an unsupervised learning task. For the classical unsupervised anomaly detection problem, the purpose is to detect outliers with no use of labeled data based on the fact that normal data greatly outnumbers anomalous data, and anomalies are very different with respect to inliers. Unsupervised anomaly detection models are not tuned for the precise domain of application but are generally based on identifying rules based on specific data characteristics, such as density based algorithms, distance based methods etc. [36, 110, 112, 138]. However recent literature [61, 217] distinguishes the outliers to the anomalies: the first are the points highlighted by an unsupervised model, while the second are the ones the user actually sees as anomalous. As

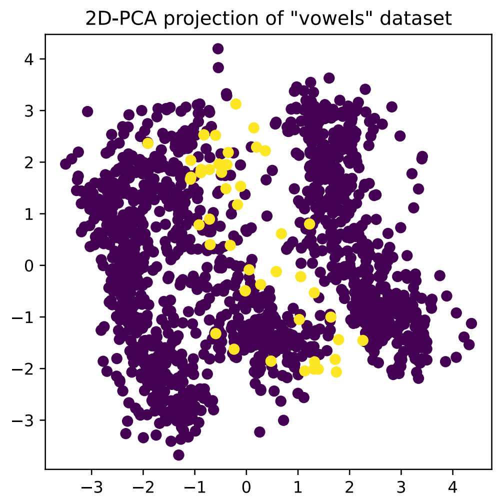


Figure 43: Projection of the *vowels* dataset on 2 dimensions using Principal Component Analysis. Purple data points are normal data, yellow points are anomalous data. Two aspects are clearly visible: i) it is impossible to separate anomalies from normal data with only two features and ii) anomalies tend to form a different class and might be quite different from general outliers. Not all the data points lying in low density areas or far from the majority are defined as anomalies, but just the ones lying in a specific part of the space. As a consequence, in the considered scenario, identifying anomalies might be a challenging task for an unsupervised detector.

the unsupervised model is not directly tuned to the detection of the anomalies, the outliers might weakly correlate with them.

Therefore, running unsupervised anomaly detection algorithms may be risky and often misleading: as stated above anomalies are strongly domain dependent and as a direct consequence, an unsupervised detector might not identify anomalous data which should be considered as such, as well as could wrongly detect as anomalies points which are normal based on the context taken under consideration [61]. Figure 43 presents a visual representation of the strong connection between anomalous data points and context domain. Specifically, the plot shows the two-dimensional projection of the *vowels* dataset [204]. As it can be seen, anomalies are not defined just as data points lying far or in low-density regions, but they form a class with a specific pattern defined by domain-experts, making complicated for the automatic detector to correctly identify them.

Among the unsupervised models, as stated in Chapter 4 a very popular anomaly detection algorithm is the Isolation Forest (IF) [160, 162], which presents a very different approach w.r.t. the majority of models: instead of creating a profile for normal data, it explicitly tries to isolate anomalies. To do it, IF relies on two assumptions: anomalies are fewer in number and they have very different attributes compared to normal data.

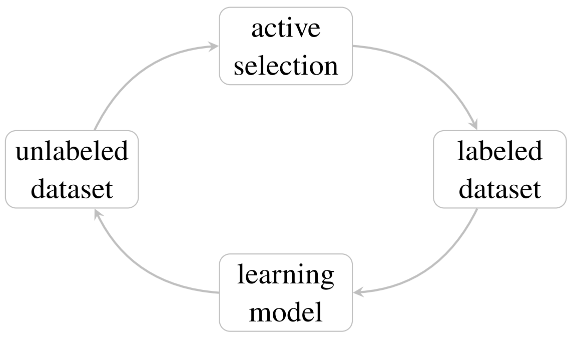


Figure 44: Active learning core structure. At each iteration a novel point is actively selected from the unlabeled set of data and the corresponding label is requested. Based on the received information, the model is modified.

In Decision Support Systems (DSS) [132], data streams are analysed in order to quickly extract strategic decisions on complex problems. Such process is monitored by users who frequently interact with the system and represent the actual decision maker of the whole process. In such framework, the proposed algorithm ALIF, represents an extremely appealing approach. Specifically, if a DSS is present, as a direct consequence, a user is already overseeing the process and inspecting data points: considering an unsupervised anomaly detection problem, inexpensive labels may be obtained in a fast way and using ALIF the model may be inexpensively updated.

In this Chapter we describe a procedure able to tune the detector model on domain specific anomalies by interacting with a human expert. To perform the proposed tuning method, not every training data are presented and labeled but a subset is automatically selected so that the number of interactions between the system and the human is minimised. The core idea is to ask labels corresponding to the most significant points to reduce the labeling cost and at the same time to maximize the detection performance. As a direct consequence, the proposed procedure may be regarded as an Active Learning (AL) based model [144].

Indeed AL represents a training approach particularly suitable when labeled samples are too expensive or difficult to obtain. Specifically, AL is a particular ML algorithm based on a key idea: despite the shortage of labeled data, high accuracy results may be obtained if the training algorithm is allowed to choose the points to be labeled and learn from them [218]. An AL algorithm asks an oracle to label the data considered most informative with an iterative approach. Doing so, since the queried points are directly selected by the learning algorithm, the amount of necessary labeled data is much smaller than that required for classical supervised ML approach. Figure 44 shows the core structure of any AL algorithm: at each iteration the model is updated using the labelled dataset, and is allowed to ask for a new label in the unlabelled dataset. This process repeats until the model reaches sufficient performances or when the number of iterations reaches the maximum budget.

This Chapter focuses on the Isolation Forest detector, and suggests a strategy to tune it towards the user definition of anomaly. In this work the authors compare two AL query policies to ask the user new labels, and other two policies to update the internal structure with minimal computational effort. The goal is to increase the performance of the detector as much as possible, keeping very low both the labelling effort and the updating procedure. Moreover this method has two key advantages over the supervised and computationally expensive models: as it relies on an initial unsupervised training, it can start to work when there are no labels, but more importantly it can work even if instances from only one class are labelled. This is particularly useful when obtaining labels from the anomalous class is very uncommon or expensive.

The rest of the Chapter is organized as follows. Initially, in Section 7.1 we outline the essential details about the Isolation Forest in detail useful to understand the proposed work, and we indicate an existing active learning-based anomaly detection algorithm that will be used as a benchmark in this work. Then, in Section 7.2 we illustrate the proposed model ALIF: namely, we describe the strategies suggested to query the points as well as the approaches employed to update the model. In Section 7.3 we test ALIF, comparing it with other models in relation to multiple real set of data. Finally, in Section 7.4 we draw conclusions for the present work.

RELATED WORKS Similarly to Random Forest [111], Isolation Forest training phase constructs an ensemble of decision trees, also known as isolation trees (iTrees), relying on the fact that anomalies are easier to be isolated, i.e., partitioned from the rest of the data, due to their distinctive features. First, Isolation Forest randomly sub-samples the dataset so that each iTree is obtained with a different set of data. Then, the structure of an iTree is generated in a completely random way: each partition is produced with a random selection of an attribute value from the subset at disposal and by the choice of the split value, selected randomly in the range of the picked attribute. This recursive procedure is repeated until all points are isolated or a predefined limit height is reached. After that, a novel random sub-sample is selected and the isolation procedure is repeated, in order to create a new random iTree. Once the training phase is completed and every iTree is fully grown, data traverse the different iTrees and its depths, i.e., the number of edges traversed from the root node to the external node containing it, are collected. Based on these depths $h^u$, also known as path lengths, the anomaly score is computed, i.e., an indicator of the likelihood that a point is an anomaly. Differently to Chapter 4, in the present Chapter we will make use of the superscript $u$ to denote the depth $h$ obtained by the model trained in the traditional unsupervised way.

Let $x \in X$, then the corresponding anomaly score is defined as

$$s(x) = 2^{-\frac{E(h^u(x))}{c(\psi)}} \tag{7}$$

where $E(h^u(x))$ is the average path length of $x$ with respect to all trees and $c(\psi)$ is a normalization factor with the sub-sample set size as input.

Recall that, $c(\psi)$ defines the average path length of an unsuccessful search in a binary search tree computed with a set of cardinality $\psi$. Based on Eq. (7): when $E(h^u) \rightarrow \psi - 1$, $s(x) \rightarrow 0$ and it is quite safe to consider $x$ a normal point; on the contrary when $E(h^u) \rightarrow 0$, $s(x) \rightarrow 1$ and $x$ is most likely an

| **Symbol** | **Description** |
|---|---|
| $X$ | generic sample set |
| $X^s$ | set of labelled training points |
| $X^u$ | set of unlabelled training points |
| $n_X$ | number of observations in $X$ |
| $n_s$ | number of observations in $X^s$ |
| $n_u$ | number of observations in $X^u$ |
| $x_j$ | $j-$th queried point |
| $F$ | forest |
| $T_t$ | tree |
| $n_T$ | number of trees |
| $L$ | leaf |
| $n_L$ | number of leaves |
| $P$ | partition of $X$ made by $L$ |
| $h_L$ | depth of $L$ |
| $I$ | number of inlier points in $L$ |
| $O$ | amount of anomalies contained in $L$ |
| $h^u(x)$ | "unsupervised" path length of generic point $x$ |
| $h^s(x)$ | "supervised" path length of generic point $x$ |
| $k(L)$ | color of leaf $L$ |
| $\lambda_T(x)$ | compute leaf containing $x$ with respect to tree $T$ |
| $H_{jt}$ | compute the path length of $x_j$ with respect to tree $T_t$ |
| $H \in \mathbb{R}^{n_T \times n_u}$ | matrix of elements $H_{jt}$ |

Table 11: List of symbols used.

anomaly; when $E(h^u) \rightarrow c(\psi)$, namely when the average path length of $x$ is close to the normalization factor, then $s(x) \approx 0.5$ and the sample has no recognizable anomalous factor. Note that, in order to classify whether or not a point is anomalous, it is necessary to define a score based border value, establishing a division between anomalous points and normal ones. However, the choice of such border value is strongly data dependent, relying upon its target subject, making it not a trivial task to be managed [113].

The proposed algorithm starts by growing a standard Isolation Forest. After that, an active learning based approach is carried on: in an iterative way, the model is allowed to ask for a point to be labeled, obtaining the true information about its nature as a direct consequence. Based on it, the inner structure of each tree is modified to exploit the incoming information and to improve/calibrate the model if necessary.

A similar active learning anomaly detection algorithm using an optimization based approach is presented in [61]. The paper describes an Active Anomaly Discovery (AAD) method where the points having the highest anomaly score are presented to an expert analyst in an iterative way, requesting the corresponding true label. Based on the information received, the algorithm tries to place the points labeled as anomalies as close as possible to the higher part of the model. Specifically, the model considers the structure defined by the Isolation Forest and associates to each leaf a weight value, starting from a uniform fixed value and, based on the feedback received, such weights are iteratively updated. Such updating approach is performed with the use of an iterative optimization problem, where the optimal weights are computed so that every labeled anomaly has a score which is higher with respect to the labeled normal points ones. Note that, in this approach the partitions computed by the Isolation Forest are not modified, only the corresponding weights are. The proposed method is applied into a tree-based detector [62], where the described updating approach is used into the Isolation Forest algorithm. Such algorithm is called IF-AAD.

Here we present a novel approach, called Active Learning-based Isolation Forest (ALIF), which differs from the aforementioned works because of its easy yet efficient formulation: independently from the size of the input set, the algorithm will execute in constant time, modifying the average path length of the input points but leaving unchanged the Isolation Forest partitions, with the perk of using both current and past information. Differently from IF-AAD and Random Forest, the proposed approach does not need to pass through an optimization procedure, leading to a much lighter update. In Section 7.3 we report the performance benchmarking of our approach, IF-AAD and Random Forests.

## 7.2 PROPOSED MODEL

In this work we present an adaptive anomaly detection model for fixing leaf depths according to labels received from domain experts. The core idea of the proposed approach is presented in Algorithm 6 and relies on developing an

Isolation Forest based model in which the detector has the possibility to query domain experts for labels. In this iterative environment, once the Isolation Forest is fully grown, the algorithm can choose the points to be labeled and, based on the novel information achieved, its core structure is modified, in order to obtain a strong increase in the performance, with the use of only a limited number of labeled points. For every iteration, such modification only takes place in the external node containing the queried point, maintaining the main structure of the Isolation Forest untouched. In this way, the main goal is, given the novel information, to update the Isolation Forest structure, readjusting it based on the labeled points.

```
Algorithm 6: ALIF($((x^s, y^s), X^u, F)$)
  Data: labeled point $(x^s, y^s)$, unlabeled dataset $X^u$, forest $F$
  Result: updated forest $F$, query point $x^s$
  $F \leftarrow LeafDepthUpdate((x^s, y^s), F)$;
  $H \leftarrow GetDepthMatrix(X^u, F)$;
  $x^s \leftarrow GetQuery(H)$
```

```
Algorithm 7: LeafDepthUpdate($((x^s, y^s), F)$)
  Data: labeled point $(x^s, y^s)$, unlabeled dataset $X^u$, forest $F$
  Result: updated leaf $L$
  for $T_t$ in $F$ do
      $L \leftarrow \lambda_{T_t}(x^s)$ ;
      if $y^s$ == anomaly then
          $O \leftarrow O + 1$ ;
      else
          $I \leftarrow I + 1$ ;
      $h_L \leftarrow h^s(k(L))$
```

```
Algorithm 8: GetDepthMatrix($(X^u, F)$)
  Data: unlabeled dataset $X^u$, forest $F$
  Result: updated leaf $L$
  for $T_t$ in $F$ do
      for $x^u_j$ in $X^u$ do
          $L \leftarrow \lambda_{T_t}(x^u_j)$ ;
          $H_{jt} \leftarrow h^s(k(L))$
```

Specifically, the proposed approach may be outlined as follows. Let $X = \{x^1, \ldots, x^n\}$ be a generic training dataset, where $x^i \in \mathbb{R}^m$, $i = 1, \ldots, n$. First, the Isolation Forest algorithm is trained, leading to a forest $F = \{T_t\}_{t=1}^{n_T}$ of fixed number $n_T$ of fully grown iTrees. By construction, $T = \{L_l\}_{l=1}^{n_L}$ namely each tree $T$ is characterized by a variable number of leaves $L$. We use the following form to describe each leaf $L$: $L = (P, h_L, I, O)$ where

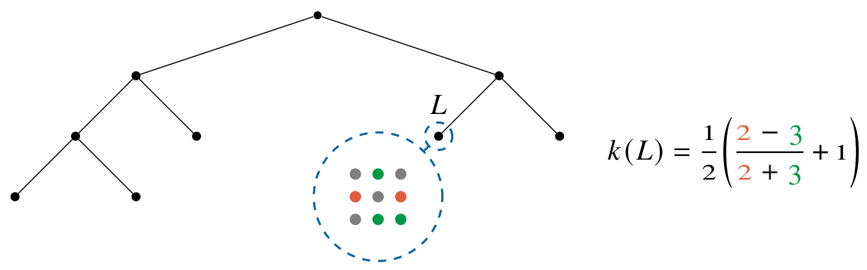


Figure 45: Every time a novel point is queried and the corresponding label is obtained, every iTree is investigated and the leaves containing the queried point are considered. A visual representation of the color of a generic leaf $L$ is displayed: green dots represents normal labeled instances; labeled anomalies are depicted by red dots. The corresponding color $k(L)$ is computed using Equation (8).

- $P$ defines the partition of $\mathrm{R}^d$ made by $L$, describing where a leaf is located with respect to the input space;
- $h_L$ is the depth of $L$;
- $I$ refers to the number of normal points in $L$;
- $O$ specifies the amount of anomalies contained in $L$.

As a direct consequence, initially each data $x \in X$ is assigned with the corresponding average path length $E(h^u(x))$ and anomaly score $s(x)$ as computed by the Isolation Forest.

From this step, the proposed iterative active learning approach can start. At each iteration, a point $x^s$ is selected and the corresponding true label $y^s$ is requested. Accordingly, each iTree $T$ is investigated, determining the leaf where the queried point lies, and modified as a result. We define $X^u = \{x^u\}$, $X^s = \{x^s\}$ and $Y^s = \{y^s\}$ respectively the set of unlabeled training points, the set of labeled inputs and the set of corresponding labels. Note that, by definition we have that $X = X^u \cup X^s$.

The key intuition behind the modification of the structure of each iTree is very simple. First, a queried point is selected and the corresponding trusted label is obtained. Secondly, for each $T$ of the model, the external node containing the point is analysed: its depth is updated based on the achieved information so that true anomalies are located closer to the root node while true normal points are far away from it.

Based on this scenario, it is important to define two essential yet independent tasks on which the entire algorithm relies on:

i. *Update strategy:* The actual approach employed to modify the classical Isolation Forest model;

ii. *Query strategy:* The plan of action to choose the optimal method to select the queried points.

Both tasks are detailed in the following.

UPDATE STRATEGY Let $L$ be the leaf under investigation. Then, based on the labeled points contained in it, i.e., based on the proportion of anomalies and normal points contained in $L$, we define the *color* of $L$

$$k(L) = \frac{O - I}{O + I}, \tag{8}$$

where $I$ and $O$ are respectively the number of labeled normal points and the number of labeled anomalies sampled in the leaf. Therefore, the color of a leaf defines its inner structure, describing how the total amount of labeled data contained in it is distributed. Specifically, it outlines the probability of a leaf to be anomalous with respect to the labeled data in it. Based on the color of a leaf, the corresponding fixing procedure is performed. Figure 45 shows how the *color* of a leaf is computed in a visual way.

Every time a novel point is queried, the model is updated based on the received information. In relation to each iTree, the update exclusively takes place with respect to the leaf containing the queried point. Specifically, let us consider a generic iTree $T$: once the queried point $x^s$ is selected and the true label is obtained, $T$ is investigated and the leaf containing $x^s$ is considered. Its depth $h^u(x)$ (computed by the Isolation Forest) is forgotten and substituted with a supervised value $h^s(x)$, entirely depending on the supervised data contained in the leaf, i.e., on $I$ and $O$. Algorithm 7 summarizes the above described procedure.

Equations (8) is used to obtain a leaf coefficient value $\overline{k}(L)$, describing the structure of the leaf with respect to the current labeled point as well as taking into account the past information received. Specifically, based on the information in use, the corresponding leaf value becomes

$$\overline{k}(L) = \frac{1}{2}\big(k + 1\big). \tag{9}$$

For consistency with the rest of the manuscript we renamed the function in (9) as $k$. Note that, using Equation (9), the codomain of function $k(L)$ is given by the closed interval $[0,\ 1]$. Specifically, the following evaluations are made:

- If $L$ contains only normal points and it is fully labeled, then $k(L) \longrightarrow 0$;
- If $L$ contains only anomalies and it is fully labeled, then $k(L) \longrightarrow 1$;
- If $L$ has a balanced number of both anomalies and normal points then $k(L) = \frac{1}{2}$.

Obviously $k(L)$ is computed only for leaves containing some labelled data while the algorithm keeps untouched the leaves that are not actively sampled.

The novel supervised depth $h^s(k)$ takes Equation (9) as argument. Specifically, we define two different approaches to compute $h^s(k)$ defined as:

1. *Piece-wise Linear* supervised depth

$$h^s(k) = \begin{cases} 2k\big[c(\psi) - h_{max}\big] + h_{max}, & \text{if } 0 \le k < \frac{1}{2} \\ \\ 2k\big[h_{min} - c(\psi)\big] + 2c(\psi) - h_{min}, & \text{if } \frac{1}{2} \le k \le 1 \end{cases} \tag{10}$$

where $h_{max}$ and $h_{min}$ are respectively the minimum depth and the maximum depth of the Isolation Forest computed during the unsupervised training.

2. *Logarithmic* supervised depth

$$h^s(k) = -c(\psi)\log_2(k). \tag{11}$$

Note that, in Equations (10)

- when $k(L) \longrightarrow 0$, then $h^s(k) \longrightarrow h_{max}$;
- when $k(L) \longrightarrow 1$, then $h^s(k) \longrightarrow h_{min}$;
- when $k(L) = \frac{1}{2}$, then $h^s(k) = c(\psi)$.

Regarding the logarithmic depth described in Equation (11), when $k(L) = \frac{1}{2}$, then $h^s(k) = c(\psi)$. Nevertheless, when $k(L) \longrightarrow 1$, then $h^s(k)$ converges to 0 and, analogously, $h^s(k) \longrightarrow +\infty$ when $k(L) \longrightarrow 0$. Unfortunately, this makes the logarithmic choice quite unstable when normal points are labelled. However, it is possible to apply a threshold on the logarithm to saturate it, leading to a behaviour very similar to the piece-wise linear function. Figure 46 plots the relation connecting the leaf value $k(L)$ with the supervised depth $h^s(k)$ for both the piece-wise linear depth and the logarithmic depth.

It is important to note that, the proposed update strategy does not require to fully retrain the forest but, on the contrary, when novel information is acquired, ALIF only modifies the actively sampled leaves, leaving the rest unchanged. Therefore, the time complexity of the update strategy is linear with respect to the number $n_T$ of trees in the Forest.

QUERY STRATEGY The idea of incorporating expert feedback in unsupervised anomaly detection algorithms aims at improving the achieved performance adding a relatively small computational and labelling cost. To improve the model in an optimal manner, the choice of the proper strategy to select the points to be queried must be established. Beyond the employed strategy to modify the structure of the Isolation Forest based on the novel information in fact, the proposed model is significantly based on the choice of the queried points. Specifically, for the successful outcome of the method, choosing for the most appropriate point to be labeled is a key factor which could compromise its outcome. In classical active learning scenarios, several possible query strategies are presented [218].

For any iTree $T_t \in F$ and input point $x_j \in X^u$, let $\lambda$ be defined as

$$\lambda_{T_t}(x) = L, \tag{12}$$

namely, function $\lambda$ assigns at each input point $x_j$ the leaf containing it with respect to tree $T_t$. Now, using Equation (12), we define

$$H_{jt} := h^s(k(\lambda_{T_t}(x_j))) = h^s(k(L)).$$

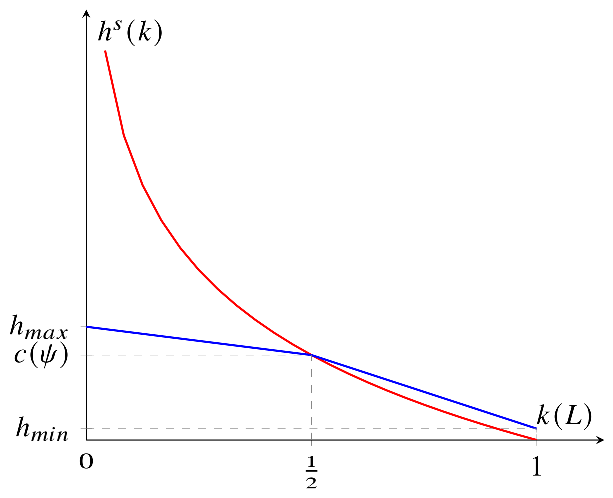


Figure 46: Visual representation of the two possible definitions of supervised depth. The blue function represents the piece-wise linear depth: when the leaf value $k(L)$ is close to 0, i.e., the leaf is fully labeled with normal points, the novel supervised depth of $L$ is tends to the maximum depth; if $k(L)$ takes a value close to 1, the leaf will be push to the minimum depth, as this situation corresponds to $L$ fully labeled of anomalies. The red function shows the logarithmic depth: in the range extremes 0 and 1, a threshold value is applied and the depth values are forcibly set to respectively $h_{max}$ and $h_{min}$.

Then, let $H \in \mathbb{R}^{n_T \times n_u}$ be the following matrix

$$H = \begin{bmatrix} H_{11} & \cdots & H_{1n_u} \\ \vdots & \ddots & \vdots \\ H_{n_T1} & \cdots & H_{n_Tn_u} \end{bmatrix}.$$

The choice of the point to request fully relies on matrix $H$. Details of the design of matrix $H$ can be found in Algorithm 8.

Based on $H$, we define the two following query strategies:

1. *Maximum uncertainty*: at each iteration the point selected to be labeled $x^s$ is the one where the iTrees disagree the most, namely

$$x^s = \underset{j=1,\dots,n_u}{\operatorname{argmax}} \underset{t=1,\dots,n_T}{\operatorname{std}} H \tag{13}$$

By doing so, the intended purpose would be to give the greater assistance to the model. Specifically, at each iteration we ask for the label of the data point where the model is more unsure, adding information with respect to the most uncertain data.

2. *Most anomalous*: at each iteration the point selected to be labeled $x^s$ is the point having the highest anomaly score value in the current iteration, namely

$$x^s = \underset{j=1,\dots,n_u}{\operatorname{argmin}} \underset{t=1,\dots,n_T}{\operatorname{mean}} H \tag{14}$$

This query strategy is the less expensive and more straightforward one and involves asking for the label of the point regarded as the most anomalous.

Note that, in order to make the proposed approach feasible, for each of the listed strategies, each point may only be queried once. When a point is queried and the corresponding label is obtained in fact, there is no need to ask for its label again since we are considering the information received undoubtedly true.

From a business process point of view, querying the most anomalous point may represent a sort of DSS, providing assistance to the decision making process and at the same time extracting information from a significant amount of data. As stated above, a DSS software gathers data considered the most informative and, based on the information achieved, it generates analysis tools useful for the decision making process.
In this scenario when a point is strongly considered an anomaly from the system, the domain expert is trigged for a checking. In this situation, the expert attention is drawn: the point has to be analysed in order to provide the actual corresponding label. Doing so, novel data information is obtained, combining the expert's knowledge together with the computerized system. In this way, the expert may validate the decision-making process and at the same time quickly extract useful information for the decision-making process.

From a model based viewpoint, asking for the point where the iTrees are more uncertain may be the most relevant strategy. With this approach in fact, the assistance provided by the expert aims at addressing the decisions where the model struggles the most and, doing so, at updating the most critical situations. Of course, given the unbalanced number of anomalies, it is highly likely that, using the maximum uncertainty strategy, the vast majority of the labeled points would be normal data, making it possibly less suitable in a more practical prospective.

As specified by Equations (13) and (14), ALIF query strategy corresponds to searching along the number of rows and columns of matrix $H$ and has time complexity $O(n_X n_T)$.

## 7.3 RESULTS

In this section, we analyse the performance of ALIF, matching both the proposed update strategies with the two described query approaches. Doing so, we hope to obtain a full and detailed evaluation of the presented model, with the purpose of analysing the efficiency of each combination as well as giving meaningful guidance on the proper use of the proposed strategies.

Firstly, we tested our approach on synthetic datasets like the challenging shape depicted in Figure 47, where the normal data make up a square toroid and the anomalous data lie inside it. In this kind of datasets, the Isolation Forest perform quite badly and there is a lot of room for improvement as it struggles to separate in few steps the anomalies: in this case it is much easier to wrongly separate normal points w.r.t. anomalies, indeed the first iteration

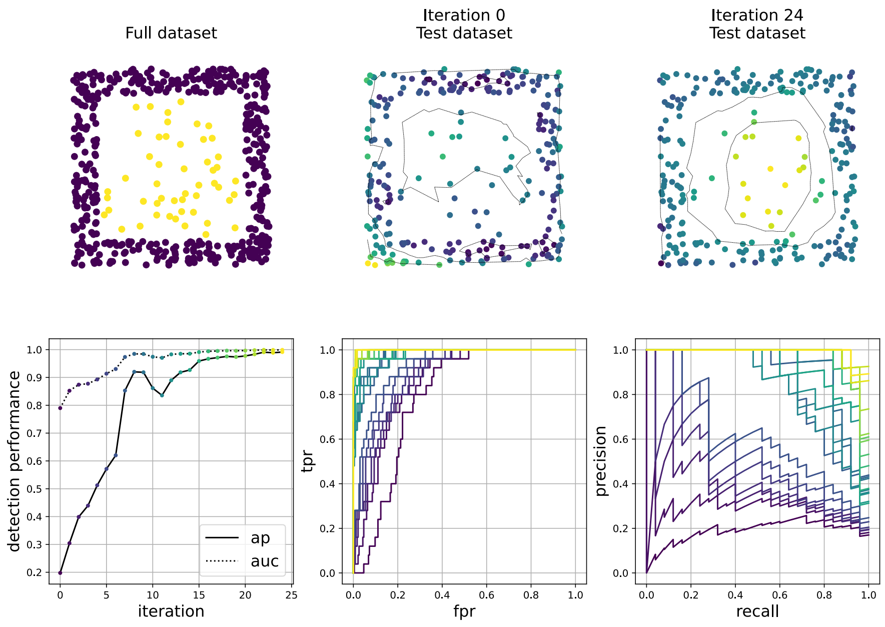


Figure 47: Test of the algorithm on a square toroidal dataset: anomalies lie inside of a box made up of normal instances. This setting is particularly challenging for the Isolation Forest algorithm as it is much easier to quickly separate normal points (depicted in purple) w.r.t anomalies (yellow). The anomaly score on test samples is shown on the second and third panel: the yellow is assigned to the points having the highest anomaly score, the purple viceversa. In the second row of panels the performances of the detector at each iteration is depicted: the *area under the ROC curve* (auc) and the *average precision* (ap) measured on the test set quickly improve.

of the model corresponding to the unsupervised training gives the highest anomaly score to the normal bottom left point of the toroid. On the contrary, at the 25-*th* iteration the model has learnt the correct function and is able to perfectly classify all the points. This is also visible in the panels of the second row of Figure 47, where the performance of the detector at each iteration is depicted with lines having different colors. As the model is allowed to query new points, the average detection performance quickly improves.

We decided to test the model on a set of 18 real data openly available [72, 204]. Table 12 presents the dataset used to test the performance of the proposed model. These come from different domains like medicine, industry and natural sciences and are characterised by various number of points as well as percentage of anomalous data. Defining the contamination as the ratio between the number of anomalies and total number of samples of the dataset, they are characterized by a wide feature range between 6 to 100 and a contamination percentage between 0.9% and 36%. It is very important to highlight how most of these datasets were built: they are adaptations of multi-class classification datasets to the anomaly detection task where one of the classes is under sampled and labelled as outlier. This means that points considered outliers are not just points randomly scattered in the features space like general anomalies, but they live in specific positions of the space that can be hard to be defined without labels. In this context, weakly supervised methods prove their efficacy, starting from an unsupervised guess, and improving continuously as labels are included in the model.

We used the average precision score metric to measure the results obtained. Splitting equally the dataset in training and testing set, we carried on a number of 25 queries, and the experiments were conducted 50 independent times to study the performances distribution. The experiments were performed on equipment with Intel Core i7-6800K CPU and 32 GB RAM.

First, we tested the four possible combinations of the two proposed update strategies together with the two query strategies so as to determine the best combination with respect to the majority of the considered test sets. Figure 48 shows the obtained results.

Given that the first point of the curve represents the performance of the fully unsupervised Isolation Forest, it can be observed as, broadly speaking, all four proposed matches represent an improvement with respect to the performance of the Isolation Forest. The two trials of the "most anomalous" query are depicted in solid line, while the "maximum uncertainty" in dashed line. Even if there are some exceptions, in most cases the first policy seems the one having the fastest improvements. This is a very interesting aspect since the "most anomalous" strategy is the cheapest and most natural policy among the two considering the DSS scenario previously described. Concerning the updating strategy, as expected, the piece-wise linear is the one having the most stable improvements in the dataset and among the datasets. As a direct consequence, we decided to use the combination "most anomalous - piecewise linear" for the comparison to the other baseline and state-of-art competitor.

| **Dataset** | **Instances** | **Features** | **Anomalies** | **Contamination** |
|---|---|---|---|---|
| AnnThyroid | 7200 | 6 | 534 | 7.42% |
| Breastw | 683 | 9 | 239 | 35 % |
| Cardio | 1831 | 21 | 176 | 9.6% |
| Cover | 286048 | 10 | 2747 | 0.9% |
| Ionosphere | 351 | 33 | 126 | 36% |
| Letter | 1600 | 32 | 100 | 6.25% |
| Mammography | 11183 | 6 | 260 | 2.32% |
| Mnist | 7603 | 100 | 700 | 9.2% |
| Optdigits | 5216 | 64 | 150 | 3% |
| Pendigits | 6870 | 16 | 156 | 2.27% |
| Pima | 768 | 8 | 268 | 35% |
| Satellite | 6435 | 36 | 2036 | 32% |
| Satimage-2 | 5803 | 36 | 71 | 1.2% |
| Thyroid | 3772 | 6 | 93 | 2.5% |
| Vertebral | 240 | 6 | 30 | 12.5% |
| Vowels | 1456 | 12 | 50 | 3.4% |
| WBC | 278 | 30 | 21 | 5.6% |
| Wine | 129 | 13 | 10 | 7.7% |

Table 12: Set of data used in the experimental phase. The first column gives the name of the dataset; the second column describes the number of instances contained in each set; the third column defines the total amount of features; the fourth column gives the number of outliers; the last column presents the contamination rates.

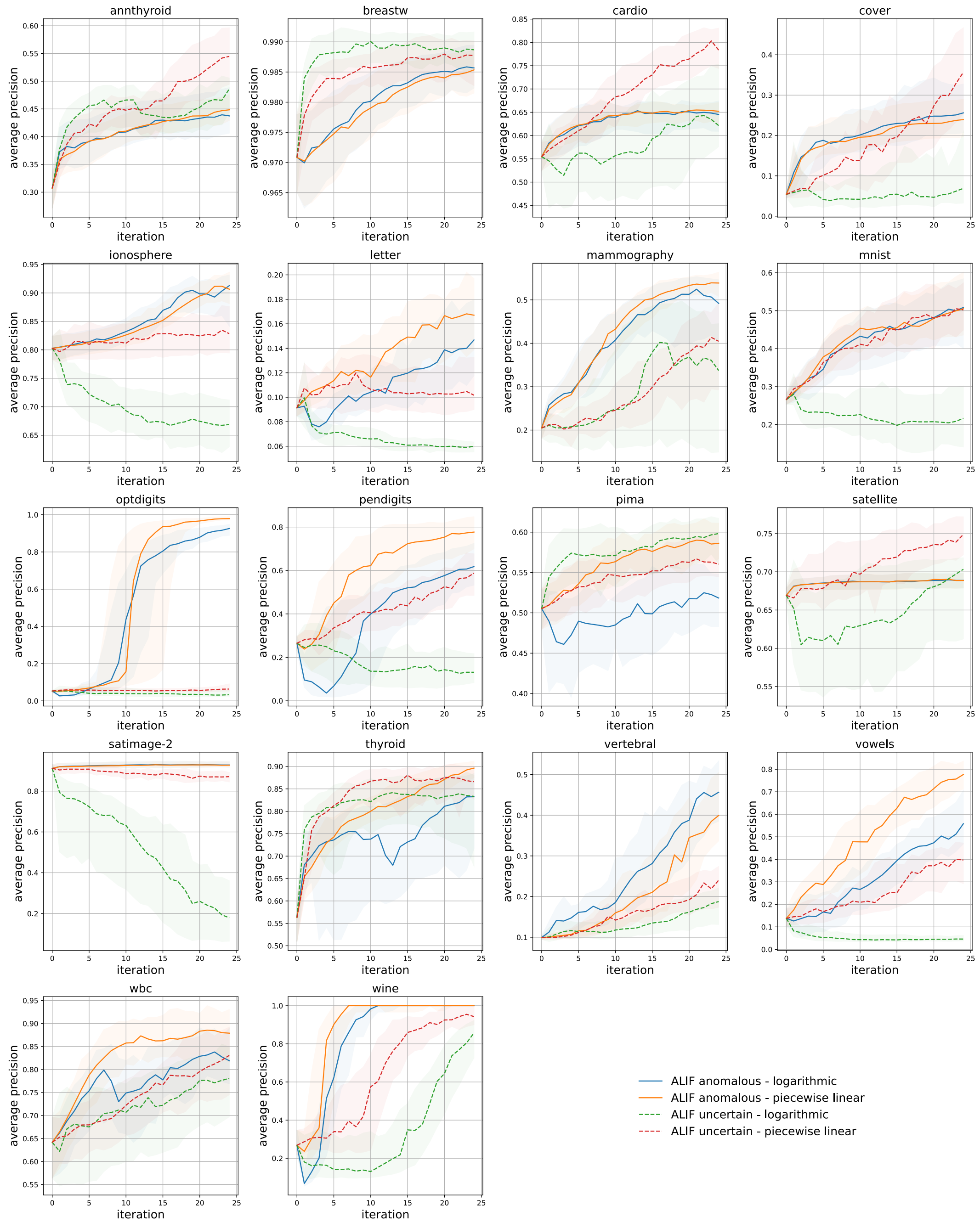


Figure 48: Performance of the four combinations of the proposed strategies with the datasets described in Table 12. As a general result, it can be noticed that querying the most anomalous point represents the most appropriate choice, overall leading to quicker improvements of the performance. When it comes to the update strategies, both the linear and the logarithmic depths seem to represent a reasonable choice. However, the linear depth appears to moderately be more stable.

As our approach ALIF essentially relies on designing an active learning based modification of the classical Isolation Forest, we decided to compare it with other tree based models: a fully supervised model, i.e. the Random Forest, and a weakly supervised model, the Isolation Forest - Active Anomaly Detection (IF-AAD). The RF represents the baseline since it is the most well known but simple classification approach; unfortunately it requires samples from both the classes inlier/outlier and it is computationally expensive as every time the user labels a new data point the forest is retrained. As the RF does not have a default method to query its points, in the following comparison the points given to the RF for training are the points queried by our ALIF. On the other side IF-AAD is a active semi-supervised model that does not need both class labels and is available online[1].

Figure 49 and Table 13 show the benchmark results: it is prominent that ALIF obtains the finest results with a very little number of labels, generally outperforming IF-AAD. Due to the strong imbalance between inliers and outliers RF receives quite late both labels from the two classes, leading to poor detection performances: in *breastw*, *satellite*, and *satimage-2* the querying process never got them over 25 iterations and 50 independent repetitions

## 7.4 CONCLUSIONS

In this Chapter a modification of the unsupervised Isolation Forest named ALIF has been suggested to improve the performance of the standard algorithm. Due to the lack of fully labelled datasets, Anomaly Detection is often performed by means of unsupervised models. However, as anomalies heavily depend on the context and are very domain specific, unsupervised models may be unable to detect them because of different definition of anomaly. To solve this problem we suggest a model that starting from an unsupervised model, iteratively tunes the model towards the user-definition of anomaly. This allows to enhance the detection performances, avoiding the need to fully label the training dataset and keeping as low as possible the number of required labels. Indeed this approach takes advantage of the Active Learning framework where the model is able to query the user and select the most interesting samples to label. ALIF relies on two important and inter-connected steps: the query policy that selects the point to be labelled, and the model update policy that allows the model to actually learn from the new query. From experiments performed on real datasets it turned out it is better to ask to label the most anomalous point, leading to a cheap and natural query strategy in practice. Concerning the update policy, the model does not need to fully retrain the forest, but needs just a simple update of the leaf depth, with a cost $O(n_T)$. Regarding the query strategy, the required cost is $O(n_X n_T)$. The cheap time complexity together with the fact that ALIF does not need labels of both anomalies and normal points provides a great advantage compared to other state-of-art algorithms. Comparing ALIF with other methods, it turns out it is also generally much faster in the learning of the correct anomaly definition, when the labelling effort needs to be kept low.

[1]https://github.com/shubhomoydas/ad_examples

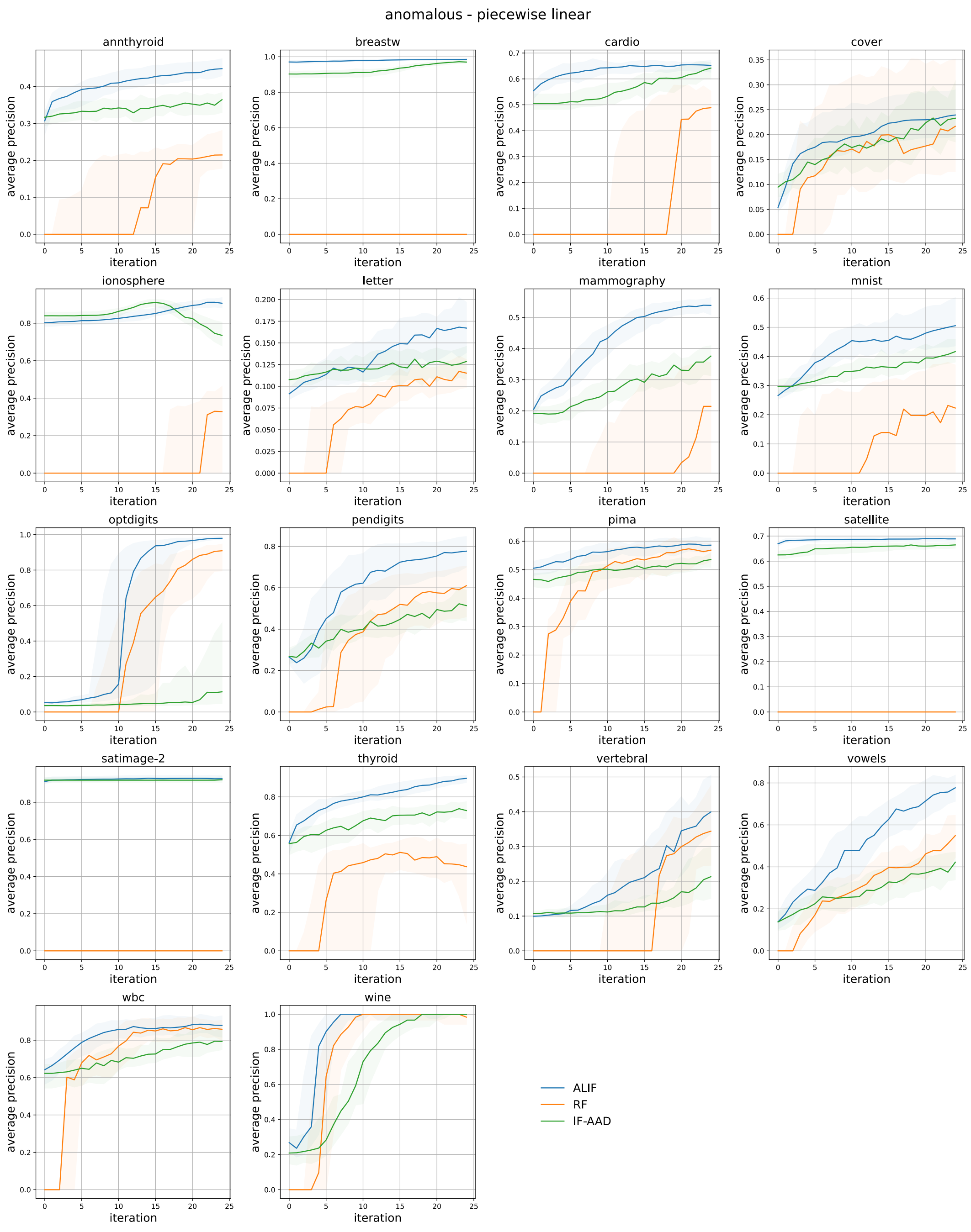


Figure 49: Comparison of most anomalous - piece-wise linear ALIF with IF-AAD and RF. It can be observed that, in general our method represents the best course of action, having the highest performance score usually with a very small amount of labeled data.

| | | anom-log | | anom-lin | | unc-log | | unc-lin | |
|---|---|---|---|---|---|---|---|---|---|
| | IF-AAD | RF | ALIF | RF | ALIF | RF | ALIF | RF | ALIF |
| **annthyroid** | 0.34 | 0.11 | 0.41 | 0.11 | 0.41 | 0.11 | 0.44 | 0.22 | **0.45** |
| **breastw** | 0.93 | 0.00 | 0.98 | 0.00 | 0.98 | 0.28 | **0.99** | 0.48 | 0.98 |
| **cardio** | 0.56 | 0.15 | 0.63 | 0.16 | 0.63 | 0.34 | 0.57 | 0.49 | **0.69** |
| **cover** | 0.18 | 0.20 | 0.21 | 0.18 | 0.20 | 0.25 | 0.09 | **0.46** | 0.19 |
| **ionosphere** | 0.84 | 0.08 | **0.85** | 0.08 | **0.85** | 0.47 | 0.70 | 0.52 | 0.82 |
| **letter** | 0.12 | 0.08 | 0.11 | 0.07 | **0.14** | 0.06 | 0.07 | 0.07 | 0.11 |
| **mammography** | 0.28 | 0.09 | 0.40 | 0.10 | **0.43** | 0.04 | 0.28 | 0.18 | 0.28 |
| **mnist** | 0.36 | 0.15 | 0.41 | 0.15 | **0.43** | 0.25 | 0.24 | 0.42 | 0.42 |
| **optdigits** | 0.11 | 0.46 | 0.49 | 0.38 | **0.51** | 0.00 | 0.05 | 0.00 | 0.06 |
| **pendigits** | 0.42 | 0.28 | 0.39 | 0.34 | **0.59** | 0.10 | 0.19 | 0.24 | 0.42 |
| **pima** | 0.50 | 0.44 | 0.49 | 0.43 | 0.56 | 0.34 | **0.57** | 0.43 | 0.54 |
| **satellite** | 0.65 | 0.01 | 0.69 | 0.01 | 0.68 | 0.42 | 0.64 | 0.49 | **0.70** |
| **satimage-2** | 0.92 | 0.00 | **0.93** | 0.00 | **0.93** | 0.25 | 0.48 | 0.53 | 0.88 |
| **thyroid** | 0.66 | 0.37 | 0.69 | 0.33 | **0.80** | 0.10 | 0.73 | 0.33 | **0.80** |
| **vertebral** | 0.14 | 0.16 | **0.27** | 0.12 | 0.22 | 0.17 | 0.14 | 0.22 | 0.17 |
| **vowels** | 0.29 | 0.29 | 0.33 | 0.30 | **0.51** | 0.07 | 0.06 | 0.18 | 0.27 |
| **wbc** | 0.71 | 0.68 | 0.76 | 0.68 | **0.82** | 0.11 | 0.69 | 0.21 | 0.73 |
| **wine** | 0.69 | 0.73 | 0.80 | 0.75 | **0.85** | 0.28 | 0.38 | 0.47 | 0.64 |

Table 13: Summary of the obtained results. The reported performances are the mean performance along the 25 iterations and 50 repetitions of the algorithm. ALIF consistently beats the other tested approached on all the datasets, in particular the combination of the most anomalous query policy and the piece-wise linear leaf update.

Future work will investigate other ways to construct the anomaly score from the available weak supervision. Indeed, these could exploit the insights discussed in [181], where the anomaly score is obtained from the weighted path of the point. In this way, not only the final leaf but also the internal nodes could play an important role. Moreover, hybrid query strategies that use a combination of the previously described most *anomalous* and *uncertain* strategy, might lead to even better results.

# 8 ACTIVE ANOMALY DETECTION: A BAYESIAN APPROACH

The previously described algorithm ALIF can be inserted in a more theoretically grounded scenario. The goal of this Chapter is indeed to adapt ALIF in order to fit into a Bayesian framework, showing the theoretical and practical benefits of such procedure. In fact, the problem can be seen as a Bayesian inference problem where the previously trained forest serves as a prior, and the posterior is updated as new points arrive. Moreover this Chapter also highlights the benefits of an Active Learning scenario in comparison to a standard Weakly-Supervised scenario where the model is not allowed to ask for labels but passively receives random data points.

## 8.1 INTRODUCTION

Anomaly Detection, also referred as Outlier Detection, is a field of Machine Learning that, as the name suggests, focuses on the identification of deviations from standard "normal" behaviour observed in the majority of the data [109]. There is no universally accepted definition of what constitutes an anomalous point and, as such, its definition may vary from task to task [85]. The motivations behind the desire for identification of anomalies might also vary: developers for example may have the need to remove outliers from the dataset to improve the robustness of some model or accuracy of statistical measures, or, as in more modern applications, anomalies might themselves be points of interest (i.e. fault detection); users, on the other hand, may be interested in AD tools to have enhanced monitoring capabilities of complex systems [19, 140, 272].

As in other ML fields, approaches for anomaly detection can be classified as supervised, semi-supervised and unsupervised[1]. The first requires a labelled dataset to train the model on. This is by far the least common class of approaches in anomaly detection since in most scenarios data might be plentiful, but accurate labels are difficult to obtain especially for the rare anomalous class. For this reason, unsupervised approaches are the most common since they do not require any labelled data.

Unfortunately unsupervised approaches are based on generic definitions of anomalies that might not coincide with the definition the end-user would provide and expect to see [217]. Moreover, these models are based on generic definitions of anomaly that typically reflect their detection strategy: as a consequence, each unsupervised model is best suited to the detection of specific sets of outliers. For example Isolation Forest assumes outliers are few and far from normal data therefore are easy to be separated. However this is

[1] Other authors in the literature refer to 'Anomaly Detection' only in case of unsupervised settings, labeling the supervised or semi-supervised scenario as 'Fault Detection'.

not always the case as in some contexts outliers form a dense cluster and the previous anomaly definition might fail. Similar issues can arise using other detection strategies based for example on density, distance or probabilistic models. As [217] highlights, domain knowledge can be used to select the best approach and to encode the user definition of anomaly into the model. This issue could be solved using supervised models but, as previously discussed, these methods require labelled data that can be hard to obtain. To solve this problem in recent years two types of semi-supervised approaches have been proposed to encode available information inside an already existing detection model. (i) If few labelled samples are available it is possible to frame the problem in a *weakly-supervised* (WS) scenario and to take advantage of the available labels by updating the model accordingly. (ii) Another promising approach is to let the model interact with the domain expert. The model can *actively learn* (AL) from a pool of unsupervised data, by asking a human expert to label only a small subset of the full dataset, saving a lot of human effort but at the same time improving the model rapidly. This approach has been extensively exploited in classification context [166, 219] and only recently also in anomaly detection [61] under the name of Active Learning.

The aim of this study is: i) to develop a model that in presence of few available labels can be updated, and ii) to develop a model able to interact with the end-user and to incorporate the information iteratively provided, iii) to frame the previously discussed ALIF algorithm in a more mathematically grounded scenario. Indeed the main goal is to develop not only a model able to update its structure, but also to query the label of the most informative point in order to reduce the necessary labels and save human effort. This is particularly appealing in contexts where Decision Support Systems are in place, allowing a fluid interaction between the detection model and the human expert.

Only recently the research community pointed its attention towards the introduction of expert feedback into unsupervised detection models, like One Class-SVM or IF [149]. Indeed, despite the existence of multiple variants to the original Isolation Forest [26], the most are unsupervised algorithms that are not able to cope with a actively supervised scenario. An exception is IF-AAD [62], that based on the feedback received, is able to adjust the leafs of an Isolation Forest by means of an iterative optimization procedure. A completely different approach that will be used as baseline is the popular Random Forest [34].

In the proposed approach, the main idea is to start from an unsupervised model, the popular Isolation Forest, and then to develop model updates and sample query strategies, similarly to the ones discussed in Chapter 7 where ALIF was discussed. While ALIF presents one of the first approaches in the literature to deal with active learning with Isolation Forest, it has some major weaknesses that are overcomed with the approach presented in this Chapter. Indeed even if ALIF works well in practice, it is completely based on heuristics. Moreover due to its simple updating procedure that do not take into account model uncertainty, it may have an unstable learning process and in presence of dense anomalous areas it suffers of slow learning rates that worsen the algorithm data efficiency. The proposed approach, named B-ALIF, solves the sames problem in an original way by formalizing it in a Bayesian framework and therefore proposing a new approach more statistically grounded and more effective.

The rest of the Chapter is organized as follows: after the description of the applicative problem, Section 8.2 gives an overview of the preliminaries on which this work is developed, i.e. the Isolation Forest algorithm (Section 8.2.1), and some basics on Bayesian inference (Section 8.2.2). Afterwards, in Section 8.3, we will discuss an alternative theoretical formulation of the ALIF model, that leverages basic probabilistic programming notions to re-frame the existing algorithm from a Bayesian point of view. Under this framework, we will discuss the possible extensions that come natural under the new interpretation, as well as the advantages brought forward by the richer description. At the end of Section 8.4, results are shown that prove the efficacy of the proposed method and conclusions are drawn in Section 8.5.

## 8.2 PRELIMINARIES

### 8.2.1 *Isolation Forest*

This Section will recall the IF algorithm explained in Chapter 4 and will provide the required notations to understand the proposed algorithm.

The IF algorithm is somehow similar to the popular supervised algorithm Random Forest (RF) [111], being an ensemble tree-based algorithm. As such, the model training consists in building a specific data structure represented by an ensemble of $n_T$ binary trees, namely a forest $F = \{T^1, T^2, ..., T^t, ..., T^{n_T}\}$, using an unlabelled dataset $X^u = \{x_1^u, x_2^u, \dots, x_{n_u}^u\}$. Each node in a given tree $T^t$ identifies a specific region of the data domain, so that the collection of its leaves $\{L_1^t, L_2^t, ..., L_i^t, ..., L_{n_t}^t\}$ defines a partition of the data domain. Each tree in the ensemble is constructed independently from the others as follows. The dataset is randomly sub sampled to size $\psi$, so that only a small portion of the original data is used in the construction of each tree. The root node is initialized to represent the whole domain. The tree is grown recursively at each node by dividing the domain region they identify with a random cut aligned with an axis, splitting the node into its two children. This process repeats until only one data point of the sub sample reaches a particular node, or when a depth $h_{max}$ is reached. Terminal nodes for this process are the leafs of the tree and each is associated with a specific domain region. We will refer to both by $L_i^t$. This "terminal" region identify a partition of the original domain. As such, given a point $x$, it lies inside one and only one terminal region $L_i^t$ for a given tree in the ensemble $T_t$, and as such an unique path in the tree from the root to that specific terminal region. The subscript $x$ will be used in the rest of the Chapter to indicate $L_x^t = L_i^t \ni x$, the leaf where the point $x$ lies. When referring to a generic partition of the space we will simply write $L$ or $L_i^t$.

Using the same notation, one can identify the "depth" of a data point for a given tree as the depth of the terminal region containing that point:

$$h_x^t = \text{depth of } L_x^t. \tag{15}$$

Once the model is trained, a point can be evaluated by considering the path taken in each of the trees. In particular, we are interested in evaluating the average observed path length $E_t[h_x^t]$: the rationale is that anomalies should, on average, be easier to be separated from other data points, and, therefore, have a shorter path lengths associated to them. The average depth of a point is associated to an anomaly score $s(x)$ in the interval $[0, 1]$, following an exponential map:

$$s(x) = 2^{\frac{-E_t[h_x^t]}{c(\psi)}}, \tag{16}$$

where $c(\psi)$ is the expected average path length already discussed in Chapter 4.

## 8.2.2 *Bayesian Inference*

In the field of AL, if the models under consideration are probabilistic, the integration of new labelled data might require an inference step [29]. This topic is an extremely vast one, and in this Section we will limit ourselves to a brief introduction of the main concepts needed to fully understand what will be presented in later Sections.

Being $\Theta$ the parameters of a given model, one could ask how they are distributed given some observed data $\mathcal{D}$ and some knowledge of the data generating process. By applying the simple Bayes formula, it is possible to reduce the problem to [29]:

$$p_{\Theta|\mathcal{D}}(\theta \mid d) = \frac{p_{\mathcal{D}|\Theta}(d \mid \theta)\; p_\Theta(\theta)}{p_\mathcal{D}(d)} = \frac{p_{\mathcal{D}|\theta}(d \mid \theta)\; p_\Theta(\theta)}{\int p_{\mathcal{D}|\theta}(d \mid \theta)\; p_\Theta(\theta)\; d\theta}, \tag{17}$$

where $p_{\mathcal{D}|\theta}(d \mid \theta)$ represents the likelihood (or generative model), $p_\Theta(\theta)$ is the prior belief about the model parameters, $p_\mathcal{D}(d)$ is named evidence and $p_{\Theta|\mathcal{D}}(\theta \mid d)$ is the posterior distribution of $\Theta$ that can be inferred from the data $\mathcal{D}$.

One obstacle in obtaining the posterior probability in most practical cases is the evaluation of the evidence term. This term is essentially only a normalization condition, and can be expressed in terms of the likelihood. However, obtaining its exact value requires the solution of a complex integral and in general it might be extremely hard to do analytically. In some special cases, the solution to this integral can be written explicitly, but this is an exception rather than the norm. The most common scenario of this kind is the one where the prior and posterior are part of the same family of probability distributions. In this case, this family is said to be conjugate of the given likelihood function. If the likelihood function admits a family of conjugate priors, then it is usually beneficial to select the original prior inside such family. Notably, all distributions in the exponential family admit conjugates. One notable example is the Normal distribution that is its own conjugate, but in this work we will use mostly the fact that the Beta distribution is the conjugate for Bernoulli and Binomial likelihoods. Since the evidence always normalizes the resulting distribution so that its integral is unitary, it is sufficient to show proportionality, and therefore the demonstration of this examples is very straightforward and only requires some basic algebraic manipulations. On this note we remind the most common ways to parameterize the Beta distribution as:

$$g(z) \sim \text{Beta}(\alpha, \beta) \propto z^{\alpha-1}(1-z)^{\beta-1}. \tag{18}$$

Alternatively, in some case it might be easier to represent the distribution in terms of its mean $\mu = \frac{\alpha}{\alpha+\beta}$ and sample size $\nu = \alpha + \beta$:

$$g(z) \sim \text{Beta}(\mu{:}\overline{\mu}, \nu{:}\overline{\nu}) \propto z^{\overline{\mu}\overline{\nu}-1}(1-z)^{(1-\overline{\mu})\overline{\nu}-1}. \tag{19}$$

One convenient aspect of dealing with the exponential family of distributions is that it is possible to express the Bayesian inference step in terms of simple update rules on distribution parameters.

This comes handy in computation, since there is no need for trapezoidal integration or approximate sampling, only a few additions and multiplications are required. In particular, when dealing with a Beta distributed prior $g(z) \sim \text{Beta}(\alpha, \beta)$, and a Binomial distributed likelihood $lk(z \mid k) \sim \text{B}(n, z) = k$, the posterior $g(z \mid k)$ becomes:

$$g(z \mid k) \propto g_0(z) lk(z \mid k) \propto z^{\alpha-1}(1-z)^{\beta-1} z^k (1-z)^{n-k} \sim \text{Beta}(\alpha+k, \beta+n-k). \tag{20}$$

The same relation also holds for a Bernoulli likelihood, that can be seen as a Binomial with parameter $n = 1$.
Therefore, the Bayesian update step for this cases reduces to simple additions, dependent on to the observed realization $k$.

## 8.3 BAYESIAN ALIF (B-ALIF)

Like ALIF, also B-ALIF is meant to be employed both on weakly-supervised (WS) and active learning (AL) scenarios. We stress that for weakly supervised scenario we mean scenarios where after a fully unsupervised training, some random labels become available and the user wants to exploit this information to update the model. On the other side we speak about active learning scenarios when the detector is allowed to interact with the domain expert. This distinction shows up in the way the model is designed, indeed it is made up of two main routines: the updating strategy, used for both WS and AL, and the query strategy used only in the AL scenario. In this Section we are going to show how these two routines adapts to the Bayesian framework introduced in B-ALIF.

### 8.3.1 *B-ALIF updating strategy*

The idea at the core of ALIF is simple yet effective. By assigning abstract information to nodes, we can maintain the efficiency of the IF tree structure, but we are also able to modify the model and adjust its predictions with almost no overhead. The iterative updating procedure of node parameters according to new information gathered by labelled data, is, in some ways, reminiscent of Bayesian updating on Conjugate priors as introduced in Section 8.2.2.

The core idea of this generalization is to allow the abstract information stored in a node to be uncertain, that is, instead of storing exact observable information, we will keep track of a probability density on the dimensions of interest. We call this approach Bayesian-ALIF (B-ALIF). In this context, we can take a step back, considering the task of AD, and what anomaly scores might represent. Given an observation $x$ in a region $L$, we are interested in assessing the probability that such point is an anomaly. We try to approximate this info by assigning to each point an anomaly score $s(x)$, such that higher anomaly scores represent higher probability of being an anomaly, but there is no real effort to make the two coincide. However, in an ideal world, we would like the two to coincide, and, in the context of Bayesian inference, working with anomaly probabilities directly is a simplifying assumption to obtain sensible results.

### *Bayesian Leaf Update*

Both ALIF and B-ALIF inherit the space partitions from the unsupervised IF, but estimate the anomaly rate of each partition (leaf) in a different way. While ALIF simply estimates it by taking the ratio between anomalies and points sampled inside the leaf, B-ALIF formalize the problem in a theoretical way using Bayesian arguments.

Therefore, we will start by directly trying to model the anomaly rate of a given region $L$, that is the probability that a random observation $x$ inside such region is anomalous:

$$\mathrm{Ar}(L) = \mathrm{P}\left[x \text{ is anomaly } \mid x \in L\right] \tag{21}$$

For this reason, we will keep track of the abstract propriety $\mathrm{Ar}(L)$ of each node in the form of a probability distribution on the interval $[0, 1]$.

$$g_L(z) = \mathrm{P}\left[\mathrm{Ar}(L) = z\right] \tag{22}$$

We will be assume the anomaly rate of non overlapping regions to be independent unless otherwise stated.

Similarly to ALIF, we will define model predictions on a point $x \in L_x^{\cap} = \cap_t L_x^t$ as a function of abstract node proprieties, as:

$$s(x) = f\left(g_{L_x^0}, g_{L_x^1}, \ldots, g_{L_x^t}, \ldots, g_{L_x^{n_T}}\right) \tag{23}$$

Under this formulation, the incorporation of new data point $x$ with label $y$ into the model consists of modifying the beliefs on the anomaly rates of different terminal regions, that we assume to be independent.
Considering each tree $T$ in the ensemble separately we can write for any region $L$ where the point $x$ lies:

$$g_L(z \mid x \in L, y) = \frac{g_L(z)\mathrm{P}\left[x \in L, y \mid z\right]}{\mathrm{P}\left[x \in L, y\right]} \tag{24}$$

We are of course interested in studying in particular the structure of the likelihood. With some elementary manipulation, we can rewrite it as:

$$\begin{aligned} \mathrm{P}\left[x \in L, y \mid z\right] &= \mathrm{P}\left[y \mid x \in L, z\right] \mathrm{P}\left[x \in L \mid z\right] && (25) \\ &= z^y (1-z)^{1-y} \mathrm{P}\left[x \in L \mid z\right] && (26) \end{aligned}$$

Where we leveraged the fact that the label probability conditioned on the anomaly rates of regions is Bernoulli distributed with parameter $z$. Let us also assume for the time being that $\mathrm{P}\left[x \in L\right]$ is independent of the anomaly rates of the other regions containing $x$ and therefore independent of $z$. This of course can only be the case if points are sampled independently of region's anomaly rates. This assumption can hold in WS scenarios but it for sure not going to be true if the model is inserted into an AL loop. We will discuss the validity of this assumption more in depth in the conclusive Section.

Under this assumptions, the likelihood reduces to a simple Bernoulli distribution, for which we have shown in previous Section 8.2.2 that exists a family of conjugate priors in the form of Beta distributions. Therefore, if we restrict our abstract node proprieties to be Beta distributed, we can very efficiently perform this Bayesian update step analytically and in closed form. Moreover, to exactly characterize the distribution $g_L(z)$ we only need to store two parameters, $\alpha$ and $\beta$. We can also identify explicitly the *update rules* for $\alpha$ and $\beta$ by considering

$$g_L(z) \sim \mathrm{Beta}(\alpha, \beta) \propto z^{\alpha-1}(1-z)^{\beta-1} \ , \tag{27}$$

$$g_L(z \mid x \in L, y) \propto z^{\alpha-1}(1-z)^{\beta-1} z^y (1-z)^{1-y} \sim \mathrm{Beta}(\alpha+y, \beta+1-y) \ . \tag{28}$$

It all essentially boils down to adding one to either one of the two parameters, according to the label $y$, and only in the region of interest where the point $x$ is found. This means that for every generic leaf $L_i^t$, we can fully characterize the abstract node propriety $g_{L_i^t}$ with only the two values $\alpha_i^t$ and $\beta_i^t$, that will be updated following the simple update rules previously described each time a new labelled point is incorporated into the model. The relation can be expressed explicitly in terms of the original values of $\alpha_i^t$ and $\beta_i^t$ (denoted by $\hat{\alpha}_i^t$, $\hat{\beta}_i^t$) and the number of observed labels of each class (denoted by $O_i^t$, $I_i^t$) obtaining: $\alpha_i^t = \hat{\alpha}_i^t + O_i^t$ and $\beta_i^t = \hat{\beta}_i^t + I_i^t$.

*Initialization and IF prior*

Since in B-ALIF node abstract proprieties parameterize a probability distribution, initialization of model parameters $\hat{\alpha}_i^t, \hat{\beta}_i^t$ can be interpreted as the selection of a prior distribution for $g_{L_i^t}(z)$. To set an uninformative prior, we can set $\hat{\alpha}_i^t = \hat{\beta}_i^t = w_0$, obtaining a totally uniform distribution over the interval $[0, 1]$ for $w_0 = 1$. If however, we want to bias the distribution towards extreme values (to represent that anomaly rates tend to be polarized), we can simply set $w_0 < 1$, with lower values of $w$ representing a larger bias of this kind.

However, the constraint $\hat{\alpha}_i^t = \hat{\beta}_i^t$ is not optimal. In fact, we do not start from a totally uninformed prospective, since we can rely on the original prediction made by the unsupervised IF model.

Therefore, one degree of freedom in the distribution can be set by enforcing the model prediction to match the underlying IF forest when no relevant labels are provided. This can be easily translated in a constraint on the mean of the distribution (for reasonable choice of ensemble prediction function). In order to fully characterize the distribution, the sample size $\hat{\nu}_i^t = \hat{\alpha}_i^t + \hat{\beta}_i^t$ needs to be determined. $\hat{\nu}_i^t$ sets the "strength" of the prior relative to the weight of new observations. ALIF completely disregards IF predictions as soon as a single relevant labelled point is obtained, and it has been shown to perform well in practice, this suggests that setting a weak prior should be a pragmatic choice, allowing the model to evolve quickly.

Therefore, we set the second degree of freedom in the parameters by minimizing $\hat{\nu}_i^t$, subject to $\hat{\alpha}_i^t \geq w_0$ and $\hat{\beta}_i^t \geq w_0$. The rationale of this strategy is to add the minimal amount of fictitious observations ($\Delta\alpha_i^t = \hat{\alpha}_i^t - w_0$, $\Delta\beta_i^t = \hat{\beta}_i^t - w_0$) needed to shift an uninformative prior for a set bias value $w_0$, to one that matches the IF prediction. For instance, for a flat uninformative prior ($w_0 = 1$), if the original IF prediction $s_0(L_i^t) = 2^{-\frac{h_i^t}{c(\psi)}}$ is matched to the mean of the distribution $g_{L_j^t}$, the constrains reduce to $\hat{\alpha}_i^t = \frac{s_o(L_j^t)}{\frac{1}{2}-|s_o(L_j^t)-\frac{1}{2}|}$ and $\hat{\beta}_i^t = \frac{1-s_o(L_j^t)}{\frac{1}{2}-|s_o(L_j^t)-\frac{1}{2}|}$.

While this approach for modelling the prior distribution allows for straightforward extension of the IF algorithm by retaining the original model predictions, one important limitation of this method that needs to be considered is that the anomaly scores obtained by the IF model are not meant to be probabilities and in general do not span the entire range (0,1). Due to the truncation applied in the tree building process, low anomaly scores are virtually impossible to achieve, even for points that are extremely unlikely to be anomalous. Therefore priors obtained this way are highly biased towards larger anomaly rates. In this work we do not explore in depth this aspect, but simply try to balance this effect by linearly re scaling the anomaly score range to the more desirable interval $(0, 1)$. Therefore, B-ALIF does not exactly match the IF predictions, but rather a re scaled version of it. This is by no means an optimal solution to the issue, but we will discuss it more in depth in the conclusions Section.

### *Anomaly score*

To provide a prediction for a given point $x$, we need to combine the distributions of model beliefs on the anomaly rate of individual leafs that contain $x$ into a unique anomaly score (Equation 23). Therefore, in the context of B-ALIF, if two points $x_1, x_2$ are such that $L_{x_1}^{\cap} = \cap_t L_{x_1}^t = L_{x_2}^{\cap} = \cap_t L_{x_2}^t$, the model predictions are bound to be the same. As such, the best prediction for any point $x$ that can be achieved in this way is closely related to the anomaly rate of the region $L_x^{\cap}$. Therefore, the anomaly score can be represented naturally as a likelihood distribution $G_x(z)$ on the anomaly rate of $L_x^{\cap}$, that can be summarized by a single value $s(x)$ if needed. There are multiple possible formulations depending on the chosen assumptions, in this Section we want to present two possibilities: *likelihood* based on strict condition $Ar(L_x^\cap) = Ar(L_x^t)$, and *naive* based on the assumption that $Ar(L_x^t) \approx Ar(L_x^t \setminus L_x^\cap)$ .

LIKELIHOOD ENSEMBLE PREDICTION This ensemble prediction method is based on the assumption that the true anomaly rate does not vary significantly in the neighborhood of the point $x$. This assumption can be translated into imposing an equality of anomaly rates across all terminal regions containing the point $x$, and their intersection.

$$Ar(L_x^\cap) = Ar(L_x^0) = Ar(L_x^1) = \ldots = Ar(L_x^{n_T}) \tag{29}$$

From this we obtain a simple formulation for the likelihood of $\mathrm{Ar}(L_x^\cap)$:

$$G_x(z) = \mathrm{P}\left(\mathrm{Ar}(L_x^\cap) = z\right) \propto \prod_t \mathrm{P}\left(\mathrm{Ar}(L_x^t) = z\right) \tag{30}$$

From this, we can easily recover an explicit formulation for the likelihood $G_x(z)$ as the product of $t$ Beta distributions. Making explicit the dependence of the parameters $\alpha_x^t$ and $\beta_x^t$:

$$G_x(z) \propto \prod_t z^{\alpha_x^t - 1}(1-z)^{\beta_x^t - 1} \sim \mathrm{Beta}\left(1 + \sum_t (\alpha_x^t - 1)\ ,\ 1 + \sum_t (\beta_x^t - 1)\right) \tag{31}$$

Conveniently, this formulation not only provides us with a simple closed form solution for $G_x(z)$, but gives us a straightforward way to determine the anomaly score $a(x)$ as the maximum of the likelihood function, that can be recovered from the mode of the Beta distribution.

$$s(x) = \mathrm{maxlikelihood}\, G_x(z) = \frac{\sum_t (\alpha_x^t - 1)}{\sum_t (\alpha_x^t + \beta_x^t - 2)} \tag{32}$$

This result is especially neat, simplifying model's predictions to the proportion of excess anomalies in all terminal regions across the ensemble relevant to the point $x$. This once again, does not involve the solution any optimization problem but can be obtained in closed form.

NAIVE ENSEMBLE PREDICTION The second ensemble prediction approach is based on the assumption that for a given point $x$ the intersection of its terminal regions $L_x^\cap = \cap_t L_x^t$ is much smaller than each individual term $L_x^t$, and as such the anomaly rate of the region $L_x^t \setminus L_x^\cap$ is approximately equal to the one of $L_x^t$, that is $Ar(L_x^t \setminus L_x^\cap) \approx Ar(L_x^t)$. With some simple algebraic manipulation, this assumption imposes the relation:

$$\mathrm{Ar}(L_x^\cap) \approx \frac{1}{n_T} \sum_t \mathrm{Ar}(L_x^t) \tag{33}$$

Unfortunately, the distribution of the average of $n$ beta distributed random variables is not known explicitly in closed form, but for large $n_T$, it can be well approximated as a Beta distribution. Therefore, following this approximation, by equating the distribution moments we obtain:

$$G_x(z) \sim \text{Beta}\left(\mu: \frac{1}{n_T}\sum_t \frac{\alpha_x^t}{\alpha_x^t + \beta_x^t},\ \nu: \sum_t (\alpha_x^t + \beta_x^t)\right) \tag{34}$$

Using this expression for the distribution for the anomaly rate of the intersection, the anomaly score of the point $x$ can be computed ad the probability of it being an anomaly under the approximate distribution $G_x$.

$$s(x) = \text{E}_{G_x}\left[\text{P}[\text{x is anomaly}|\,\text{Ar}(L_x^{\cap}) = z]\right] = \frac{1}{n_T}\sum_t \frac{\alpha_i^t}{\alpha_i^t + \beta_i^t} \tag{35}$$

Which is exactly the mean of the distribution $G_x$.

### 8.3.2 *Connections with ALIF leaf update*

In this section, we will explore the connections between the ALIF algorithm and B-ALIF. In fact, one might recognize that the parameters used in B-ALIF are closely related to the abstract proprieties defined in ALIF, since $\alpha_i^t = \hat{\alpha}_i^t + O_i^t$ and $\beta_i^t = \hat{\beta}_i^t + I_i^t$, and in fact coincide in the limit for the prior weight $w_0$ going to zero. Also note that in this case, the node's color under ALIF exactly matches the mean of the belief distribution in B-ALIF.

$$k_i^t = \frac{O_i^t}{I_i^t + O_i^t} = \text{mean}(\text{Beta}(\alpha_i^t, \beta_i^t)) = \frac{\alpha_i^t}{\alpha_i^t - \beta_i^t} \tag{36}$$

In this sense, we can interpret ALIF as a specific case of B-ALIF, where labelled data is assumed to be sampled independently of individual region's anomaly rates, the prior weight is set to zero, and the ensemble prediction function is simply the geometric average of log-predictions of individual trees.

For instance, if we consider the *logarithmic* variant of ALIF, it is possible to write the equivalent prediction function for a point $x \in \cap_i L_x^{T_i}$ in terms of beliefs distributions in a concise way:

$$s(x) = 2^{-\frac{E[-c(\psi)log_2(k_x^t)]}{c(\psi)}} = \sqrt[n]{\prod_i k_x^t} = \sqrt[n]{\prod_i \frac{\alpha_x^t}{\alpha_x^t + \beta_x^t}} \tag{37}$$

### 8.3.3 *Query strategy*

When it comes to the query strategy, we can make the same considerations made for ALIF, where actively trying to query the most anomalous points in terms of anomaly score is likely to result in improvement of performance. However, we also propose an alternative to the uncertainty based one. While for ALIF, model uncertainty was assessed by considering the disagreement

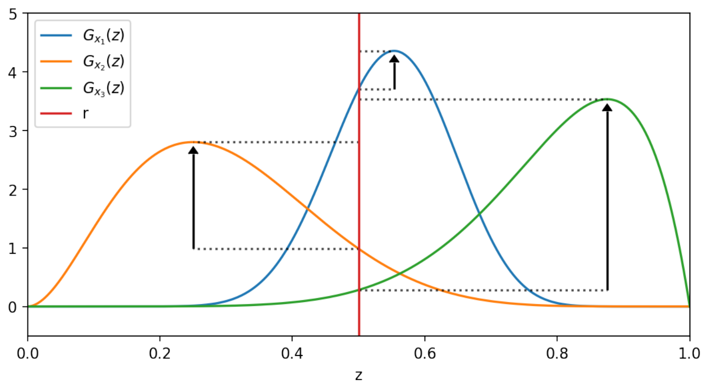


Figure 50: Example of visualizations of likelihood margins for different likelihood prediction functions. The figure illustrates how the margin values take into account both the location of the peak and the dispersion of the distribution

across different trees in the ensemble, and this could very well be reproduced one to one in B-ALIF, the reinterpretation under the Bayesian framework naturally gives us a measure of model uncertainty in the form of a likelihood distribution we can derive analytically and exactly.

In fact, since model predictions are naturally expressed as a likelihood function, and are only afterwards condensed into a single maximum likelihood value when an anomaly score is needed, we can assess how localized are the model beliefs on the anomaly score of a specific point. If the likelihood is flat, the model is unsure about the true value. Moreover, if the range of most likely values is near the decision boundary, small deviation in the mode might result in a qualitative change in prediction, however, if the range is near the extremes, further information is unlikely to change the model output. For this reason we propose a query strategy based on the likelihood ratio evaluated at the mode, and at a fixed point $r$:

$$\text{margin}_r(x) = \frac{\max\limits_z G_x(z)}{G_x(r)} \tag{38}$$

This represents the likelihood ratio between the most likely values for the positive and negative classes, if the decision boundary was set to $r$. An uninformative choice of $r$ is 0.5, while for $r = 1$, querying the points with largest margin is equivalent to the "most anomalous" strategy. In general, in-between values represent a trade off between the two approaches, as "margin on $r$" query strategy tries to clear unsure predictions in the region of the vicinity of $r$, in order to perfectly separate the two classes.

## 8.4 RESULTS

In this Section we present some experimental results both on synthetic data sets and on real world data. We follow a similar testing procedure to the one employed in the original ALIF work, giving in depth considerations on synthetic tests, and introducing an optimization for the anomalous querying strategy. We then show real world performance of the AL loop comparing across B-ALIF variants and to ALIF.

At its core, most tasks in anomaly detection can be interpreted as a binary classification problem, even if highly unbalanced, where anomalies need to be separated from normal data instances. Even when the model does not directly classify points into anomaly or inlier, but yields for instance an anomaly score, it generally needs to be converted into a classifier by finding an appropriate threshold. Unfortunately, as discussed in [212], when the dataset is highly unbalanced like in anomaly detection settings, true positive rate and false positive rate are not reliable metrics, so in this work we will use precision and recall to measure the detection performances. In particular, to obtain an unique score that assess the detection performances independently on the threshold we will use the average precision score.

### 8.4.1 *Experiments on synthetic dataset*

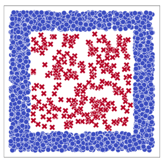
Following the experimental setup of Chapter 7, we test the proposed algorithm on a synthetic dataset represented by a two-dimensional square toroidal shaped ring of densely packed normal data points that surrounds a sparse cloud of anomalous points in the center (Figure 51a). This particular setup is especially challenging for the IF algorithm, since the anomalous central points are more difficult to isolate compared to normal points at the edges of the domain.

On this setup, ALIF has been shown to be able to quickly fix the poor predictions of the underlying IF, achieving high performance with just a few iterations. However, we find that both ALIF and B-ALIF performance on this dataset are highly subject to changes in hyper parameters of the data generation process.

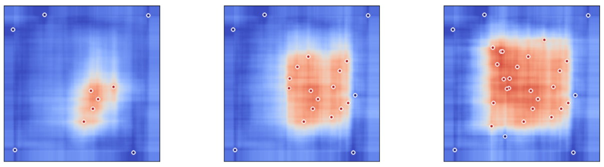
These effects seem to depend mainly on the querying strategy employed, therefore in the following we will present a few different scenarios, and provide some basic reasoning to explain the observed changes in behaviour.

Firstly, we consider a baseline case, where roughly 500 data points are generated, with approximately a 10% anomaly rate. To obtain training and test data, we then perform a 50-50 stratified random split. This is the setup was already presented in Chapter 7. By running simulations, we can qualitatively describe what the algorithm tries to do.

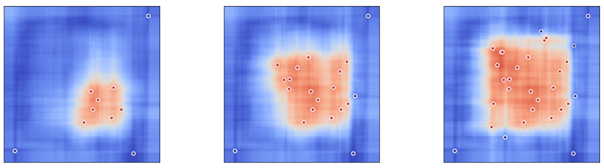
By querying for the most anomalous datapoint (Figure 51b), the model firstly asks labels of points near the edges of the domain, following the inaccurate predictions of the baseline IF model. Doing so, it quickly learns that the edges of the domain are unlikely to be anomalies. Soon, the algorithm will start to query points in the central region, finding its first anomaly. When this happens, the models predicts point in the neighbourhood of the anomaly to be likely anomalous as well. This starts a propagation process, where the area of anomalous predictions quickly spreads with each successive query on anomalous points, until the anomalous region is perfectly identified. For this particular choice of data generation parameters, the whole process takes in the



(a) Dataset



(b) *Anomalous* query strategy



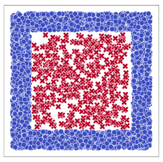
(c) *Margin* query strategy

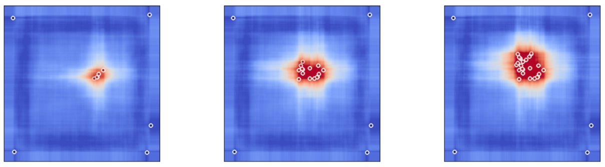
Figure 51: Square toroidal datasets and qualitative evolution of BALIF predictions on the synthetic "square toroid" dataset after 10, 20 and 30 queries. Low density experiment.

order of 30-40 queries to end, depending on how efficiently the algorithm is able to find its first anomaly, and how many successive queries are spent on correctly identifying the exact region boundary. However, we can intuitively understand that this process is highly dependant on the density of the anomalous region.

In fact, if we consider a dense version of the same dataset, 10 times the size of the original one, we can clearly see that it takes the algorithm much longer to completely profile the anomalous region (Figure 52b). This is simply because the velocity at which the algorithm is able to profile an anomalous cluster, is inversely proportional to the density of data in such region, and therefore it is severely slowed down in the dense version of the dataset.

However, it is possible to somewhat counteract this unpleasant effect by introducing a sub sampling step (Figure 53a). If we only allow the model to ask labels amongst a random subset of the original data, the dependency on density can be greatly reduced.
Doing so the model is no longer able to profile the region by small incremental steps, but instead is forced to take large jumps, speeding up the procedure. Sampling also reduces the computational complexity of the update step, since it is no longer needed to evaluate the model on the whole dataset. This can be especially useful in online applications or for performing updates on streaming data.


(a) Dataset


(b) *Anomalous* query strategy

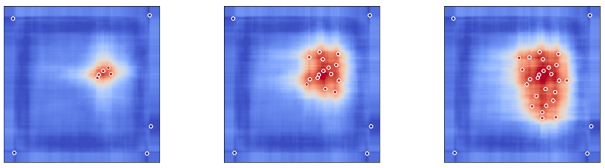
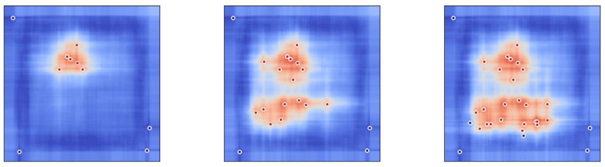

(c) *Margin* query strategy

Figure 52: Qualitative evolution of BALIF predictions on a *dense* version of the synthetic "square toroid" dataset after 10, 20 and 30 queries.


(a) *Anomalous* query strategy

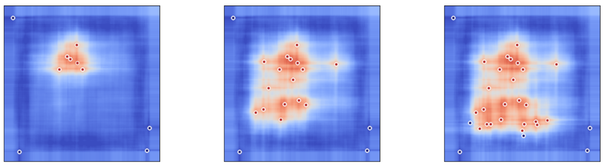

(b) *Margin* query strategy

Figure 53: Qualitative evolution of BALIF predictions on a *dense* version of the synthetic "square toroid" dataset when using query *subsampling* after 10, 20 and 30 queries.

When switching to the "margin" query strategy, the aforementioned dependence on data density is greatly reduced (Figure 52c). In fact, when using this strategy, the model is not interested in labels of points in the vicinity of already queried ones, unless they are close to a boundary between anomalous and normal regions.

However, while overall being a more robust method, the margin strategy heavily relies on the proper selection of threshold value $r$ for the margin strategy. If the value $r$ is not representative of the "uncertainty" region, the algorithm might run through a significant portion of the labelling budget trying to improve its accuracy on anomaly rate of regions, while failing to improve in the classification task. This effect is also very sensible to any bias in the prior distributions originated from the underlying IF. This is because any bias of this kind will introduce a shift in the "uncertainty" region, and therefore directly impact the validity of the margin strategy.

This being said, each query strategy has its merits, and depending on the structure of the dataset, either one can outperform the other in practice.

## 8.4.2 *Experiments on real world data sets*

In this Section we evaluate the model's performance on multiple real world data sets for anomaly detection.

The data sets can be found on the ODDS website [204], and are derived from multi-class classification problems, where one underrepresented class is considered as anomalous, while the others represent the normal data. For this reason, anomalies in these data sets aren't random points out of distribution, but rather live in specific regions of the domain, and present specific patterns. This makes this kind of data sets difficult to solve in a completely unsupervised setting.

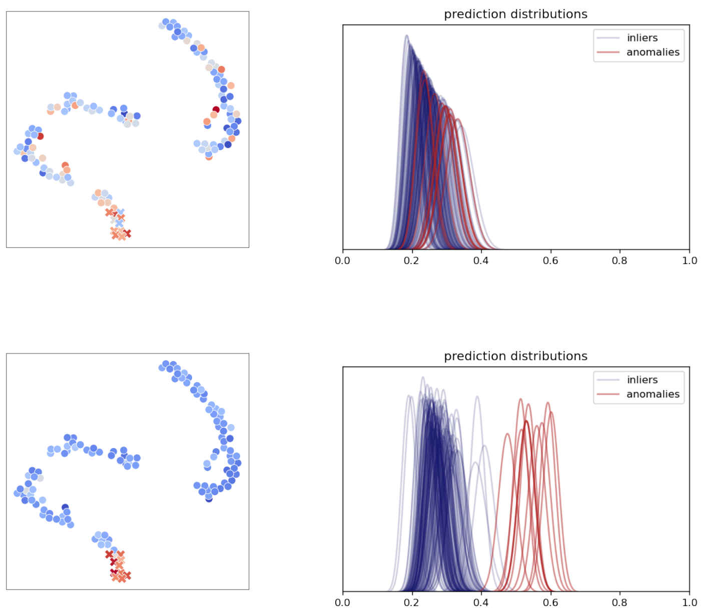


Figure 54: Comparison of B-ALIF predictions (left) and belief distributions (right) on the "wine" dataset without labelled data (top row), and after 10 queries (bottom row). Results are qualitatively similar amongst different model configurations, as well as quantitatively (see Figure 56 plots).

To remove as much as possible any bias coming from the selection of the train-test split (as described in the previous section), all splits are performed on a 80-20 basis, and the model is only able to ask info for points in a random subset of the original training set. This subset is different at each iteration and its size is at most 512.

To give qualitative idea of the results obtained with the B-ALIF algorithm, in Figure 70d we show the results on the wine dataset before and after 10 iterations of the B-ALIF algorithm with anomalous querying strategy. In the left Figures it is possible to see a t-SNE projection of the wine dataset on two dimensions where each point is coloured with the associated anomaly score, and the true anomalies are marked with a cross. On the right for each point a distribution is depicted showing the ensemble prediction. It is easy to see that without supervision the predictions are very poor, while after 10 iterations (Figure 54b) the true anomalous points are much more separated.

In doing the experiments we were guided by three main research questions:

1. In a weakly supervised scenario where it is not possible to query a domain expert or where few labelled data are already present, which is the best model among ALIF-based methods and other more classic approaches like the Random Forest?

2. If the domain expert is available, is it worth to let the model query some points, or it is better just to label random samples? Which strategy is the most useful for B-ALIF?

3. Among the active learning algorithms that are based on IF, which performs the best?

We try to answer Question 1 evaluating the performance evolution for models following a random query strategy that simulates the passive weakly supervised approach where the provided information is sampled without a specific strategy (Figure 55). This indeed allows us to assess performance of ALIF and B-ALIF when used as weakly supervised approaches, by eliminating the AL step. We then compare it to a random forest trained only on the labelled data. We indeed find that B-ALIF provides a steady improvement over the pure IF, and is generally preferrable to the random forest when labelled data is scarse, however the benefits of the weakly supervised approach are highly dependent on the dataset considered. In general, this method shines the most when the underlining IF achieves moderately good performance, that can however be refined by incorporating information from labelled points into the model. It is interesting to note in this passive setting B-ALIF is systematically better or equal than ALIF.

We then try to answer Question 2 looking at the difference between active and passive query strategies for the B-ALIF algorithm. Secondly, we evaluate the effects of different query strategies (anomalous or margin) for B-ALIF, as well as the effects of different choices of ensemble prediction function (Figure 56). We use as a baseline for comparison the passive (weakly-supervised) variant of B-ALIF (random query), to assess the impact of the AL step on performance. We find that the ensemble prediction function seems to have little impact on the overall model behaviour, with the naive variant being the best one in most cases. Conversely, the querying strategy used has a much more significant impact. Comparing the two proposed approaches, the margin strategy seems to provide a more stable option, being consistently better than the random query process. This is not the case for the anomalous query strategy, that even if slightly optimal for some datasets, can sometimes perform worse than querying random points. However in general we see that AL strategies performs much better than passive ones.

Finally in Figure 57 we answer to Question 3 comparing the active approaches i.e. B-ALIF (margin query strategy), ALIF, and IF-ADD. The perfomance are quite dataset dependent, however it is clear B-ALIF (green) generally performs best, followed by ALIF (blue) and then IF-AAD (red).

## 8.5 conclusion

The motivation of this work was based on extending the core ideas presented in Chapter 7 into a rigorous Bayesian framework.
In doing so we shed some light on the underlying reasons for the efficacy of the simple heuristic rules employed in the original work, as well as proposing some extensions that come natural under the new formulation. This Bayesian interpretation of the ALIF algorithm has proved to be a viable alternative, giving multiple extra dials that can be used to adapt the model to specific situations in a natural way.
This include setting a relative strength of the prior in order to give more or less importance to the original underlying IF model, to an alternative formulation of the query strategy that is more in line (in spirit) with the core ideas of AL, asking for information that clears up uncertainty in the model.
We also explored more in depths the trade-offs of different query strategies, by qualitatively illustrating the changes in model behaviour on a simple toy dataset.
We introduced the idea of sub sampling to counteract the poor dependency of the anomalous strategy on data density, keeping the benefits of biasing the search towards anomalous data, for which info is more valuable.

Moreover, the proposed model works directly in the space of anomaly rates, relaxing the need to define an heuristic map from some abstract space to the interval $[0, 1]$.
In this context, model prediction represent the actual confidence of the model on the probability that a given observation is in fact anomalous. This gives not only a more interpretable result, but the possibility of comparing anomaly rates across different data sets.

The development of this model is still in its early stages and many future work direction can be explored. This includes, but is not limited to:

- a more detailed description of the likelihood function by considering the nature of the querying process

- more careful modelling of the original prior given by the baseline unsupervised model
- more refined modelling of the ensemble prediction function
- consideration of intra region interactions by tracking abstract properties of non terminal nodes
- extension to other tree based tessellation algorithms to profile the data distribution.

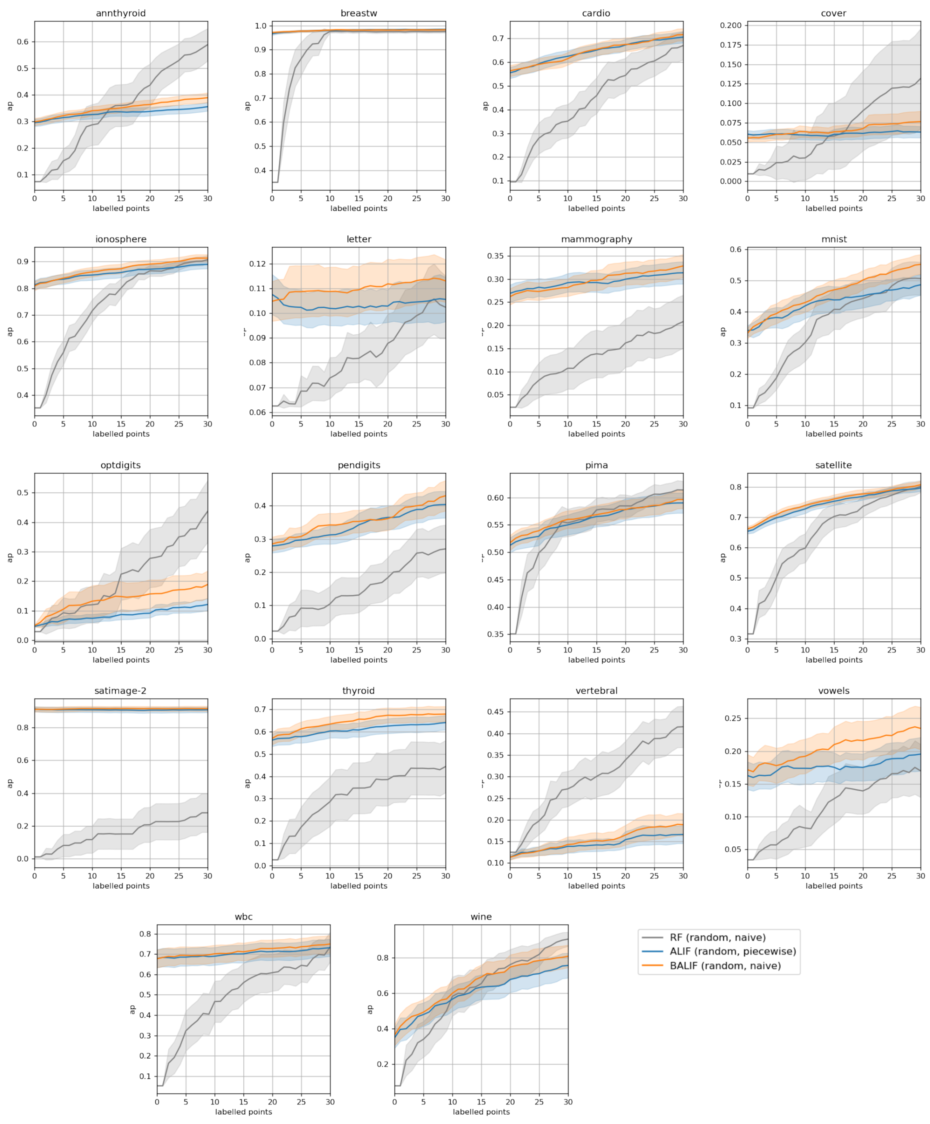


Figure 55: Question 1. Comparison between RF, ALIF and B-ALIF as weakly supervised models (no active query) on real world data sets. The figure shows how the model's Average Precision improves as a function of the number of labelled data and is compared to the one achieved with a random forest trained on the same amount of labelled data.

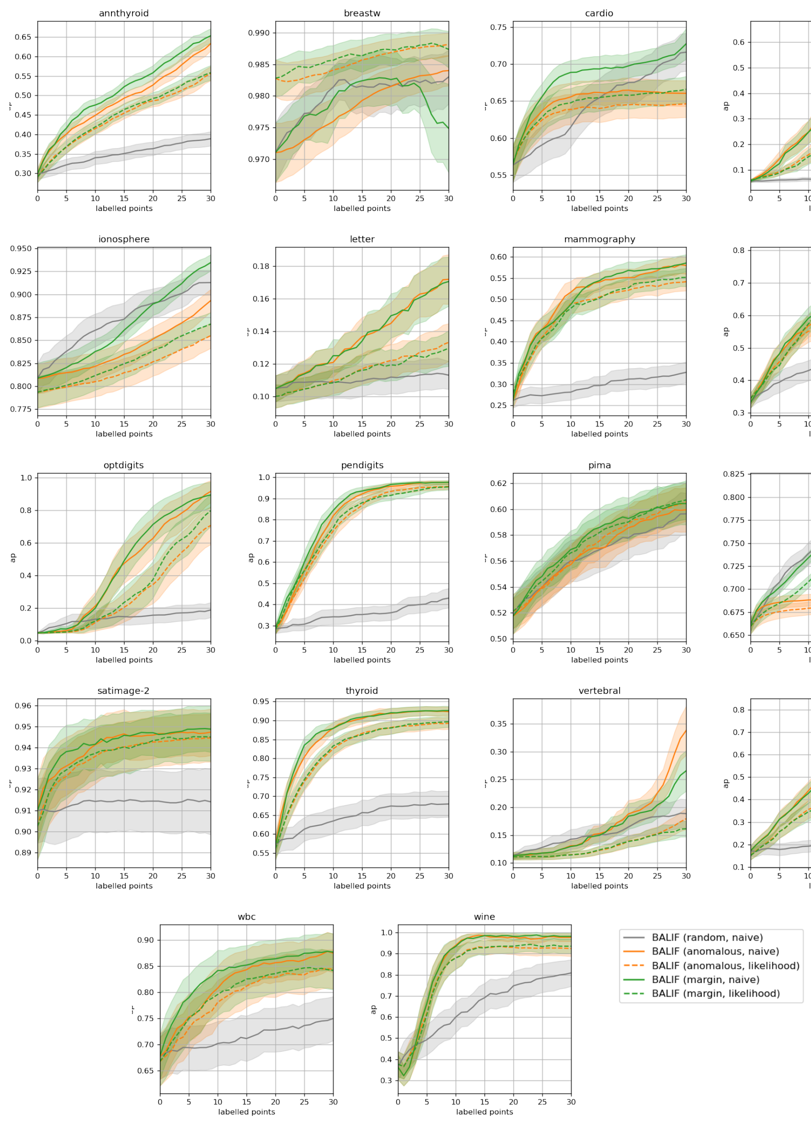


Figure 56: Question 2. Comparison between different variants of B-ALIF on real world data sets. The figure shows how the model's Average Precision improves as a function of the number of queries for all combination of ensemble predictions and querying strategies, compared with a weakly supervised variant (random query).

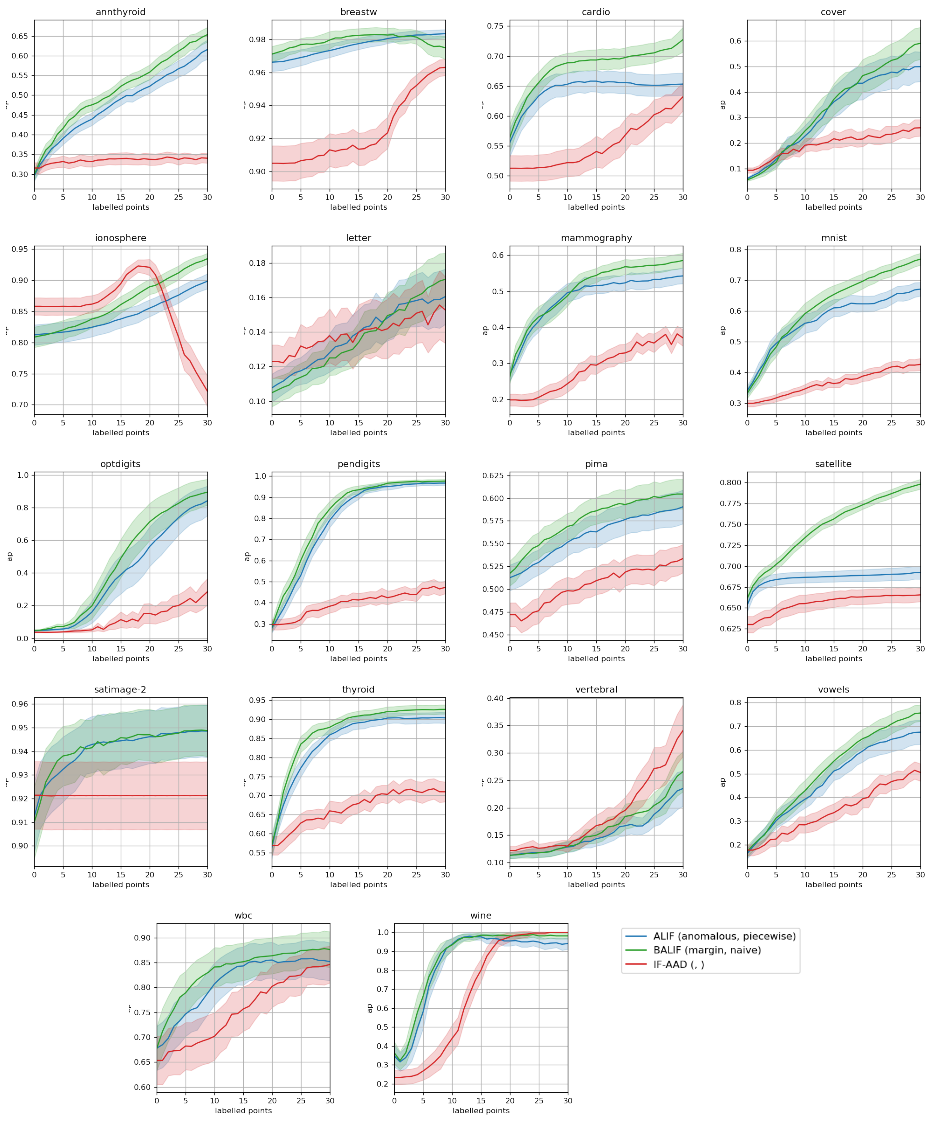


Figure 57: Question 3. comparison between different Active Learning algorithms. B-ALIF and ALIF generally outperform other methods while also introducing significantly less overhead.

# 9 EXPLAINABLE ANOMALY DETECTION

Merely identifying anomalous data is not enough in many real-works applications. Users of these systems may benefit from understanding the reasons behind the predictions, as this can facilitate root-cause analysis and increase trust in the model. Unfortunately, anomaly detection is an unsupervised task, so the development of interpretable tools quite challenging. Nevertheless, some algorithms were proposed to equip anomaly detection algorithms with explanations, including DIFFI [46], which is a model-specific approach for providing global and local feature rankings within the well-known Isolation Forest (IF). Despite its popularity, IF has some biases in the tree construction phase, which can reduce its detection performance. Thus, a new model called Extended Isolation Forest (EIF) was introduced, which uses a different isolation strategy and improves detection performance. In this Chapter, a new approach called ExIFFI is proposed to provide Extended Isolation Forest with interpretability traits, in the form of feature rankings. We test ExIFFI on both synthetic and real datasets and demonstrate its effectiveness in providing explanations. Feature selection is employed here as a proxy task to assess the effectiveness of the provided interpretability; we also show how ExIFFI can be employed as a valid feature selection approach in unsupervised settings.

## 9.1 INTRODUCTION

Machine Learning (ML) and Artificial Intelligence (AI) are playing a central role in the ongoing socio-economic change, revolutionizing various sectors such as manufacturing [142, 259], medicine [116], and the Internet of Things [137]. With the increasing deployment of ML across various industries, new problems have emerged due to the widespread application of these systems, which are often complex and opaque. Moreover, the end users of these systems are becoming more diverse, including people from various backgrounds who may not have a knowledge in data-driven methods. Therefore, it is essential to develop explanation algorithms that provide a deeper understanding of the model structure and predictions to ensure that these systems can be used effectively by a wide range of users.

Many scientific works identify explainability[1] as a key factor to enable the successful adoption of ML-based systems [54, 70, 157]. The relevance of explainability in ML-based technologies is paramount, as users are being asked to accept their integration into everyday decisions. This is particularly challenging in critical areas such as medical diagnosis, insurance, and financial

[1]While authors in the literature use the term 'interpretability' and 'explainability' associated with slightly different concepts when associated to Machine Learning/Artificial Intelligence as presented in [98], in this work we will use both terms interchangeably.

services. Given the significant influence of ML technologies, institutions are making considerable efforts to regulate them, typically by ensuring that interpretability traits are available. For instance, the European Union's General Data Protection Regulation (GDPR) [1] stipulates a right to obtain "*meaningful information about the logic involved*" for consumers affected by an automatic decision.

In the case of tabular data, there are various methods for explaining the outcomes of a model [185]. One common approach is to calculate the importance of each feature in the model's predictions. This involves evaluating the influence of each feature on both individual predictions (referred to as "local importance") and the overall dataset ("global importance"). By understanding the relative importance of each feature, users can gain insight into how the model is using the input data to inform its predictions.Feature importance has been widely exploited and popularized in recent years thanks to the Random Forest (RF) algorithm. RF provides built-in approaches to provide a ranking of the most important features [146]. Moreover, thanks to the rise of many post-hoc methods [222], like permutation importance [10] or SHAP [169], feature ranking as an interpretability measure has been extended also to models that are not tree-based.

Despite the remarkable recent advancements in eXplainable Artificial Intelligence (XAI), most approaches are designed for supervised tasks, leaving unsupervised tasks, like Anomaly Detection (AD), rarely discussed in the literature.

As discussed in Chapter 4 Isolation Forest [159] is one of the most popular AD approaches due to its high accuracy, low computational costs, and relatively simple inner mechanisms. This method relies on an iterative process. Recursively splitting the feature space along axis-aligned hyper-planes chosen at random, IF can isolate anomalous points using few space partitions. The first approach to provide explainability features for IF, named Depth-based Isolation Forest Feature Importance (DIFFI) [46] takes advantage of the inner structure of the IF to supply global and local model explanations.

Unfortunately, the one-dimensional partition process that IF relies on causes the creation of artifacts that degrade the detection of anomalies and negatively affect feature explanation. Thus, the Extended Isolation Forest (EIF) has been proposed [107]. EIF improves over the IF using oblique partitions that avoid creating the previously discussed artifacts.

In this Chapter we will contribute to the research area of interpretable AD, by providing, to the best of our knowledge, the first model-specific[2] approach, called *ExIFFI*, which can provide explanations about the Extended Isolation Forest. In designing ExIFFI, we drew inspiration from the ideas behind DIFFI. However, we introduced a more sophisticated procedure. While IF separates the input space by considering a variable at a time to define random splitting hyper-planes, EIF considers multiple variables, which leads to an *attribution*

[2]Model-specific interpretation tools are limited to specific model classes. Instead, Model-agnostic tools can be used on any machine learning model and are applied after the model has been trained [185]

*problem* when aiming at fairly distributing the contribution of the different variables to the decision made by the algorithm.

In the following sections, the effectiveness of ExIFFI will be demonstrated also by means of feature selection, problem that is employed here as a proxy task [185]. Feature selection in Anomaly Detection is a task poorly explored in the literature: while this will not be the focus of our work, we also show the potential of ExIFFI if used to guide feature selection.

The rest of the Chapter is organized as follows. First, we will recall the the EIF algorithm only briefly discussed in Chapter 4.3.1, in particular, the differences that allow EIF to overcome the structural problems that are hidden in the original IF implementation. Next, after introducing DIFFI in Section 9.3, the theory behind ExIFFI is provided in Section 9.4. Finally, ExIFFI is experimentally validated on 12 datasets, 4 synthetic and 8 real, by providing a comparison with DIFFI and Random Forest feature importance. In Section 9.5, the effectiveness of ExIFFI *global importance* is assessed using feature selection as a proxy task; moreover, also the *local importance* is analyzed by comparing the importance scoremaps of the ExIFFI algorithm compared to those of the DIFFI algorithm. Finally, in Section 9.6, conclusions are discussed.

## 9.2 EXTENDED ISOLATION FOREST

Although IF is one of the most popular and effective anomaly detection algorithms, it has some drawbacks due to its partition strategy. One of them is related to considering only hyper-planes that are orthogonal with respect to the directions of the feature space, as Hariri et al. show in [107]. In some cases, this bias leads to the formation of some artifacts where points are associated with low anomaly scores even if they are clearly anomalous. These areas are in the intersection of the hyperplanes orthogonal to the dimensions associated with the detection of inliers (Figure 58), creating misleading score maps and, in some cases, wrong predictions.

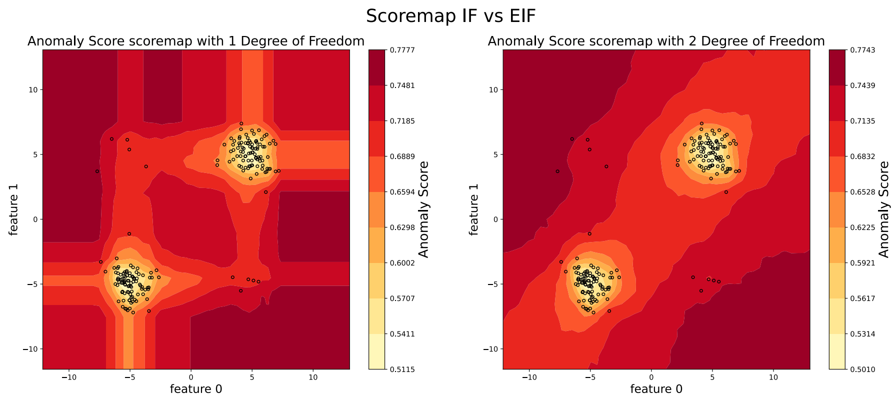


Figure 58: Scoremap differences between IF and EIF in the 2 dimensional space dataset *bimodal* described in Section 9.5.1. Figure inspired by [107].

As a consequence, Hariri et al. in [107] tried to improve the anomaly detection algorithm by correcting this bias with a different and more general algorithm called Extended IF. Instead of selecting a hyper-plane orthogonal to a single input dimension chosen at random, they suggested picking a random point $\mathbf{p}$ and a random vector $\mathbf{v}$ that are consequently used to build a hyperplane $\mathcal{H}$ that will split the space at each node of the binary tree into two smaller subspaces, as the IF does. Therefore, they proposed a more generic and extended paradigm, while maintaining the fast execution and low memory requirements of the original IF.

In the following, we will briefly describe the working principle of the EIF model. Let's consider a set $X$ of $n$ elements $\mathbf{x}_i \in \mathbb{R}^d$ with $i \in \{1, \dots, n\}$. The EIF model [107] generates a forest of $n_T$ full binary trees, in order to evaluate the anomaly score of the points.

$$F = \{T_0, T_1, \dots, T_t, \dots, T_{n_T}\}. \tag{39}$$

Every tree $T_t \in F$ is built from a bootstrap sample $X^t \subset X$.

At each node $k$ of the tree $T_t$ a random hyperplane $\mathcal{H}_k^t$ is selected by picking a random point $\mathbf{p}$ inside the distribution space limits, and $\mathbf{v}$ as a Normalized Standard Normal random vector:

$$\mathbf{v} \sim \frac{\boldsymbol{Z}}{\|\boldsymbol{Z}\|_2}, \tag{40}$$

$$\boldsymbol{Z} = (Z_1, \dots, Z_d)^\top, \tag{41}$$

$$Z_i \sim \mathcal{N}(0, 1) \ \forall\, i \in \{1, \dots, d\}. \tag{42}$$

At the root node $k_0$, the algorithm starts by generating a random hyperplane defined by a intercept point $\mathbf{p}_0$ and $\mathbf{v}_0$.

Thus the hyperplane splits the dataset $X_0$ into two subsets,

$$\begin{aligned} L_0 &= \{\boldsymbol{x} | \boldsymbol{x} \in X_0^t, \boldsymbol{v}_0 \cdot \boldsymbol{x} > \boldsymbol{v}_0 \cdot \mathbf{p}_0\}, \\ R_0 &= X_0^t \setminus L_0. \end{aligned} \tag{43}$$

The root node of the tree, $k_0$, is the space splitting made by the hyperplane $\mathcal{H}_0^t$. Then, this method is applied recursively to the subsets $L_0$ and $R_0$ until the max number of splits is reached, which corresponds to the preset max depth of the tree or when the set to split has only one element.
As a consequence every tree $T_t$ is composed of a set of nodes:

$$T_t = \{k_0, k_1, \dots, k_m\}, \tag{44}$$

and each node is a split of the space made by a particular hyperplane.

Thanks to this hierarchical tree structure, to evaluate if an element is an anomaly, the model extracts the path of the point $x \in X_t$ from the root to the leaf nodes down the tree. Then, as IF does [211], the EIF algorithm uses the average depth of the point in each tree to evaluate the anomaly score, according to the paradigm that the anomalies are isolated with few partitions. Indeed anomalous points are considered to be few with respect to inliers and far from the distribution of the rest of the dataset, therefore it is more probable that they will be isolated faster than the inliers. Hence, on average, they will have shorter paths than other points in the tree as introduced in Section 4.3.1.

## 9.3 DEPTH-BASED ISOLATION FOREST FEATURE IMPORTANCE

Depth-based Isolation Forest Feature Importance (DIFFI) is the first unsupervised model-specific method addressing the need for interpretability of the IF detector [46]. It exploits the structure of the trees in the IF algorithm to understand which features are the most relevant to discriminate whether a point is an outlier or not. It is based on two intuitions concerning the usefulness of a split node:

- the feature along which it splits the data should isolate as fast as possible the anomalies;
- this feature should create a high imbalance when isolating anomalous points, the opposite for inliers.

With this in mind, the present Section is going to describe which tools DIFFI uses to measure these intuitions. In particular, DIFFI defines:

INDUCED IMBALANCE COEFFICIENTS $\lambda$ : given an internal node $k$ of an isolation tree, let $n(k)$ be the number of points that the node divides, $n_l(k)$ and $n_r(k)$ the number of points on the left child and right child, respectively. The coefficient measuring the induced imbalance of the node $k$ is:

$$\lambda(k) = \begin{cases} 0 & \text{if } n_l(k) = 0 \text{ or } n_r(k) = 0 \\ \hat{\lambda}(k) & \text{otherwise} \end{cases} \tag{45}$$

with:

$$\hat{\lambda}(k) = g\left(\frac{\max(n_l(k), n_r(k))}{n(k)}\right) \tag{46}$$

$$g(a) = \frac{a - \lambda_{min}(n)}{2(\lambda_{max}(n) - \lambda_{min}(n))} + 0.5 \tag{47}$$

where $\lambda_{min}$ and $\lambda_{max}$ denote the minimum and maximum scores that can be obtained a priori given the number of data $n(k)$. This coefficient measures the imbalance induced by each node in the tree, i.e. the first of the two main intuitions.

CUMULATIVE FEATURE IMPORTANCES $\boldsymbol{I}$ : it is a vector of dimension $d$ where the $j$-th component is the feature importance of the $j$-th feature. It is distinguished between the one of created using predicted inlier points, denoted $\boldsymbol{I}_I$ and the one belonging to outliers $\boldsymbol{I}_O$. Concerning $\boldsymbol{I}_I$, the procedure starts with the initialization $\boldsymbol{I}_I = \mathbf{0}_d$ and considering the subset of predicted inliers for the tree $t$, $\mathcal{P}_{I,t}$. Then, for the generic predicted inlier $x_I \in \mathcal{P}_{I,t}$, DIFFI iterates over the internal nodes in its path $Path(x_I, t)$. If the splitting feature associated with the generic internal node $k$ is $f_j$, then we update the $j$-th component of $\boldsymbol{I}_I$ by adding the quantity:

$$\Delta = \frac{1}{h_t(x_I)} \lambda_I(k) \tag{48}$$

The same procedure applies for $x_O \in \mathcal{P}_{O,t}$ and $\boldsymbol{I}_O$. Intuitively at each point and each step, the vector accumulates the imbalance produced by each feature. The imbalance is measured by the previously defined $\lambda$ and weighted by the depth of the node, meaning that features that allowed to isolate faster the points, are considered to be more useful.

FEATURES COUNTER $\boldsymbol{V}$ : it is used to compensate for uneven random feature sampling that might bias the calculated cumulative feature importance. At each passage of a point through a node, the entry of the counter corresponding to the splitting feature is incremented by one. As for the cumulative feature importance, two features counter are calculated, the one for predicted inliers $\boldsymbol{V}_I$ and the one for outliers $\boldsymbol{V}_O$.

Finally, the Global Feature Importance is obtained looking at the weighted ratio between outliers and inliers cumulative feature importance:

$$GFI = \frac{\boldsymbol{I}_O / \boldsymbol{V}_O}{\boldsymbol{I}_I / \boldsymbol{V}_I}$$

## 9.4 EXTENDED ISOLATION FOREST FEATURE IMPORTANCE

As DIFFI does with the IF (Section 9.3), with the Extended Isolation Forest Feature Importance (ExIFFI) we will try to rank and evaluate the importance of the features in deciding whether a sample is an anomaly or not for the EIF model.

As seen in Section 9.2 a tree is built in a way that an hyperplane $\mathcal{H}_k^t$, that splits the subsample $X_k^t \subseteq X^t \subseteq X$, is associated to each node $k \in T_t$ of the tree $T_t \in F$. The hyperplane is well defined by a vector orthogonal to its direction and a point that belongs to it $\{\boldsymbol{v}_k^t, \mathbf{p}_k^t\}$.

The hyperplane $\mathcal{H}_k^t$ separates the points in a set of elements on the left side of the hyperplane and a set of elements on the right side of the hyperplane.

$$\begin{aligned} L_k^t &= \{\mathbf{x} | \mathbf{x} \in X_k^t, \mathbf{v}_k^t \cdot \mathbf{x} > \mathbf{v}_k^t \cdot \mathbf{p}_k^t\}, \\ R_k^t &= X_k^t \setminus L_k^t. \end{aligned} \tag{49}$$

We want to know which features influenced the most the isolation procedure for a sample $x$. ExIFFI assigns a vector of feature importances to each node of the tree, based on two paradigms:

- The importance of the node for a point $x$ is higher when it creates a greater inequality between the number of elements on each side of the hyperplane and $x$ is in the smaller subset. Indeed, greater inequality means a higher grade of isolation for the points in the smaller set.

- In each node of the model, the relative importance of each feature is determined by the projection of the normal vector of the splitting hyperplane onto that feature. If the split occurs along a single feature, that feature will receive the importance score. If the splitting hyperplane is oblique, the importance scores of multiple features will be calculated based on their respective projections onto the normal vector of the hyperplane.

Then we assign an importance value function to every node of the trees that is the projection on the normal vector of the splitting hyperplane of the quotient between the cardinality of the sample before a particular node and after it, following the path of a sample $x$. Thus, knowing that the splitting hyperplane $\mathcal{H}_k^t$ of that node is determined by the pair $\{\mathbf{v}_k^t, \mathbf{p}_k^t\}$:

$$\boldsymbol{\lambda}_k^t(\mathbf{x}) = \begin{cases} \left(\frac{|X_k^t|}{|L_k^t|}\right) |\mathbf{v}_k^t|, & \text{if } \mathbf{v}_k^t \cdot \mathbf{x} > \mathbf{v}_k^t \cdot \mathbf{p}_k^t \\ \left(\frac{|X_k^t|}{|R_k^t|}\right) |\mathbf{v}_k^t|, & \text{otherwise} \end{cases} \tag{50}$$

The vector of importances evaluated in the tree $T_t$ for an point $x$ is the sum of all the importance vectors of all the nodes that the element $x$ passed through on its path to the leaf node in the $t$-th tree defined in Equation (**??**):

$$\boldsymbol{I}_t(x) = \sum_{k \in \mathcal{P}_x^t} \boldsymbol{\lambda}_{t,k}(x). \tag{51}$$

We then calculate the sum of the importance vector of the point $x$ for all the trees in $\mathcal{T}$:

$$\boldsymbol{I}(x) = \sum_{t \in T} \boldsymbol{I}_t(x) \tag{52}$$

$\boldsymbol{V}(x)$ is the sum of the vectors orthogonal to the hyperplanes of the nodes that an element $x$ passes in a tree, then we calculate the sum of the values in all the trees:

$$\boldsymbol{V}(x) = \sum_{t \in T} \sum_{k \in \mathcal{P}_x^t} \boldsymbol{v}_k^t \tag{53}$$

## 9.4.1 *Global and Local interpretation*

On one hand, the results of the EIF can be interpreted globally, meaning that they apply to the entire model. This type of interpretation provides a ranking of the features in terms of their contribution to the outliers. On the other hand, interpretability can also be local, meaning that it focuses on understanding which features caused a specific sample to deviate from the distribution of the dataset.

### *Global Feature Importance*

To globally evaluate the importance of the features, as DIFFI does with the IF interpretation, we divide $X$ into the subset of predicted inliers $Q_I = \{\mathbf{x}_i \in$

$X|\hat{y}_i = 0\}$ and the one of predicted outliers $Q_O = \{\mathbf{x}_i \in X|\hat{y}_i = 1\}$ where $\hat{y}_i \in \{0, 1\}$ is the binary label produced by the thresholding operation indicating whether the corresponding data point $\mathbf{x}_i$ is anomalous ($\hat{y}_i = 1$) or not ($\hat{y}_i = 0$).

We define the global importance vectors for the inliers and for the outliers by summing out the importance values introduced in Equation (52):

$$\begin{aligned} \boldsymbol{I}_I &= \sum_{\mathbf{x}\in Q_I} \boldsymbol{I}(\mathbf{x}), \\ \boldsymbol{I}_O &= \sum_{\mathbf{x}\in Q_O} \boldsymbol{I}(\mathbf{x}). \end{aligned} \tag{54}$$

Likewise the sum of the orthogonal vectors:

$$\begin{aligned} \boldsymbol{V}_I &= \sum_{\mathbf{x}\in Q_I} \boldsymbol{V}(\mathbf{x}), \\ \boldsymbol{V}_O &= \sum_{\mathbf{x}\in Q_O} \boldsymbol{V}(\mathbf{x}). \end{aligned} \tag{55}$$

Due to stochastic sampling of hyperplanes, in order to avoid the bias generated by the fact that it is possible that some dimensions are sampled more often than others, the vectors of importance have to be divided by the sum of the orthogonal vectors.

$$\begin{aligned} \hat{\boldsymbol{I}}_I &= \frac{\boldsymbol{I}_I}{\boldsymbol{V}_I}, \\ \hat{\boldsymbol{I}}_O &= \frac{\boldsymbol{I}_O}{\boldsymbol{V}_O}. \end{aligned} \tag{56}$$

To evaluate which are the most important features to discriminate a data as an outlier we divide the importance vector unbiased of the inliers by the one of the outliers Equation (56), and we obtain the global feature importance vector as the DIFFI algorithm [46]:

$$\boldsymbol{GFI} = \frac{\hat{\boldsymbol{I}}_I}{\hat{\boldsymbol{I}}_O}. \tag{57}$$

*Local Feature Importance*

Let's take into account an element $x$, the Equation (52) gives a vector of importances $\boldsymbol{I}(x)$ of the sample $x$ for each feature. Then the vector $\boldsymbol{V}(x)$ is the normalization factor of the feature importance. The Local Feature Importance ($\boldsymbol{LFI}$) of an element $x$ is the quotient:

$$\boldsymbol{LFI}(x) = \frac{\boldsymbol{I}(x)}{\boldsymbol{V}(x)}. \tag{58}$$

*Non Depth-Based importance*

DIFFI and ExIFFI differ in a key concept, with DIFFI being depth-based. The DIFFI algorithm assigns importance scores to each node based on the inverse of the depth of the node. As emerges from in Equation (48) DIFFI weights

$\lambda_I(k)$ by the inverse of the node depth $\frac{1}{h_t(x_I)}$, as it is assumed that the deeper the cut is in the isolation tree, the less important it is in determining if a sample is an outlier or not. In contrast, as shown in Equation (50), ExIFFI does not take depth into account when computing importance scores. This is for two main reasons:

1. if there is a cut at the beginning of the tree that is important in terms of isolation, it can generate a larger quotient because it has to split a larger dataset. However, if the cut does not create any isolation, there is no reason to prioritize it over other cuts;

2. given a sample, the importance of the feature that determines if it is an outlier or not is the summation of all the importance vectors of the path followed by the sample $x$ from the root node to the leaf as shown in Equation (51). On one hand, if a sample is an outlier, it has fewer nodes in its path, so the summation is done using only the initial hyperplanes. These hyperplanes are essential as they separate the outlier from the rest. The most significant features of the outlier are highly correlated to these hyperplanes. On the other hand, if a sample is an inlier, it has to pass through many nodes, so the importance vector is a summation of many elements that characterize the sample less.

Modifying the contributions of the deeper nodes in a decision tree by using the inverse of the depth or does not significantly alter the resulting feature importance scores. It is possible to observe that the ranking of the features remains similar in the histograms of the importance scores in Figures 59a and 59b. Because the results obtained using different methods for weighting the contributions of the deeper nodes in a decision tree are similar, it is more efficient to use a simpler algorithm that does not require the addition of a depth-based parameter. Therefore, we decided that the ExIFFI method does not incorporate depth as a factor in its calculations.

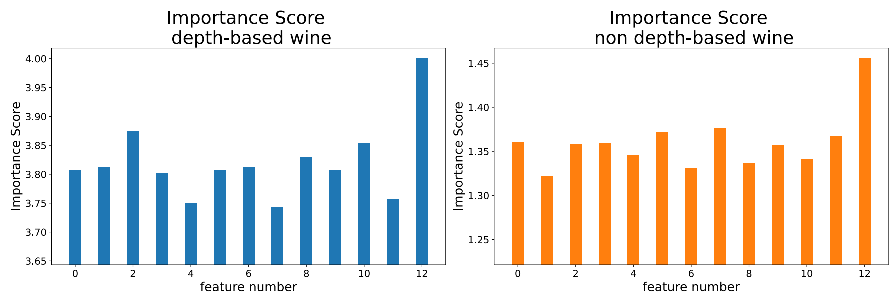
## 9.5 EVALUATION OF INTERPRETATION ALGORITHMS

### 9.5.1 *Datasets*

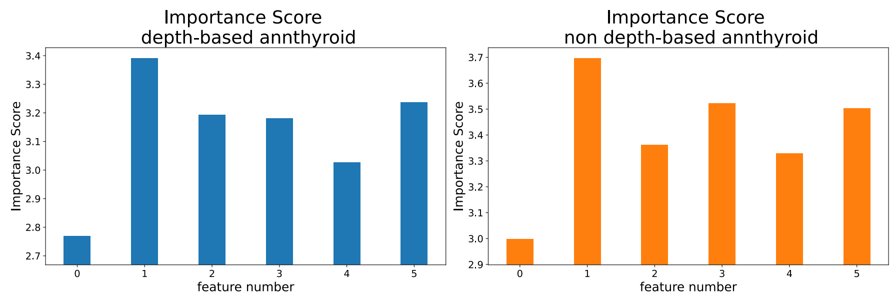
To evaluate the interpretation of the algorithms twelve datasets with labelled anomalies will be used as a benchmark. These datasets consist of four synthetic datasets, which were created to investigate the differences of the models, and eight open source datasets based on real applications. In the rest of the Chapter the datasets will be indicated using an *italic* font.

#### *Synthetic datasets*

The synthetic datasets are designed to highlight the limits and strengths of the two models. The first dataset, henceforth called *bimodal*, is a very simple two-dimensional dataset that is composed of two Gaussian distributions around the positions $(1, 1)$ and $(-1, -1)$. Moreover, it has some anomalies in the other



(a) Histogram with the mean importance score for each feature in the case of a depth-based algorithm and non depth-based



(b) Distances

point of intersection of the orthogonal axis defined by the equations $x = 1$ and $y = -1$. It has the perfect characteristics to show the structural consequences of the bias generated by the IF model. The other three synthetic datasets come from the paper by Carletti et al. [46]. Inliers are generated according to a zero-mean six-dimensional Gaussian distribution. The three datasets differ in the distribution of the anomalies: they are distributed along the first dimension (feature 0) in one dataset named *synt xaxis*, along the second dimension (feature 1) in another dataset named *synt yaxis* and along the bisection line of the first two dimensions in the third dataset named *synt bisec*.

## *Real datasets*

The model was also tested on eigth widely used real benchmark datasets for anomaly detection listed in Table 14 and openly available at [204]. The datasets are very diverse in terms of domains since they come from different fields like medicine, industry, and natural sciences.

Their size is very heterogeneous: the *wine* dataset has just 129 elements, while the largest is *shuttle* with 49097 elements. To ease the burden of the experiments, we undersample large datasets in a way that preserves the original proportion of inliers and outliers, but limits the sample size to 2500 elements at most. As shown in Table 14, we can divide the datasets into three levels of dimensionality: (i) Low (less than 7 features), (ii) Medium (between 7 and 20 features) and (iii) High (more than 20 features). There are even three types of contaminations, in *pima*, *ionosphere* and *breastw* more than 30% of the dataset are outliers so they have very high contamination rate. On the contrary, *mammography* and *pendigits* have approximately only 2% of contamination rate. The other datasets have roughly the 8% of outliers. It is important to outline how these datasets were built: starting from multi-class classification datasets, a class was under-sampled and labeled as an outlier, usually the one that *a priori* is known to be more structurally different than the others. As a consequence, the anomalies are not just uniformly distributed random points far from the normal distribution but are a particular class coherent with the other anomalies and sometimes even with the dataset.

| | n. data | n. anomalies | contamination | n. features | (Dimensionality) |
|---|---|---|---|---|---|
| | $n$ | | % | $d$ | |
| *annthyroid* | 7200 | 534 | 7.42 | 6 | (Low) |
| *breastw* | 683 | 239 | 35 | 9 | (Middle) |
| *cardio* | 1831 | 176 | 9.6 | 21 | (High) |
| *ionosphere* | 351 | 126 | 36 | 33 | (High) |
| *pendigits* | 6870 | 156 | 2.27 | 16 | (Middle) |
| *pima* | 768 | 268 | 35 | 8 | (Middle) |
| *shuttle* | 49097 | 3511 | 7.15 | 9 | (Middle) |
| *wine* | 129 | 10 | 13 | 7.7 | (Middle) |

Table 14: Set of data used in the experimental phase. The first column gives the name of the dataset; the second column describes the number of instances contained in each set; the third column gives the number of outliers; the fourth column presents the contamination rates; the fifth column defines the total amount of features; the last shows the dataset dimensionality.

One significant challenge in the evaluation of ML interpretability approaches is that it is difficult to determine the quality of an explanation. Indeed, there is no objective metric for comparing the performance of different algorithms in terms of their ability to provide good explanations. Since the popular book written by Molnar [185], many papers tried to set up an objective theory on what is an explanation and how to evaluate it [54, 70, 157, 217, 275]. To evaluate the XAI algorithm, we first assess its effectiveness on synthetic benchmark datasets. Such datasets are designed to analyze if the algorithm is able to detect the dimensions along which the anomalies are distributed, both with Local and Global Feature Importance scores. We will then make experiments on "real datasets" where the distributions are not defined ad-hoc for our purposes.

As discussed in [185], assessing the effectiveness of XAI approaches in real world problems is not straightforward. One way to provide an evaluation is through a 'proxy task', i.e. a related problem which can be instead easily assessed in a quantitative way: we then exploit the evaluation on the proxy task as a surrogate for the evaluation of the original task, the XAI problem. As done in [46] we will exploit feature selection as a proxy task: we will use the indication of ExIFFI and competitor methods for guiding the feature selection, i.e. by keeping the $i$ features (for different values of $i$) that are globally considered as the most important by the XAI approach. While feature selection in anomaly detection is not the focus of this work, we will implicitly show the effectiveness of exploiting ExIFFI also in this task.

Moreover, we will resort to an additional test for assessing the quality of the ExIFFI interpretation, which is a comparison with the interpretation provided by Random Forest (RF). RF is a popular supervised algorithm, which is also an ensemble approach based on trees, with the main difference that in RF we are resorting to decision trees, not isolation trees as in the IF/EIF. RF is equipped with a procedure for providing feature rankings, that is widely used by the scientific community and by practitioners: we will compare the feature rankings provided by RF, when considering the AD problem as a binary classification one (with inliers and outliers being the two classes), with the ones provided by ExIFFI. While the rankings provided by RF are not to be considered a ground truth and RF may not be effective with unbalanced classification problems, RF has the advantage of exploiting labels for both solving the classification and the XAI task: in light of this, we consider such test as a relevant one for the purpose of evaluation ExIFFI capabilities.

### 9.5.2 *Synthetic datasets*

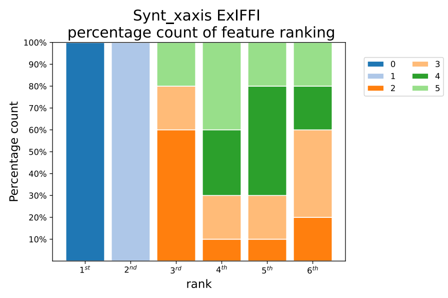
#### *Global Feature Importance*

In order to analyze the effectiveness of the global feature importance, we decided to use the synthetic dataset setup shown in the DIFFI paper [46] introduced in the Section 9.5.1. In this way we will have a previously validated proxy to show how much and to what extent ExIFFI works. The three examples have a central ball with inliers and some anomalies distributed along specific directions: the first example has the anomalies along feature 0, the second example along feature 1, and the third along a combination of the previous directions.

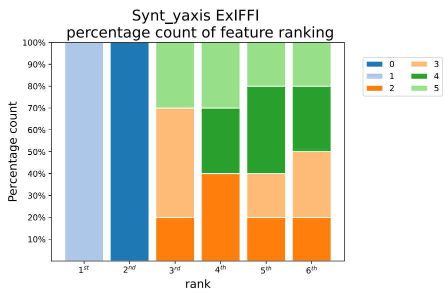
Figure 59 shows the global feature importance of ExIFFI in the case of the first and the second synthetic example. We can observe in Figure 59a that the model detects the correct feature consistently, this means that it detects with strong belief that feature 0 is the most important one to determine an outlier. In the case of Figure 59b, the feature 1 is correctly identified as the most important one.

Finally, as Carletti et al. did in [46], we want to analyse what happens when the anomalous points are distributed along the bisector line of the feature 0 and feature 1 as shown in Figure 60.

Since the EIF model is stochastic and both of the first two features are important, we expect that computing the model multiple times the feature ranking changes: sometimes the first one is feature 0, and sometimes feature 1, meaning that both of them are important in Figure 61a. The histogram depicted in the Figure 61b illustrates the mean importance scores for each feature in the case of the three 6-dimensional synthetic datasets. In the *synt xaxis* and *synt yaxis* datasets, the importance scores for feature 0 and feature 1 are not equal, with one score being greater than the other. In contrast, in the *synt bisec* dataset,



(a) Importance barplot of the dataset of anomalies distributed along the feature 0



(b) Importance barplot of the dataset of anomalies distributed along the feature 1

Figure 59: Feature importance obtained on datasets *synt xaxis* and *synt yaxis* explained in Section 9.5.1 where anomalies lie respectively on feature 0 and feature 1.

both feature 0 and feature 1 have very high and nearly identical importance scores.

### *Local Feature Importance*

It is crucial to know which features are most important for identifying whether a single sample is an outlier. For example, in an industrial process where an AD model detects an outlier, understanding the most relevant features can facilitate root cause analysis and allow for fast and effective corrective actions.

In [107], the authors demonstrated that Extended IF is able to address the bias limitations of IF through the use of a comparative approach. They presented the anomaly score maps of both algorithms, highlighting the ability of EIF to more accurately generalize the score map.

Similarly to that analysis, here we present a scoremap of the Local feature importance score originated from the DIFFI and ExIFFI algorithms, where the corresponding IF and EIF models were trained on a dataset named *bimodal* introduced in Section 9.5.1, mainly made up of points distributed into two separate balls along the bisector line. Then the scoremap is made by taking in consideration the whole space around the distribution, demonstrating how the ExIFFI is able to generalize better the importance scores related to the anomalies.

In Figure 62, the scoremaps display the relative importance of two features using red and blue colors. Red regions indicate higher importance of feature 1, while blue regions indicate higher importance of feature 0. The intensity of the color reflects the magnitude of the feature importance at each point. Lighter colors correspond to lower importance values, and thus points in these regions are less likely to be considered outliers with respect to the features under analysis. Additionally, the score map includes contour plots, represented by grey lines surrounding the data points. These contour plots depict different levels of the combined importance of the features.

Similar to IF, the score map of the DIFFI algorithm exhibits a strong bias along the direction orthogonal to the features. This bias can negatively impact

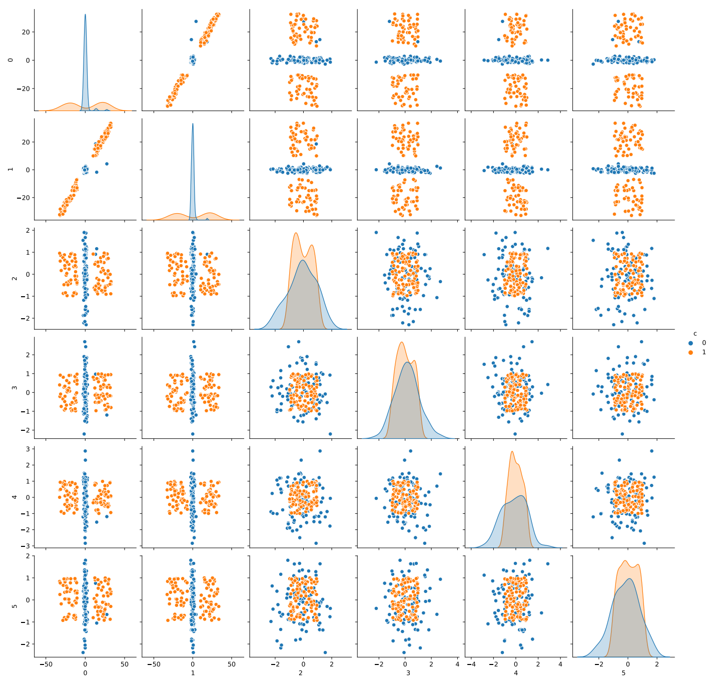
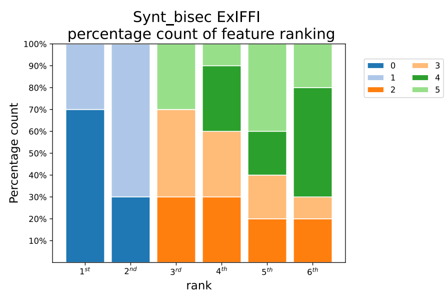


Figure 60: Synthetic dataset *synt bisec* with anomalies in orange distributed along the bisection line of the feature 0 and 1. Training data are depicted in blue, while testing data are depicted in orange.

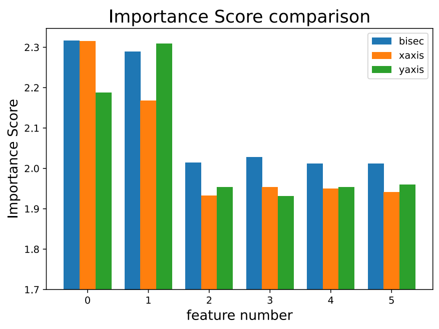
the model's comprehension of the feature, as it limits the ability of the DIFFI model to generalize well beyond the region of the feature space where the original distribution is concentrated. As a result, the model may struggle to accurately identify the features that have the greatest influence on determining whether a sample is an outlier in various regions of the feature space.

In a real-world scenario, it is possible that the interpretation algorithm is required to provide accurate interpretations for points that are located far from the distribution of the training dataset. If the DIFFI algorithm exhibits this bias, evidenced with the scoremap, it may struggle to accurately interpret such points, potentially leading to the provision of incorrect information.

To illustrate this point let's consider the scoremap presented in the image on the left of Figure 62. The bundles crossing the dataset orthogonally with respect to the principal directions exhibit lighter colors, indicating that data samples located in these regions are less anomalous compared to darker regions. However, it is not clear whether points located within the bundles but farther from the dataset are necessarily less anomalous than points that are at a similar distance but not within the bundles. Additionally, the top-left and bottom-right corners of the image appear to have a plain dark color, without distinguishing the region by intensity or feature colour.



(a) Importance barplot of the dataset of anomalies distributed along the feature 1



(b) Importance barplot of the dataset of anomalies distributed along the feature 1

Figure 61: Feature importance obtained on datasets *synt bisec* explained in Section 9.5.1 where anomalies lie on feature 0 and feature 1.

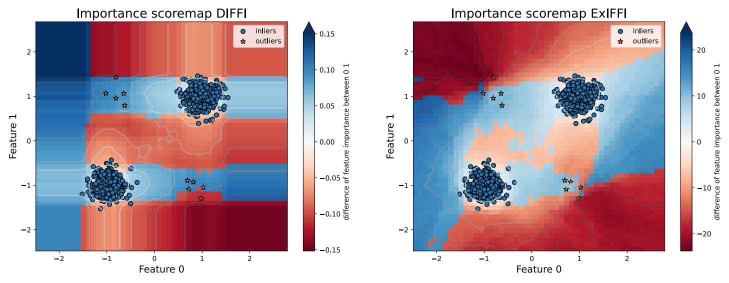


Figure 62: Feature importance scoremap of DIFFI and ExIFFI along the first two features of dataset *bimodal*.

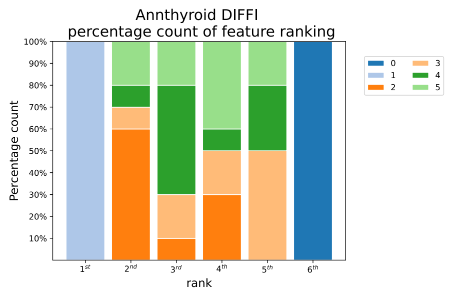
### 9.5.3 *Real data examples*

We used 8 widely used real datasets benchmark for anomaly detection introduced in Section 9.5.1 and listed in the Table 14. Obtaining a gold standard reference for evaluating explanations in the context of explainability can be challenging, as it often requires either a large group of people to assess the quality of a set of explanations or a domain expert to provide expert explanations of anomalies. As an alternative, we use feature importances provided by a supervised method, specifically Random Forest (RF), as a proxy measure of the performance of our unsupervised approach. The use of RF feature importances is motivated by the assumption that they provide good explanations because it is a supervised algorithm and the similarity between RF and EIF, both based on a decision tree structure. We compare the ranking of features produced by our unsupervised approach with those produced by RF to understand how much ExIFFI is close to RF in terms of feature selection. This allows us to assess the efficacy of our unsupervised approach.

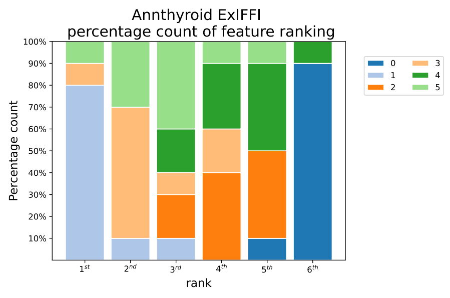
The feature selection experiments have been repeated 10 times: the detection model has been trained on the full set of features, and the least important feature has been removed at each step, retraining the model and recording the detection performances. At the end of the procedure, the model is allowed to observe only the most important feature. The same procedure has been computed using the feature importance provided by different approaches: the random approach (baseline), the DIFFI (old approach), the ExIFFI (proposed approach) and the RF one that is the only allowed to see the labels of the points.

The first real dataset where ExIFFI was tested is *annthyroid*, a medical dataset related to patients with thyroid diseases. The results of the repeated feature importance for DIFFI and ExIFFI shown respectively in Figures 63a and 63b: in such panels statistics concerning the most important feature are reported; it can be seen that such statistics are in accordance with the ones obtained with the RF shown in Figure 63c; however, it can be seen how DIFFI tends to give a high rank to feature 2 and 4 that are the least important according to RF. In Figure 63d it is interesting to observe the behaviour of the feature selection provided by DIFFI and ExIFFI on *annthyroid*: removing the unnecessary features, the detection model is able to perform much better than with the full set of features. On the contrary, removing a random feature (red line), the model is not able to improve.

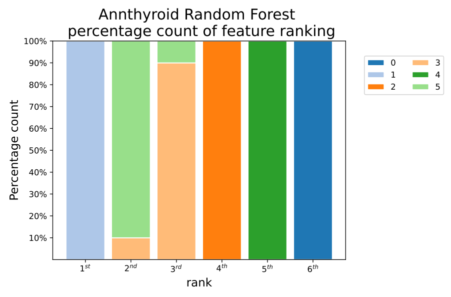
Looking at the dataset distribution in Figure 71a it is possible to observe that the outliers, shown in the shape of a star, are easily detected using the feature 1 alone.



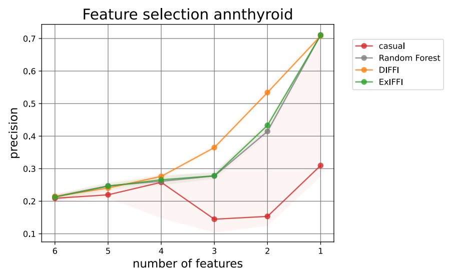
(a) DIFFI



(b) ExIFFI



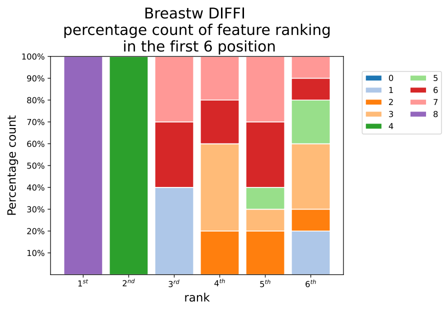
(c) RF



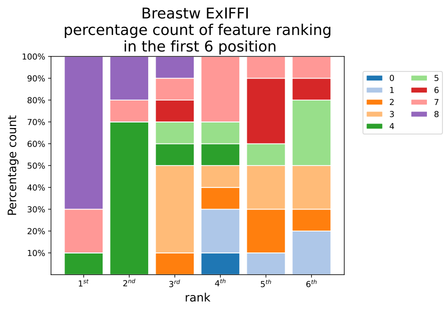
(d) Feature selection

Figure 63: Consistency of the feature importance and feature selection applied to the *annthyroid* dataset.

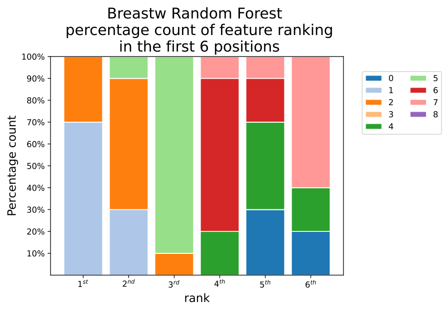
In the case of *breastw* dataset while both DIFFI and ExIFFI agree on which are the most important features to detect the anomalies, Random Forest feature importance have completely different outcomes. By analyzing Figure 64d, we can observe that RF has lower performances compared to the unsupervised feature selection algorithms, while ExIFFI has consistently the highest performances. This suggests the possibility to perform unsupervised feature selection.



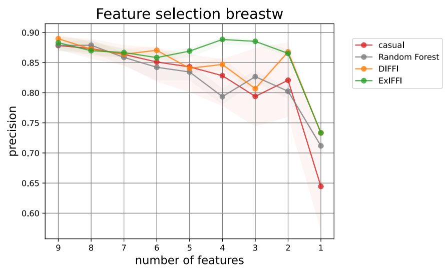
(a) DIFFI



(b) ExIFFI



(c) RF



(d) Feature Selection

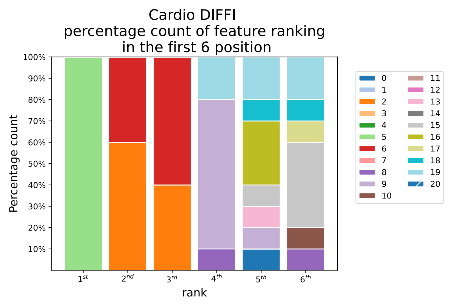
Figure 64: Consistency of the feature importance and feature selection applied to the *breastw* dataset

In the case of the dataset named *cardio* we can observe that DIFFI and ExIFFI algorithm agree that the feature 6 and the feature 2 play a central role to detect anomalies, moreover the DIFFI observes the feature 5 as the most important one very consistently. The only feature that the unsupervised feature ranking algorithm have in common with Random Forest feature importance is the feature number 6.

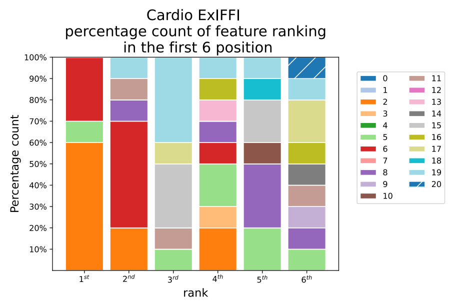
What it is possible to infer from the feature selection in Figure 65d is that the best feature selection is the one based on the ranking given by RF. In this case DIFFI performs better than ExIFFI.

*ionosphere* is a very complex dataset due to its high dimensionality (33 features), and this complexity reflects in the different behaviour of the models. We can observe in Figure 66a that the feature 0 is consistently selected by DIFFI as the most important feature while in Figure 66b ExIFFI does not show to have any preferences on the most important feature. RF on the contrary highlights the features 2,3 and 25. What we can observe looking to the feature selection graph is that DIFFI is only slightly better than the other algorithms. This fact underlines that as far as the *ionosphere* dataset is concerned, there are not real preferences among the features.

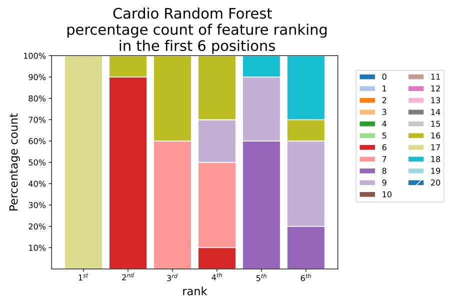
*pima* is also a medical dataset showing diabetic patients. Here it seems DIFFI and ExIFFI capture different aspects of the problem, indeed in Figure



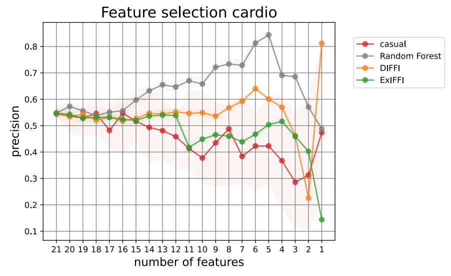
(a) DIFFI



(b) ExIFFI

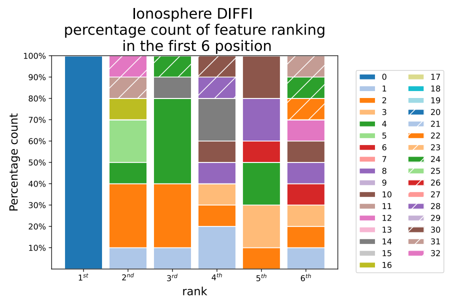


(c) RF

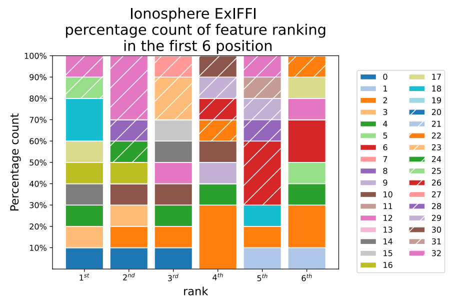


(d) Feature Selection

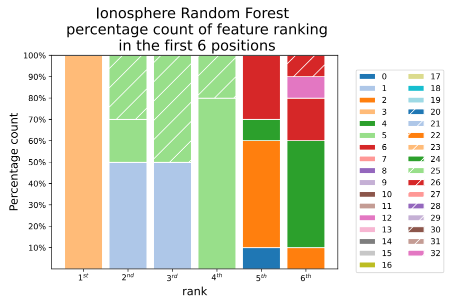
Figure 65: Consistency of the feature importance and feature selection applied to the *cardio* dataset



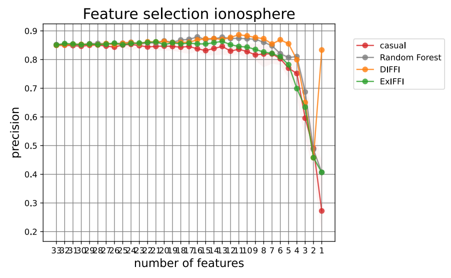
(a) DIFFI



(b) ExIFFI



(c) RF

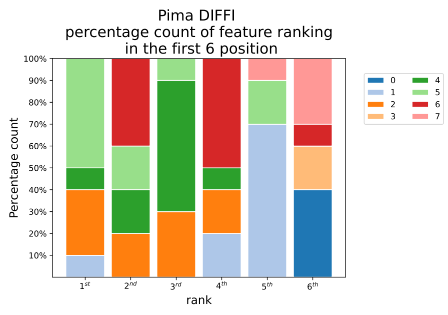


(d) Feature selection

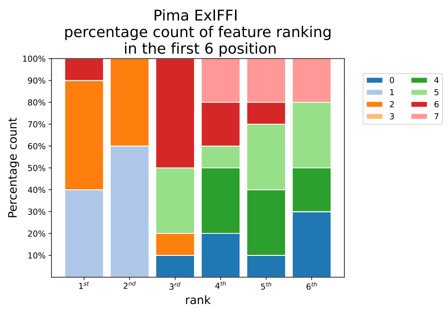
Figure 66: Consistency of the feature importance and feature selection applied to the *ionosphere* dataset

67 DIFFI is more consistent in highlighting feature 2 and 5 while ExIFFI 1 and 2. However RF consistently prefers feature 1 and feature 5 while discards common feature 2.

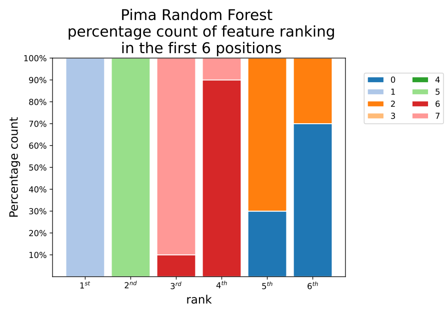
That said, Figure 67d shows that with many features DIFFI and ExIFFI perform quite the same, but selecting the three most important ExIFFI starts to perform much better with respect to DIFFI, with performances similar to RF. However in this dataset it is possible to observe that for some combinations of the features, the random feature selection can achieve even better results. This indicates that RF is a good reference, but might not be the best possible option. The last dataset considered is called *pendigit*, in this case we can



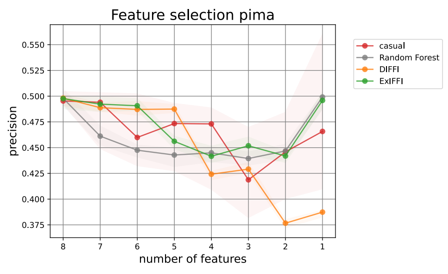
(a) DIFFI



(b) ExIFFI



(c) RF



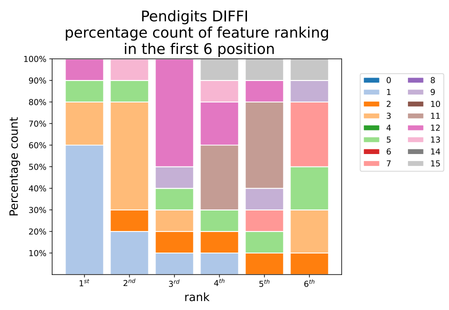
(d) Feature selection

Figure 67: Consistency of the feature importance and feature selection applied to the *pima* dataset

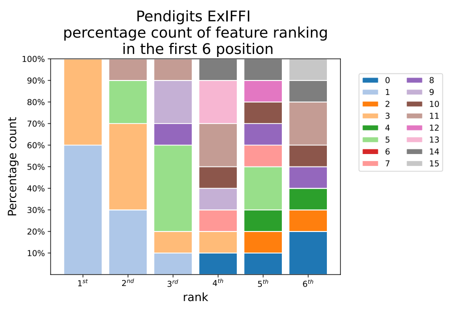
observe that, even if DIFFI and ExIFFI consistently agree on the two most important features, RF strongly disagrees sharing with them only feature 5. ExIFFI initially follows the performances of RF, while DIFFI has the same performances of the casual feature selection.

*shuttle* is shown in Figure 69. It is interesting to note DIFFI consistently ranks feature 3 in the first position while RF never suggests it as a relevant feature. On the contrary, ExIFFI gives results more similar to RF, highlighting feature 8 and feature 0. Similarly to *pima* feature selection in Figure 69d we can observe that DIFFI algorithm have a drop down in precision. Unfortunately, while RF achieves the best possible results and despite the good explanation results, ExIFFI performs on average as the random feature selection.

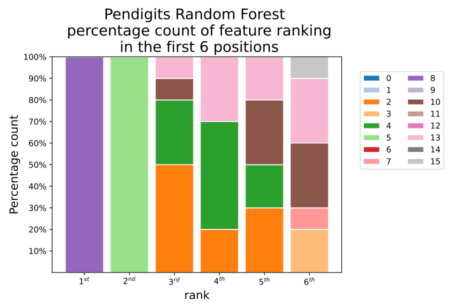
A very different result is obtained in *wine*, the smallest dataset with 129 elements and 13 dimensions with only 10 anomalies. In this context DIFFI and



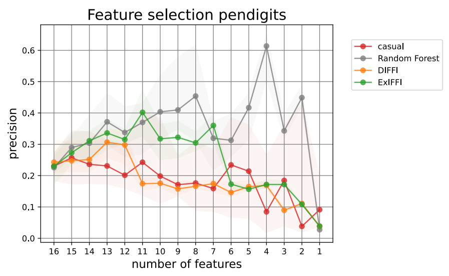
(a) DIFFI



(b) ExIFFI



(c) RF



(d) Feature Selection

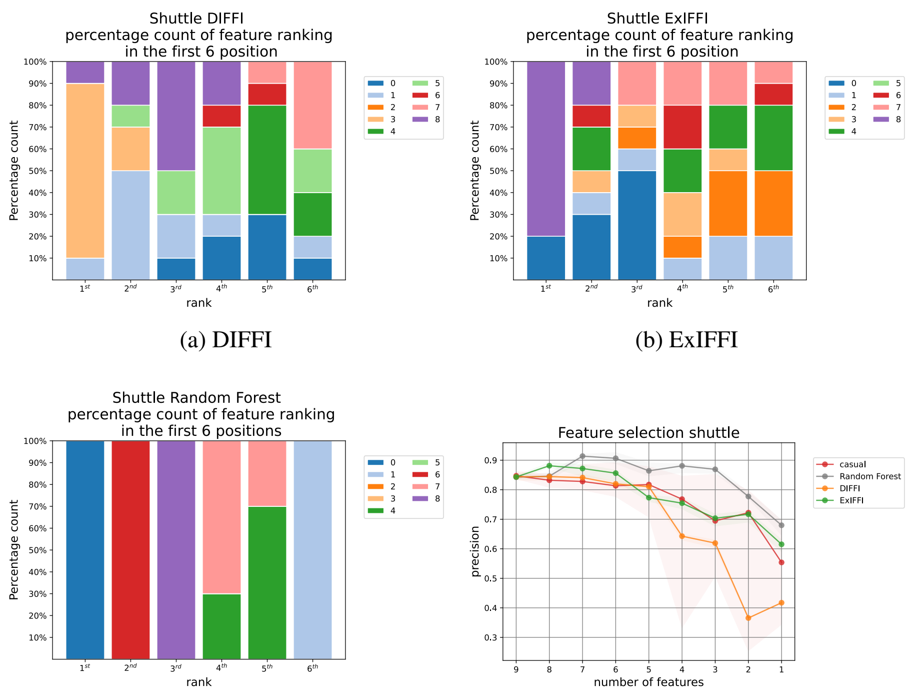
Figure 68: Consistency of the feature importance and feature selection applied to the *pendigits* dataset

ExIFFI strongly disagree, but only ExIFFI is able to detect the most important feature, the number 12 (Figure 70). This directly reflects in Figure 70d, where ExIFFI performs much better than DIFFI as it is able to select quite easily the most important feature for the detection.

In almost all feature selection Figures we have previously discusses, we can observe that the feature selection made using the ranking made by the ExIFFI algorithm are often above the "casual" feature ranking, stressing the fact that it is able to detect consistently the feature that contribute the most in defining if a sample is an outlier or not. Moreover we can observe that in datasets like *annthyroid*, *wine*, *breastw* and *pendigits* ExIFFI is one of the best feature selection strategies.

### 9.5.4 *Scoremap*

As in the results on synthetic dataset presented at the beginning of this Section, also on in the case of real datasets shown in Figure 71 the score map of ExIFFI has much less artifacts with respect to DIFFI. This is particularly evident in Figure 71d where we present the scoremaps of 4 different datasets showing the two most important features highlighted by DIFFI and ExIFFI. As in Section 9.5.2 we can observe how much the ExIFFI is able to generalize the importance in each point of the space, instead the DIFFI method is very biased both on the specific dataset of training and on the direction orthogonal to the principal dimensions.



(a) DIFFI

(b) ExIFFI

(c) RF

(d) Feature Selection

Figure 69: Consistency of the feature importance and feature selection applied to the *shuttle* dataset

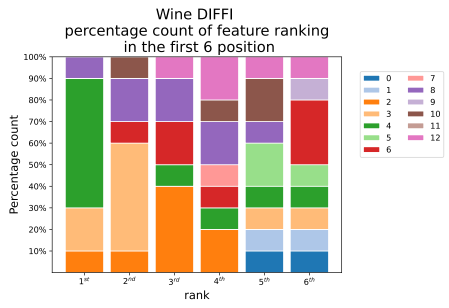
## 9.6 CONCLUSIONS

Anomaly detection is the unsupervised machine learning task to detect unusual patterns or behaviors in complex datasets and systems. However, simply identifying anomalous data is often not sufficient in real-world applications. It can be beneficial for users to understand the reasons behind the predictions, as this can help with root cause analysis and increase trust in the model.

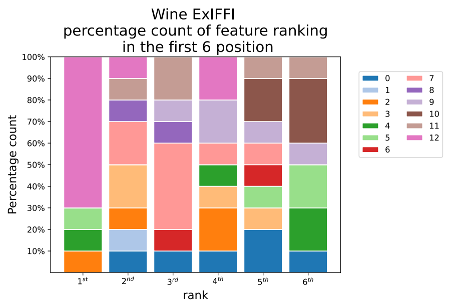
This Chapter introduces ExIFFI, a model-specific algorithm for providing both local and global interpretability for the Extended Isolation Forest (EIF). This unsupervised anomaly detection model addresses the training artifacts often present in Isolation Forest (IF). Drawing inspiration from DIFFI, a model-specific interpretability algorithm designed for IF, ExIFFI aims to improve upon it by generating explanations that are more similar to those provided by supervised proxies, such as feature importance computed by a Random Forest.

Future works should explore how to take advantage of the information encoded in the splitting nodes in other ways. Indeed, uncertainty often arises when multiple variables compete for the role of the most relevant feature rather than a single variable. This topic is a direction for further research.

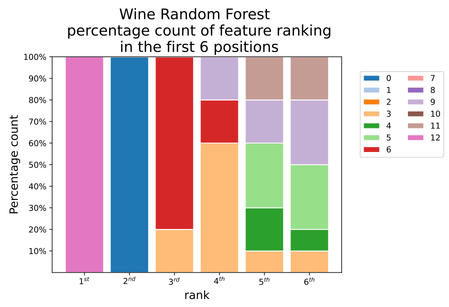
Moreover, an additional research direction could be to further assess the usage of ExIFFI for feature selection. XAI approaches for feature selection have been employed in few works in the literature (for example in [90]), but the task of feature selection in unsupervised AD settings is even more complex: typically unsupervised feature selection techniques completely discard the task



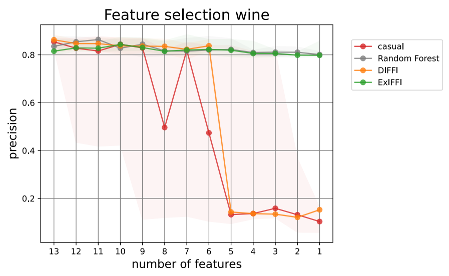
(a) DIFFI



(b) ExIFFI



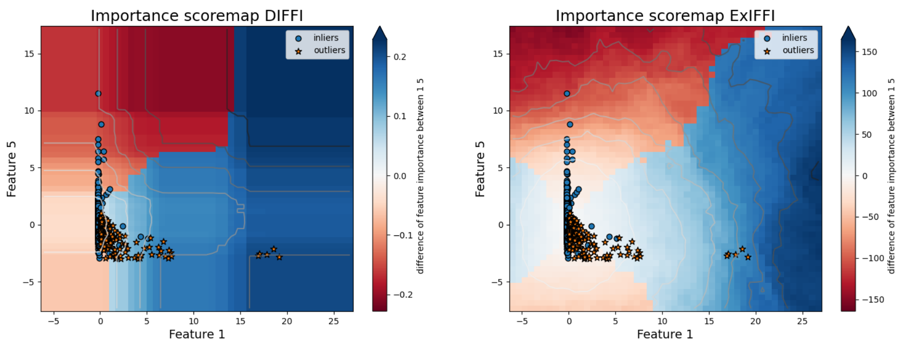
(c) RF



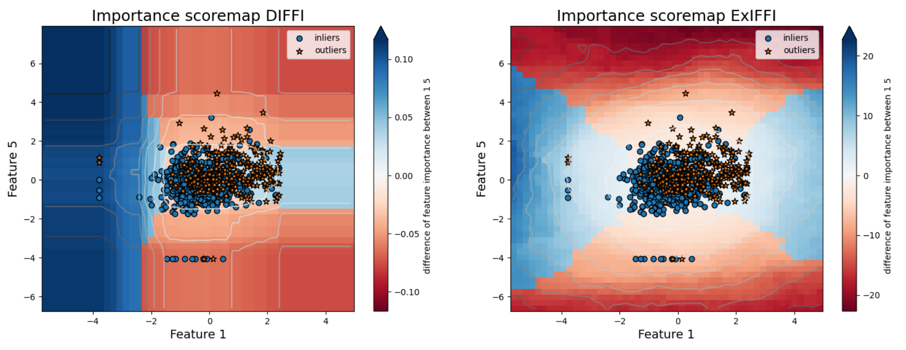
(d) Feature selection

Figure 70: Consistency of the feature importance and feature selection applied to the *wine* dataset

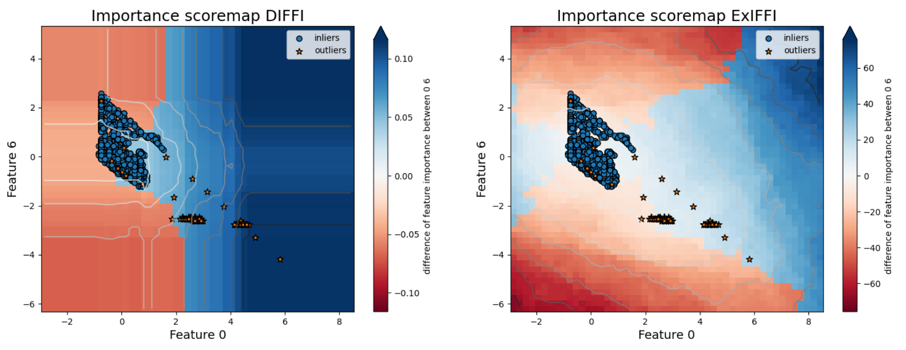
at hand in their mechanisms [198], which can potentially lead to suboptimal choices. While the potential of ExIFFI for feature selection have been showed here, focused research activities should be carried out in the future.



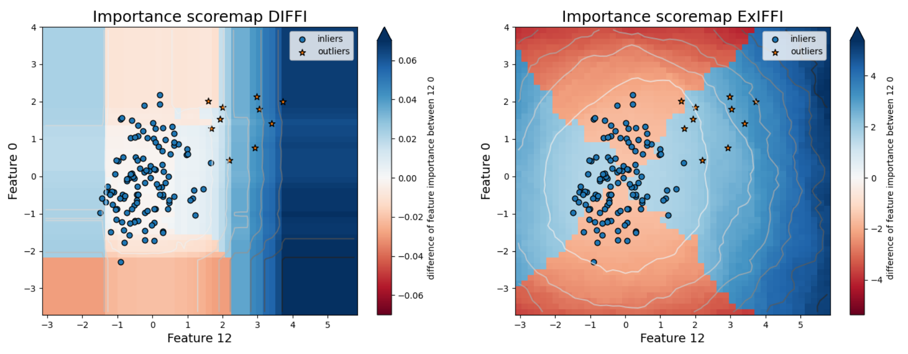
(a) *annthyroid*



(b) *pima*



(c) *shuttle*



(d) *wine*

Figure 71: Scoremaps of the importance value of the two most important features, according the RF feature ranking, in the space that surrounds the data distribution. Representation of the data with the inliers in form of a circle and the outliers in form of a star.

# 10 DOMAIN ADAPTATION FOR ANOMALY DETECTION

Despite the huge success of data-driven technologies, ML-based solutions present many scalability issues [94], especially in production environments where multiple similar systems are deployed. Even if the goal is to apply the same model to every system, this is rarely achievable because ML-solutions are very tied to the specific system they are trained on and do not easily scale to similar systems. This forces the developer to train multiple dedicated models, resulting in wasted resources.

In recent years new approaches are being developed to tackle this problem. Domain Adaptation (DA), that is a branch of Transfer Learning, is the research area that studies how to transfer knowledge between different but related tasks. It has been demonstrated DA utility in developing models that can adapt to different systems (for example, in the industrial context, to different machines), without the need to completely re-sample data from new system that needs a detector.

In this Chapter, the goal is to transfer some knowledge in the form of data samples between different but related systems. This allows to train a new detector that requires less data and therefore it can be deployed in a faster and cheaper way, enabling scalability of AD solutions.

## 10.1 INTRODUCTION

As said before, Anomaly detection (AD) is a very compelling task in applications where high dimensional processes need to be monitored. In recent years, many methods have been developed to automatically detect data having an unusual behaviour, allowing the AD problem to be solved. These methods learn from data adapting to the specific process they are asked to observe but usually require a large amount of training samples that might not be always available. When there is no sufficient data to train a model, it might be useful to transfer some knowledge from a different but related process.

An inspiring example for this scenario was the following: a newly installed industrial machine (a new multiphase flow meter for example) typically has no available data describing its behaviour; on the other hand, similar machines could have already being installed in the past and data related to their functioning could be available. In such scenario, the simplest idea might be to train a model on data collected from other machines and directly applying it to the new one: unfortunately due to different sensor calibration, the performance of the model applied to the old and new instrument might be very different. Even after tuning, the performances might be different due to different manufacturers, environmental conditions and sampling process. A strategy that might be worth to try is to adapt data coming from the old machine, to the new one, and this is known in literature as Domain Adaptation (DA) [148]. Both of the machine types (old ones and new) can be mathematically represented by two distributions, namely the *source* distribution for the oldest and best sampled machine, and the *target* distribution for the newest one. The final goal is to create a model able to detect anomalies in the *target* domain, using the knowledge acquired in the *source* and a small dataset coming from the target distribution. To transfer the data from the source to the target distribution, it is necessary to previously learn a map between the two domains, and this is the main problem discussed in this Chapter. Once the samples from the old machine are mapped in the new machine domain, it is possible to train a new model in the new dataset enriched by the transported source samples.

### 10.1.1 *Problem settings and transport problem*

NOTATION Let $\Omega_s$ and $\Omega_t \subset \mathbb{R}^d$ denote the source and target domains respectively and $\mathcal{Y} = \{0, 1\}$ be the set of labels indicating the presence or absence of anomaly, where 0 is used for the *inlier* class and 1 corresponds to the *outlier* class. Given the source and target datasets denoted as $\mathrm{X_s} = \{\mathbf{x}_{s_i}\}_{i=1}^{n_s}$ and $\mathrm{X_t} = \{\mathbf{x}_{t_i}\}_{i=1}^{n_t}$, $\mu_s$ and $\mu_t$ are the marginal distributions of the source and target datasets defined as:

$$\mu_s = \sum_{i=1}^{n_s} m_s^i \delta_{x_i^s} \qquad \mu_t = \sum_{j=1}^{n_t} m_t^j \delta_{x_j^t}$$

where $\delta_{x_i}$ is the Dirac delta function at the location $x_i$ and $m^i$ is the mass associated with that point $\sum_{i=1}^{n_s} m_s^i = m_s$ and $\sum_{j=1}^{n_t} m_t^j = m_t$.

ADAPTATION PROBLEM To make the transfer learning problem meaningful, the *source* and the *target* distributions are assumed to be different but somehow related to each other, i.e., sharing some kind of structure. However the target domain is assumed to contain fewer operating conditions (i.e. classes): in this context, common techniques like standardization and normalization are not applicable and even more refined techniques are hard to be applied.

Following the same assumption made in [58], the domain drift is assumed to be due to an unknown map of the input space $T : \Omega_s \longrightarrow \Omega_t$. This map may have a physical interpretation such as thermal noise, change in acquisition conditions or even sensor drifts. This map $T$, known as *push-forward* operator or *transport map*, is assumed to preserve the conditional distribution:

$$\mathrm{P}_s(y|x^s) = \mathrm{P}_t(y|T(x^s))$$

meaning that the probability of a sample to be an outlier in the two domains does not depend on the map $T$. Applying the transport $T$ to the measure $\mu$ is it possible to obtain the *image measure* $T_\#\mu$ that satisfies the following property.

The push-forward operator $T$ applied to the measure $\mu_s$ satisfies the inequality:

$$T_\#\mu_s(x) \geq \mu_t(x), \quad \forall x \in \Omega_t, \tag{59}$$

which traduces the fact that the *source* data are fully sampled, in the sense that the source data have been acquired under all existing conditions, contrary to the target data that are less numerous and acquired under specific conditions; a pictorial representation of such scenario is reported in Figure 72. Note that (59) induces a potential loss of mass from the source to the target datasets.

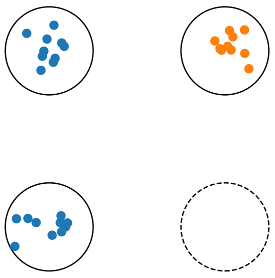


Figure 72: A simple example of the addressed problem. Source data are depicted in blue, while the available target data in orange. Between the two distributions there is a simple translation but the target is under sampled in the sense it was not sampled in every condition and misses a cluster. As the data are unlabelled, it is difficult to know a priori which source structure should match the available target.

MAIN IDEA Following the same steps described in [58], the subspace mapping problem can be formalized as follows:

1. find a *transport map* $T$ from $\mu_s$ to $\mu_t$ satisfying the previously discussed assumptions.
2. use the estimated map $T$ to transport samples $\mathrm{X_s}$ from $\Omega_s$ to $\Omega_t$
3. train an anomaly detection model on $\Omega_t$, using $\mathrm{X_t}$ and the transported samples $T(\mathrm{X_s})$.

The core of the problem lies in Step 1, while Step 2 only consists in the application of a map to the source dataset, and Step 3 requires to apply a generic anomaly detector model to the transported samples. That said, this work will focus in the solution of Step 1 i.e. in the estimation of the transport map between the source samples and the available target set. The background of the proposed solution will be discussed in Section 10.2, while the main idea concerning Step 1 will be described in Section 10.3. Then in Section 10.4 similar approaches will be compared on both synthetic and real dataset and in Section 10.5 conclusions will be drawn.

## 10.2 BACKGROUND

MONGE FORMULATION Among all the possible transformations $T$, one may look for the one that minimizes a transportation cost subjected to some regularization terms and some constraints, like the mass conservation that avoids the mass to be lost during the transportation. More precisely, the Monge formulation is looking for the transformation $T$ that minimizes the cost function [83, 192]:

$$C(T) = \int_{\Omega_s} c(x, T(x)) d\mu_s(x)$$

where $c : \Omega_s \times \Omega_t \rightarrow \mathbb{R}^+$ is a distance that can be viewed as the energy to move a mass from $x$ to $T(x)$, subjected to the mass conservation equation. Unfortunately the aforementioned problem does not always have a solution, e.g., when the total source and target masses are different.

KANTOROVICH FORMULATION Kantorovitch formulation relaxes the Monge problem by looking at a general probabilistic coupling $P$ between $\Omega_s$ and $\Omega_t$ such that:

$$P_0 = \underset{P}{\operatorname{argmin}} \int_{\Omega_s \times \Omega_t} c(x^s, x^t)\ dP(x^s, x^t),$$

that transports all the samples of the source to the target and still satisfies the mass conservation. In a discrete setting, the Kantorovitch formulation can be reduced to:

$$\boldsymbol{P}_0 = \underset{\boldsymbol{P}}{\operatorname{argmin}} \langle \boldsymbol{C}, \boldsymbol{P} \rangle,$$

where each entry of $\boldsymbol{C}$ denoted as $C_{ij}$ represents the cost to move the mass $P_{ij}$ from the source $i$ to the target $j$ and $\boldsymbol{P}$ is an $\mathrm{n_t} \times \mathrm{n_s}$ matrix containing all the couplings $P_{ij}$. This problem is constrained by the mass conservation equations:

$$\boldsymbol{P}\mathbb{1}_{\mathrm{n_t}} = m_s,$$
$$\boldsymbol{P}^T \mathbb{1}_{\mathrm{n_s}} = m_t,$$

where $\mathbb{1}_{\mathrm{n_t}}$ and $\mathbb{1}_{\mathrm{n_s}}$ are vectors of ones of sizes $\mathrm{n_t} \times 1$ and $\mathrm{n_s} \times 1$. These constraints impose no loss or creation of mass during the transport. After computing the optimal *coupling* $\boldsymbol{P}$, it is possible to map the samples from $\Omega_s$ to $\Omega_t$. This can be done by using the *barycentric mapping*, which can be written as follows (when using a squared Euclidean cost $c$ between pairs of source and target points) [192]:

$$\hat{\mathrm{X}}_\mathrm{s} = \mathrm{diag}(\boldsymbol{P}\mathbb{1}_{\mathrm{n_t}})^{-1} \boldsymbol{P}\, \mathrm{X_t} \tag{60}$$

In this way, each source sample is mapped into the convex hull of the target dataset. Unfortunately, computing these barycenters is not equivalent to estimate the previously discussed transport map $T$ since the barycentric mapping (Equation (60)) only applies to the samples used to compute the coupling and $T$ should be much more general and applicable to any point in the source domain.

MAPPING ESTIMATION A solution proposed in [192] to solve this problem is to jointly learn the coupling and the transport map by means of a new cost function:

$$\left\| T(\mathrm{X_s}) - \hat{\mathrm{X}}_\mathrm{s}(\boldsymbol{P}) \right\|_{\mathcal{F}}^2,$$

where $\|\cdot\|_{\mathcal{F}}$ indicates the Frobenius norm.

When the (probability) masses of the two distributions are different, it is not possible to directly apply the tools previously discussed; indeed, in such cases the mass conservation constraints are no longer applicable. The present work introduces a new solution by learning the transport map in an unbalanced setting, i.e., when *source* and *target* share only some operating conditions and the mass cannot be preserved during the coupling as the two distributions may not show corresponding concepts, as represents in Figure 72. In that simple example source and available target, even if related, do not contain the same amount of information: the available target is not only shifted but it was only sampled in one condition. Without knowledge of the missing information, in unsupervised scenarios, any approach that maps the source to the available target risks to estimate the wrong transport map.

## 10.3 PROPOSED TRANSFER LEARNING STRATEGY FOR MACHINE DATA

### 10.3.1 *Transport map estimation*

Given the ideas discussed so far, the loss to be minimized *jointly* learns the coupling and the transport map in the so-called unbalanced scenario (corresponding to masses not preserved during transport) and it is defined as follows:

$$\underset{\boldsymbol{P},T}{\operatorname{argmin}}\, f(\boldsymbol{P},T) = \underset{\boldsymbol{P},T}{\operatorname{argmin}}\, g(\boldsymbol{P}) + h(\boldsymbol{P},T) \tag{61}$$

$$g(\boldsymbol{P}) = \frac{\lambda_P}{\max(\boldsymbol{C})}\langle \boldsymbol{C},\boldsymbol{P}\rangle_{\mathcal{F}} + \left[\langle \boldsymbol{\lambda}_s, \mathrm{m_s} - \boldsymbol{P}\mathbb{1}_{\mathrm{n_t}}\rangle + \langle \boldsymbol{\lambda}_t, \mathrm{m_t} - \boldsymbol{P}^T\mathbb{1}_{\mathrm{n_s}}\rangle\right] \tag{62}$$

$$h(\boldsymbol{P},T) = \frac{1}{\mathrm{n_s\,d_t}}\left\| [T(\mathrm{X_s}) - \hat{\mathrm{X}}_\mathrm{s}(\boldsymbol{P})]\,\mathrm{W} \right\|_{\mathcal{F}}^2 + \frac{\lambda_T}{\mathrm{d_s\,d_t}} R(T) \tag{63}$$

with:

$$\boldsymbol{P}\mathbb{1}_{\mathrm{n_t}} \leq m_s$$

$$\boldsymbol{P}^T\mathbb{1}_{\mathrm{n_s}} \leq m_t$$

This loss is composed of many terms that will be analysed in detail in the next paragraphs, and are named *coupling* term, *mass destruction* term, *map estimation* term and the last, *regularization* term.

COUPLING TERM The first part of the coupling term $g$ in Equation (62) results from from the standard Kantorovitch formulation of the optimal transport problem. It is defined as the Frobenius scalar product between the cost matrix $\boldsymbol{C}$

and the coupling matrix $\boldsymbol{P}$. Each entry $C_{ij}$ of $\boldsymbol{C}$ expresses the cost to transport a unit of mass from the source position $x_i^s$ to the target $x_j^t$, while $P_{ij}$ shows how much mass flows from the source $i$ to the target $j$. In the standard Kantorovitch formulation, mass conservation equations $\boldsymbol{P}\mathbb{1}_{\mathrm{n_t}} = m_s$ and $\boldsymbol{P}^T\mathbb{1}_{\mathrm{n_s}} = m_t$ are introduced, meaning that the full (probability) mass is transferred between the source and target domains. This work relaxes this constraint to avoid couplings between samples that do not correspond to the same situation, e.g. operating conditions seen by the source machine but not the target or anomalies in the source dataset that are not necessarily present in the target dataset. This means only some source samples share their mass with the target samples: these samples will be referred to as *transported vectors* because they support the transport mapping described in the following paragraphs.

MASS DESTRUCTION The relaxation on the mass preservation constraint has to be counterbalanced by a penalization term that avoids all the mass is destructed during coupling. Many different divergences can be found in literature to solve this issue [48, 214]. The mass destruction term considered in this work defined in $g$ (second addend in Equation (62)) simply weights the mass that has not been coupled by a cost $\lambda_i$ associated with the $i$th sample of the source or target database. The higher this cost, the more difficult the $i$-th sample is removed from the optimization process. The penalization vectors $\boldsymbol{\lambda}_s = (\lambda_1, ..., \lambda_{\mathrm{n_s}})^T$ and $\boldsymbol{\lambda}_t = (\lambda_1, ..., \lambda_{\mathrm{n_t}})^T$ can be adjusted manually or learnt automatically using some datasets: in the former strategy, a general rule might be to set the cost to delete a target sample higher than the source sample to take into account the small number of target samples. To avoid complex hyper-parameter tuning, it was decided to rely on uniform deletion costs for each source and target point. In principle, these costs might also encode some prior knowledge on the samples, e.g., anomalous source points should be cheaper to delete with respect to inliers. However, assigning a different cost to each source point may be a difficult task. An alternative is to estimate these costs by training an anomaly detector on the source dataset and to assign to each point a cost proportional to its anomaly score, as suggested in [186]. In this case, anomalous points are easy to delete and tend not to be mapped on the target, whereas inlier source points are more likely to be mapped in the target domain.

The $g$ term can be rewritten as:

$$\begin{aligned} g(\boldsymbol{P}) &= \langle \hat{\boldsymbol{C}}, \boldsymbol{P} \rangle_{\mathcal{F}} + \text{const.} \\ \hat{\boldsymbol{C}} &= \frac{\lambda_P}{\max(\boldsymbol{C})}\boldsymbol{C} - [\boldsymbol{\lambda}_s \mathbb{1}_{\mathrm{n_t}}^T + \mathbb{1}_{\mathrm{n_s}} \boldsymbol{\lambda}_t^T] \end{aligned}$$

defining a new cost matrix $\hat{\boldsymbol{C}}$ that takes into account the mass destruction penalty term.

MAP ESTIMATION The map estimation term (Equation (63)) allows the transport map $T$ to be estimated based on the estimated coupling $\boldsymbol{P}$ [192]. Note that $T(\mathrm{X_s})$ is the transport map applied to the source data, while $\hat{\mathrm{X}}_{\mathrm{s}}(\boldsymbol{P})$ is the barycentric mapping of the source data on the target samples. As a consequence, minimizing the map estimation term tends to reduce the discrepancy between these two terms that ideally should be identical. Estimating a correct transport map is the primary goal to develop an effective detector for the target machine. Once the transport map has been computed between the source and target transported vectors, this map can be applied to every source sample enriching the target domain not only with the source inliers but also with the source outliers and the unmatched operating conditions. The form of the map $T$ can be arbitrary, depending on the complexity of the relationship between the source and target databases. This work considers linear transformations $T(\mathrm{X_s}) = \mathrm{X_s}\,\boldsymbol{L}$ that have the advantage to be simple to compute and to avoid overfitting issues. Moreover, in case of adaptation between machines that differ in size, neglecting small nonlinearities, linear maps seems to be the most reasonable transformations. In the following, the biased and the unbiased choices for the $L$ matrix will be tested. An additional weighting matrix W is also introduced to account for the discrepancy between the source samples mapped with the barycentric function and the transported source samples. One possibility could be to weight it by the deletion cost ($\mathrm{W} = \mathrm{diag}(\lambda_s)$), enforcing a strong prior on the solution. A more natural approach that is indeed used in our formulation is to introduce weights defined using the masses of the coupled points, i.e., $\mathrm{W} = \mathrm{diag}(\boldsymbol{P}\mathbb{1}_{\mathrm{n_t}})$, allowing only transported vectors to contribute to the estimation.

REGULARIZATION TERM The penalization term (second part addend of Equation (63)) is intended to regularize the transport map estimation problem. This work relies on the same regularization investigated in [192], that tries to ensure samples are not moved in too far locations. This regularization term is controlled by the hyper-parameter $\lambda_T$.

If $\mathrm{W} = \mathrm{diag}(\boldsymbol{P}\mathbb{1}_{\mathrm{n_t}})$ the $h$ term can be rewritten in this form:

$$
\begin{aligned}
h(\boldsymbol{P}, T) = & \frac{1}{\mathrm{n_s}\,\mathrm{d_t}} \left\| [\mathrm{X_s}\,\boldsymbol{L} - \mathrm{diag}(\mathrm{P1_{n_t}})^{-1}\,\boldsymbol{P}\,\mathrm{X_t}]\ \mathrm{diag}(\mathrm{P1_{n_t}}) \right\|_{\mathcal{F}}^2 \\
& + \frac{\lambda_T}{\mathrm{d_s}\,\mathrm{d_t}} \|\boldsymbol{L} - \boldsymbol{I}\|_{\mathcal{F}}^2
\end{aligned}
$$

$$
\begin{aligned}
& T(\mathrm{X_s}) = \mathrm{X_s}\,\boldsymbol{L} \\
& \hat{\mathrm{X}}_{\mathrm{s}}(\boldsymbol{P}) = \mathrm{diag}(\mathrm{P1_{n_t}})^{-1}\,\boldsymbol{P}\,\mathrm{X_t} \\
& \mathrm{W} = \mathrm{diag}(\mathrm{P1_{n_t}}) \\
& R(T) = \|\boldsymbol{L} - \boldsymbol{I}\|_{\mathcal{F}}^2
\end{aligned}
$$

### 10.3.2 *Optimization using Franke-Wolfe and Block-Coordinate Descent*

The optimization problem described in Equation 61 is solved using the Block-Coordinate Descent (BCD) algorithm as in [193]. The BCD algorithm recursively estimates $\boldsymbol{P}$ for a fixed $T$ and vice versa. These two steps are detailed in the next sections.

*Solving for* $\boldsymbol{P}$ *with T fixed*

The optimization problem is a constrained quadratic problem and is solved by means of the Frank Wolfe algorithm [87], that reaches the solution solving iteratively the linearized problem. Indeed this algorithm is based on two main steps:

1. Direction finding: given an initial solution $\boldsymbol{P}_k$, find $\boldsymbol{S}_k$ that satisfies the constrains and minimises $\boldsymbol{S}^T \nabla f(\boldsymbol{P}_k)$. This step allows to find the direction in $\boldsymbol{P}_k$ that minimises the function $f$ inside the space of possible solutions. This step can be solved by means of a linear program solver.

2. Step size determination: find the step size $\alpha_k$ that minimizes the function along the direction $\boldsymbol{S}_k$, therefore $\alpha_k = \text{argmin}\, f(\boldsymbol{P}_k + \alpha_k(\boldsymbol{S}_k - \boldsymbol{P}_k))$.

Then, the solution is updated $\boldsymbol{P}_{k+1} = \boldsymbol{P}_k + \alpha_k(\boldsymbol{S}_k - \boldsymbol{P}_k)$ and the cycle continues.
Applying the first step to our problem requires to compute the gradient w.r.t. $\boldsymbol{P}$. To compute this algorithm, the gradient w.r.t. $\boldsymbol{P}$ is necessary:

$$\begin{aligned}
\nabla_{\boldsymbol{P}}\, f(\boldsymbol{P}, \boldsymbol{L}) &= \nabla_{\boldsymbol{P}}\, g(\boldsymbol{P}) + \nabla_{\boldsymbol{P}}\, h(\boldsymbol{P}, \boldsymbol{L}) \\
\nabla_{\boldsymbol{L}}\, f(\boldsymbol{P}, \boldsymbol{L}) &= \nabla_{\boldsymbol{L}}\, h(\boldsymbol{P}, \boldsymbol{L})
\end{aligned}$$

and:

$$\begin{aligned}
\nabla_{\boldsymbol{P}}\, g(\boldsymbol{P}) &= \hat{\boldsymbol{C}} \\
\nabla_{\boldsymbol{P}}\, h(\boldsymbol{P}, \boldsymbol{L}) &= \frac{2}{\mathrm{n_s\, d_t}} \Big[ \operatorname{diag}\left(\operatorname{diag}^{-1}\left(\mathrm{X_s}\, \boldsymbol{L}\boldsymbol{L}^T\, \mathrm{X_s}^T\right)\right) \boldsymbol{P} \mathbb{1}_{\mathrm{n_t}} \mathbb{1}_{\mathrm{n_t}}^T \\
&\quad - \operatorname{diag}^{-1}\left(\boldsymbol{P}\, \mathrm{X_t}\, \boldsymbol{L}^T\, \mathrm{X_s}^T\right) \mathbb{1}_{\mathrm{n_t}}^T \\
&\quad - \operatorname{diag}(\boldsymbol{P}\mathbb{1}_{\mathrm{n_t}})\, \mathrm{X_s}\, \boldsymbol{L}\, \mathrm{X_t}^T \\
&\quad + \boldsymbol{P}\, \mathrm{X_t}\, \mathrm{X_t}^T \Big]
\end{aligned}$$

*Solving for T with* $\boldsymbol{P}$ *fixed*

This step is the simplest, indeed it simply involves a matrix inversion.

$$\begin{aligned}
\nabla_{\boldsymbol{L}} f(\boldsymbol{L}, \boldsymbol{P}) = &2\Big[\frac{1}{\mathrm{n_s\, d_t}}\, \mathrm{X_s}^T \operatorname{diag}(\boldsymbol{P}\mathbb{1}_{\mathrm{n_t}})^2\, \mathrm{X_s} + \frac{\lambda_T}{\mathrm{d_s\, d_t}} \boldsymbol{I}\Big] \boldsymbol{L} + \\
&- 2\Big[\frac{1}{\mathrm{n_s\, d_t}}\, \mathrm{X_s}^T \operatorname{diag}(\boldsymbol{P}\mathbb{1}_{\mathrm{n_t}}) \boldsymbol{P}\, \mathrm{X_t} + \frac{\lambda_T}{\mathrm{d_s\, d_t}} \boldsymbol{I}\Big]
\end{aligned}$$

## 10.4 EXPERIMENTS

Due to the lack of datasets describing two similar machines that are both suited for anomaly detection task, i.e. that are labelled with inlier/outlier class, we decided to resort to a different (more qualitative) approach.

The goal is to test different strategies to train an anomaly detector in presence of few target samples. The specific anomaly detection model is not important but in the following discussion we rely on One Class SVM because it is conceptually simple and has a decision function easy to visualize. To assess the performance of the training we will compare two decision boundaries: the one obtained with the full target dataset (that is assumed not be available) and the one obtained with partial information. Looking at the proximity between these two boundaries we will qualitatively understand the best adaptation strategy among the tested. We tested two main training choices, that are:

- training on raw data without adaptation.
- adapting source data to target data and then training the model on the adapted data $T(X_s)$.

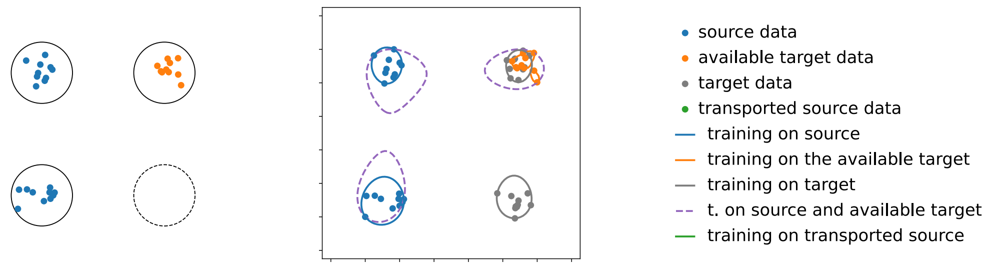
Concerning the first option, the three choices are training on the source $X_s$, training on the available target $X_t$ and training on the union between these two basic datasets $X_s \cup X_t$. Concerning the adaptation option, we tested 6 adaptation strategies that can be grouped into 3: *baseline* strategies (like *standardization* and *normalization*), *balanced* and *unbalanced* Optimal Transport strategies (shortly B-OT and U-OT). Learning the transport map we tested the biased estimation choice, and the unbiased.

### 10.4.1 *Synthetic dataset experiments*

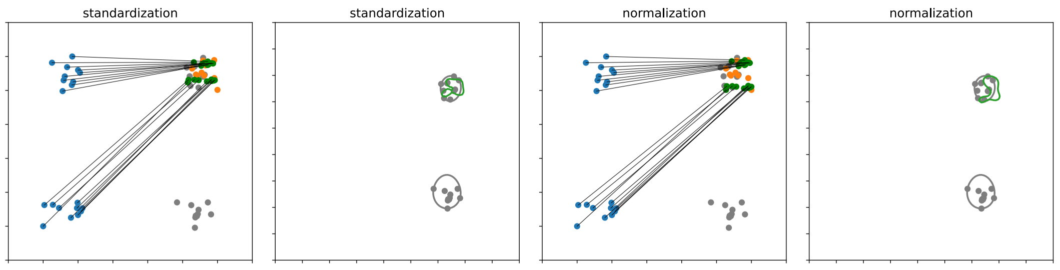
Five synthetic datasets were generated to validate the discussed methodology, three of which are made up of clusters that clearly define different classes or operating conditions while the last two showing a more challenging and less discrete cases. The available target data lacks one cluster or a portion of operating area making impossible for traditional adaptation stategies to recover the correct map and transfer the information of the missing data.

The simplest of the toy datasets tested in this Chapter is depicted in Figure 73: the source is composed of two clusters while the available target contains only a portion of all the target conditions, i.e. it only shows the upper cluster. Between the source and the target domain there is only a simple shift of coordinates. It is easy to see that only the U-OT allowed to estimate the bias vector is able to correctly reconstruct the transport. The other methods fail by construction or because they look for a match that saves the dataset mass or because they are not allowed to learn the shift like in U-OT without bias estimation.

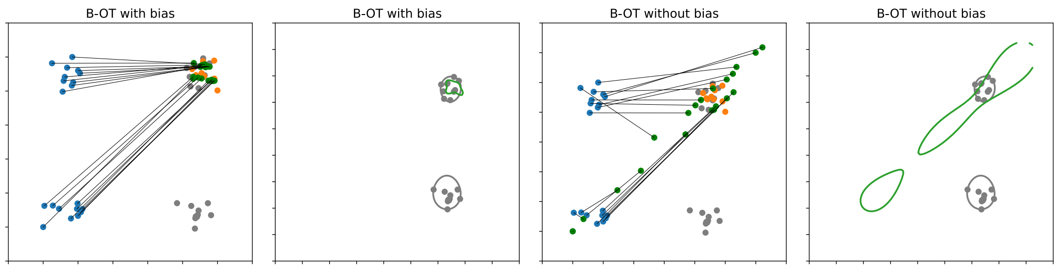
Two slightly more complex datasets are depicted in Figures 74,75 where a structure made up of three clusters is shifted and rotated. Here the missing information that is asked to transport between the two domains is a cluster in the corner of the triangle. As in Figure 73, also in Figure 74 the baselines and B-OT map the source in the convex hull of the available target leading to wrong adaptations, while U-OT with bias estimation is able to map the correct cluster in missing position. On the other hand Figure 75 shows a more complex scenario where training the detection algorithm directly on the union of source



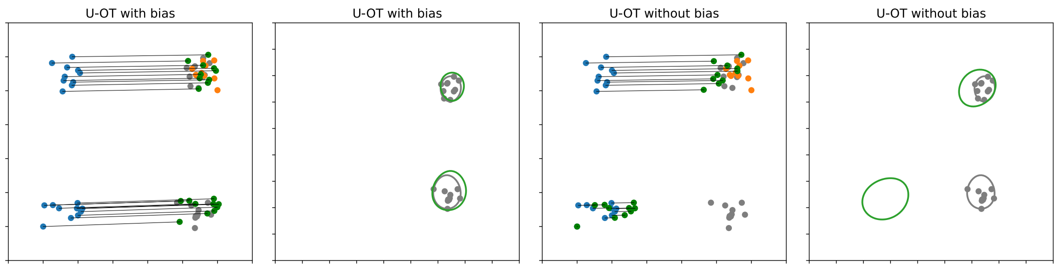
(a) Data and baseline models.



(b) Baseline adaptations.



(c) Balanced OT adaptations.



(d) Unbalanced OT adaptations.

Figure 73: Shifted double cluster. Blue dots are source samples, orange are the available target and gray are the full target dataset that unfortunately is hidden to the learner. The lines corresponds to models trained using source data (blue line), available target data (orange line), full target data (gray line), source and available target (purple dashed line), transported source data (green line)

and available target might be a competitive choice against the training on the adapted source due to the similarity between source and target. However in this case the best performing adaptation algorithm is the U-OT that is allowed to estimate the rotation only.

At this point, two more complex toy datasets are presented, where the missing information is not a discrete class but a continuous area of the space. The first example is the made up of a line that is horizontal in the source domain, and rotated in the target domain (Figure 76). The difficulty here lies in the target sampling process that avoided the acquisition of half of the rotated line. As expected, even if U-OT with bias and B-OT without bias get close results, the only method that accurately reconstructs the rotated and broken line is U-OT without bias estimation.

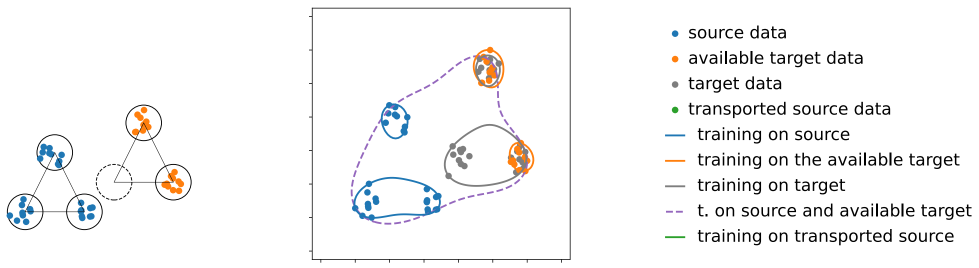
The last fully synthetic dataset is an adaptation of the banana dataset shown in [260]. In the cited paper the goal was to transfer the support vectors obtained in a source dataset with a banana shape, to a new target banana that is less sampled, slightly rotated and translated. In the context of this work an interesting adaptation is shown in Figure 77 where the target banana is assumed to be sampled in a much smaller area, a half of the full area. In this settings, to reconstruct the correct transformation between source and target is a much more challenging problem and requires the algorithm to automatically learn which is the portion of the datataset that has to be matched and where is the information that has to be transferred between the two domains. Also in this case U-OT performs significantly better than the other approaches.

A transition between synthetic datasets and real ones is the Iris dataset [72] where the popular dataset has been sub-sampled in order to get the source and the target set, then the target has been rotated and deprived of a flower specie associated to the class 0. This test is important because it shows the presented approach does not work only in low dimensional cases like the 2D example in Figure 78, but it also works on the full set of features whose results are shown in Figure 79. Intuitively, the best approach maps source points belonging to a class, to target points of the corresponding class, therefore the pairwise distance of adapted source and target points belonging to the same class is a good proxy of the adaptation performances. In fact, Figures 78 and 79 show the barplots of the pairwise distances between points belonging to each class, computed using the tested approaches. Here it is easy to see U-OT is able to reduce the classes mismatch, in particular the mismatch concerning the missing information.

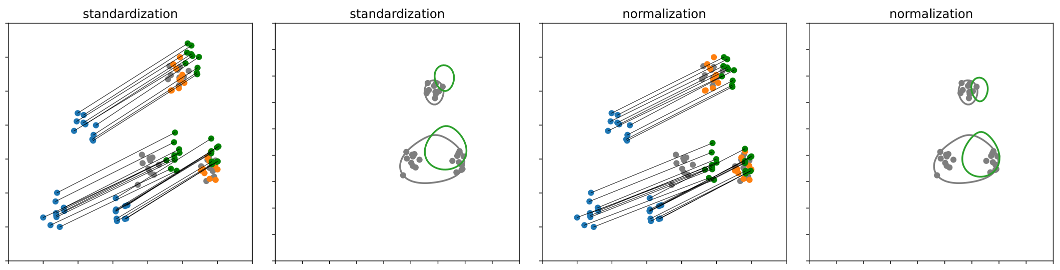
### 10.4.2 *Real dataset experiments*

In the following, test of the unbalanced transport on real datasets (related to different domains, biology and industry) will be reported.

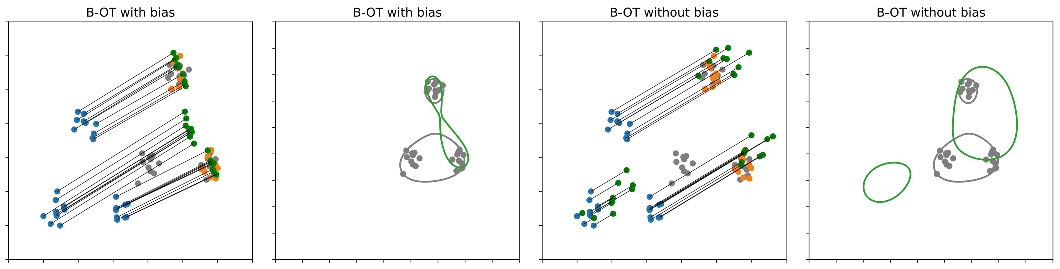
Firstly, it has been tested on the Palmer penguins datasets [115] describing three species of penguins living in three different islands, both male and female. In the present work the male penguins were used as source dataset, the female as target and one specie of penguin (class 2) was masked to simulate the



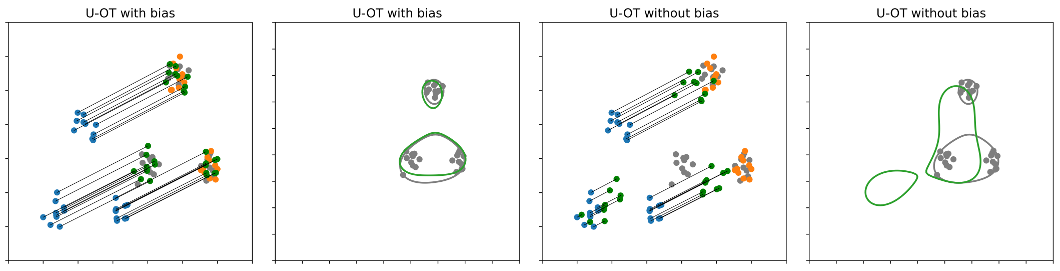
(a) Data and baseline models.



(b) Baseline adaptations.

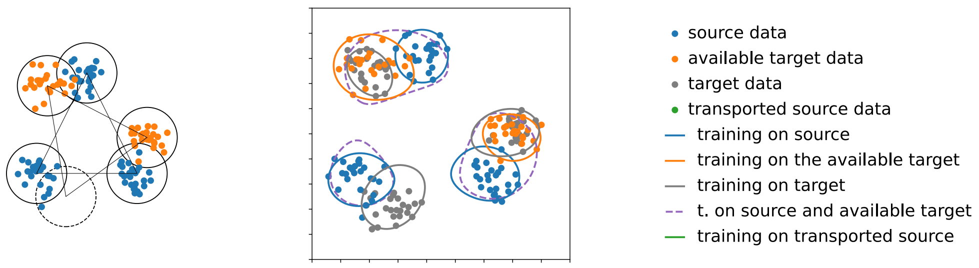


(c) Balanced OT adaptations.

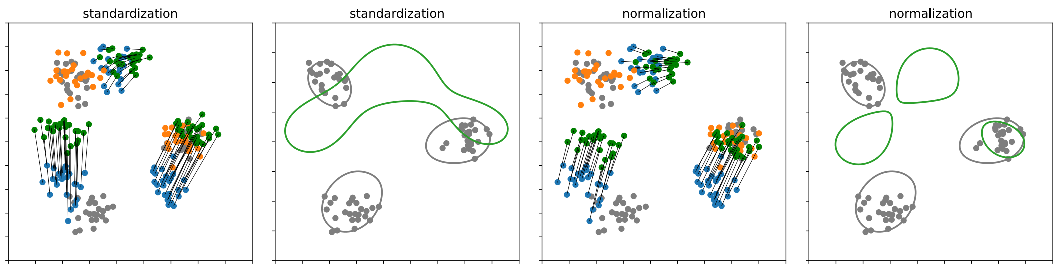


(d) Unbalanced OT adaptations.

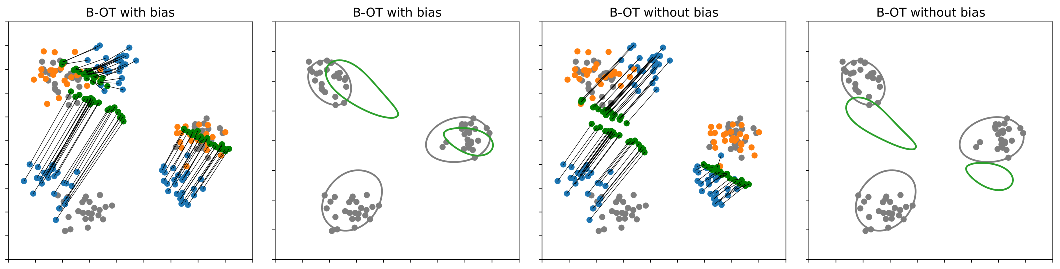
Figure 74: Shifted clusters' triangle.



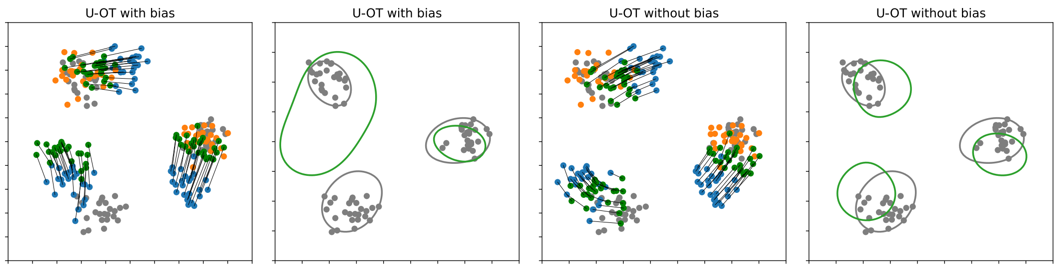
(a) Data and baseline models.



(b) Baseline adaptations.

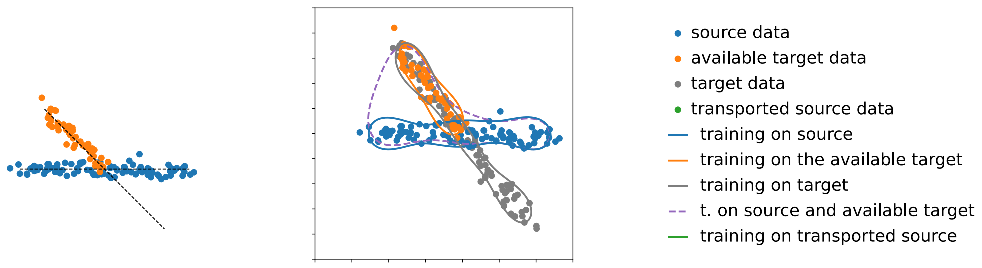


(c) Balanced OT adaptations.

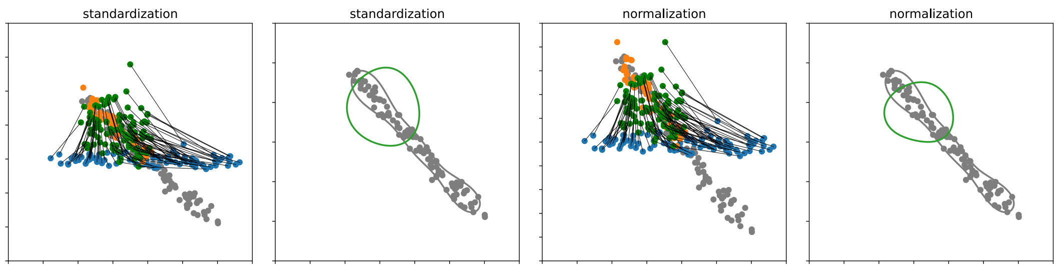


(d) Unbalanced OT adaptations.

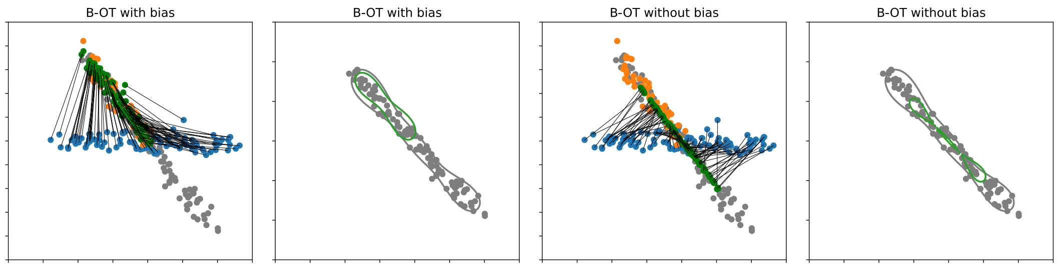
Figure 75: Rotated clusters' triangle.



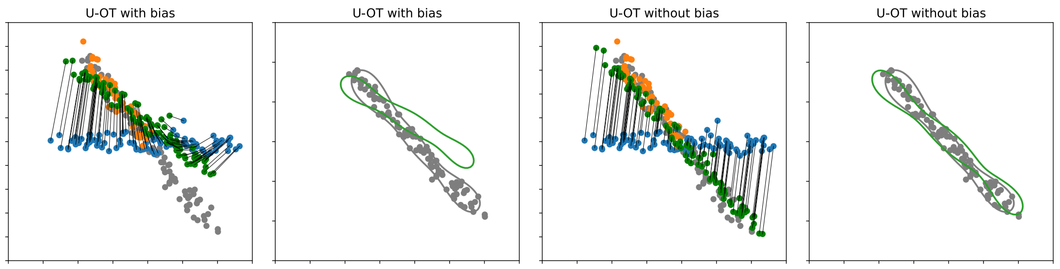
(a) Data and baseline models.



(b) Baseline adaptations.

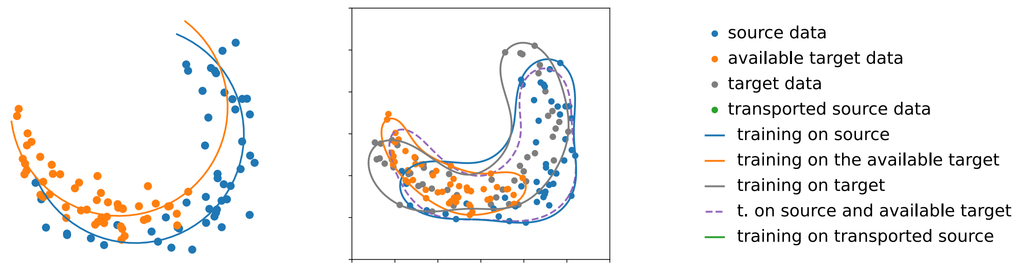


(c) Balanced OT adaptations.

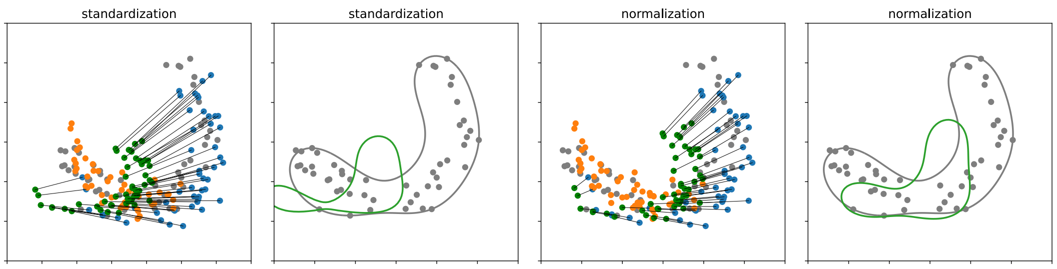


(d) Unbalanced OT adaptations.

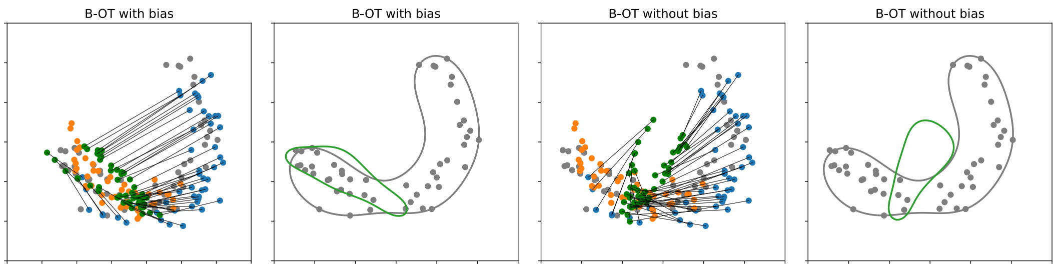
Figure 76: Rotated line.



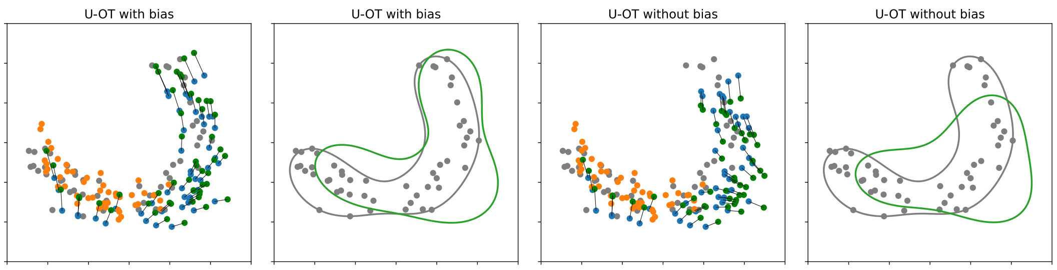
(a) Data and baseline models.



(b) Baseline adaptations.

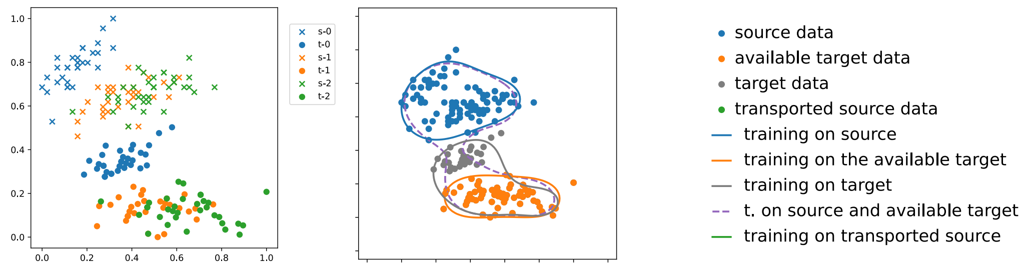


(c) Balanced OT adaptations.

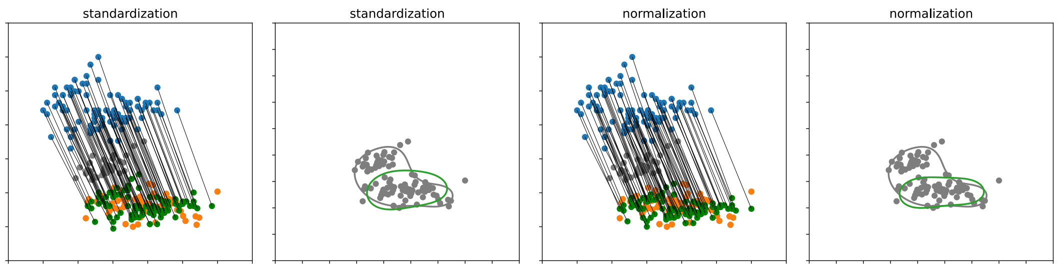


(d) Unbalanced OT adaptations.

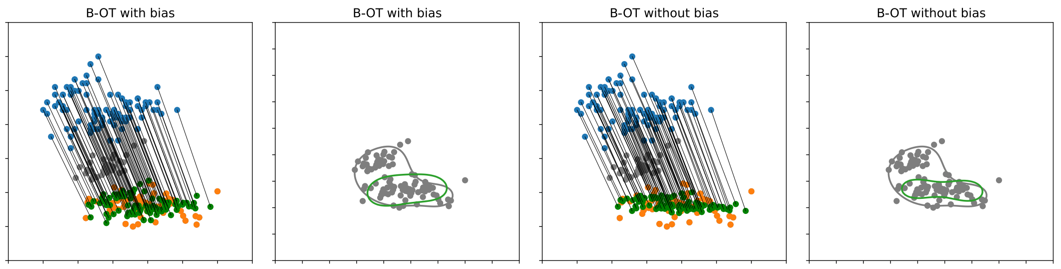
Figure 77: Banana dataset adapted from [260].



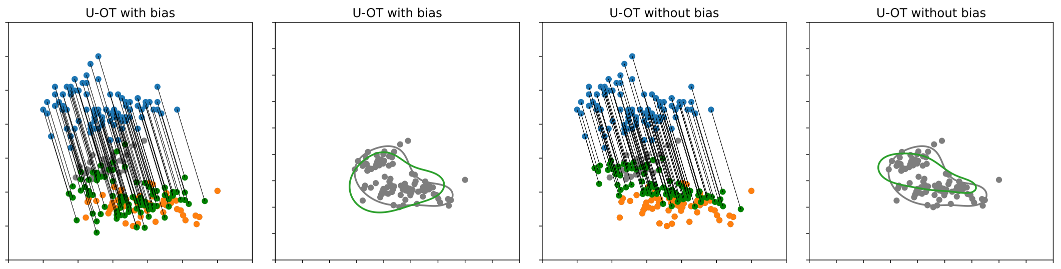
(a) Data and baseline models.



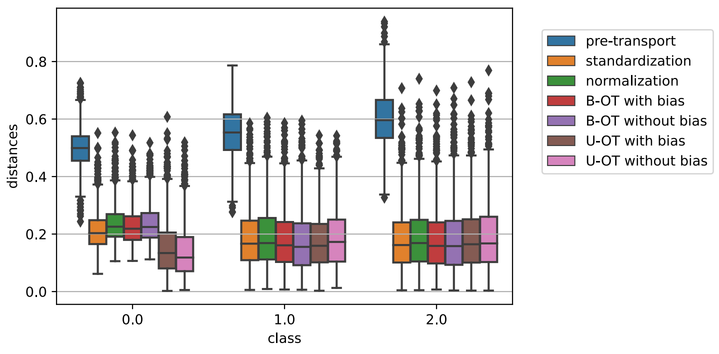
(b) Baseline adaptations.



(c) Balanced OT adaptations.



(d) Unbalanced OT adaptations.



(e) Distances

Figure 78: Rotated 2d Iris dataset

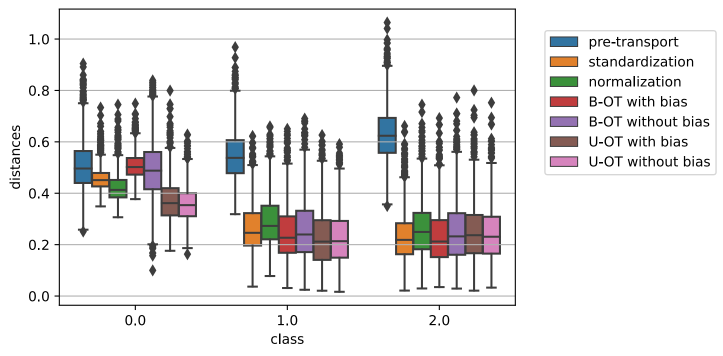


Figure 79: Rotated multidimensional Iris dataset

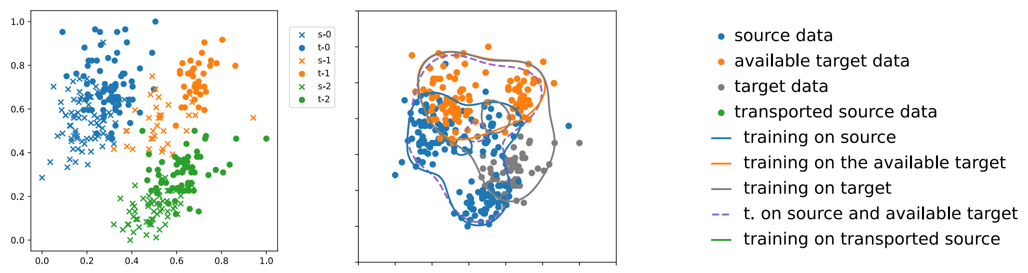
lacking of information in the target dataset (Figure 80). Figure 80e confirms what is visually clear in Figure 80d i.e. only U-OT is able to correctly map the missing class on the target domain.

The same problem has been tested over the dataset containing the full set of features, obtaining even more clear results (Figure 81).

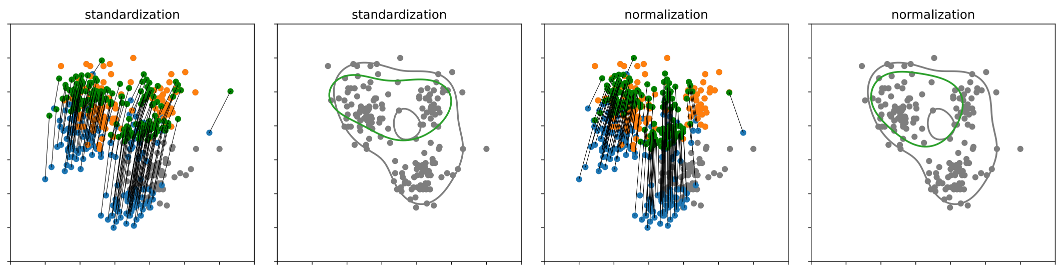
The adaptation strategy has been tested also over real data coming from industry, the primary objective of this work. The first case presented in Figure 82 shows data coming from an experimental campaign where two Multiphase Flow Meters have been tested on about the same working conditions, in particular the two features describe the density and the conductivity of a mixture of oil, gas and water. The two meters are similar as they measure the same quantities but they differ in size and in sensors' manufactures. As previously, the target dataset was undersampled to simulate the lack of knowledge of the whole production parameter space. As expected U-OT is able to recognise the missing information, however U-OT with bias estimation maps some data out of correct region, risking to train a detector with high false negatives, while U-OT without bias estimation maps the missing corner in a slighly wrong way.

The second case showing an industrial adaptation scenario is depicted in Figure 83 where data coming from gas sensors, retrieved in the UCI Machine Learning repository [72, 84], show the measurements of two similar gas sensors exposed to 4 different gases. The assumption made in Figure 83 is that sensors measure the same quantities, but the target has measured less kind of gases. Two configurations were tested: the results of the adaptation on only two features is presented in Figure 83, while the reuslts on the full set of features is shown in Figure 84; on both the configurations the missing class is the fourth. It is easy to see that U-OT is perfectly able to estimate the correct sensor calibration between the two boards.

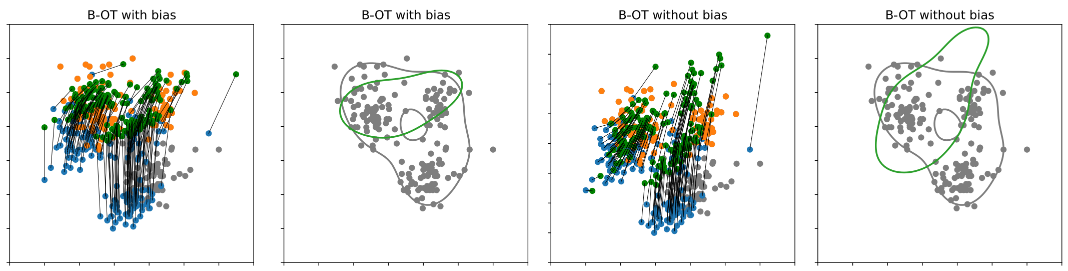
However, looking at the distances between points of corresponding classes (Figures 83e and 84) it is less evident that the unbalanced transport is the best. Even if it closer matches the missing information, the first class is matched slightly worse than using other approaches. Presumably, allowing the model to learn a non linear map, this small dis-alignment would be solved but we leave this experiment as future work as it opens a new set of challenges.



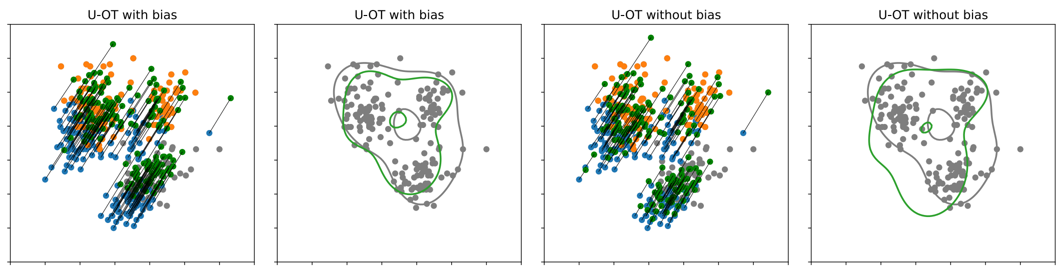
(a) Data and baseline models.



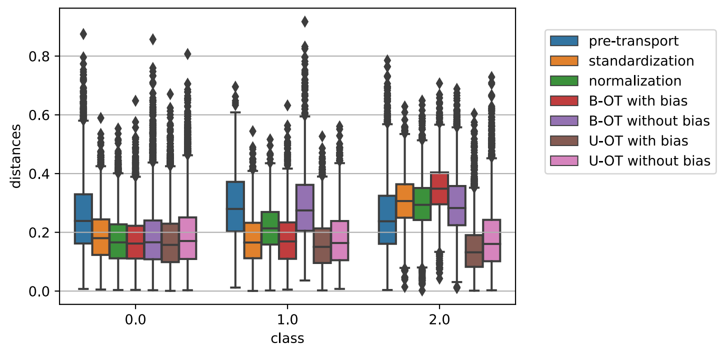
(b) Baseline adaptations.



(c) Balanced OT adaptations.



(d) Unbalanced OT adaptations.



(e) Distances

Figure 80: 2D penguin dataset

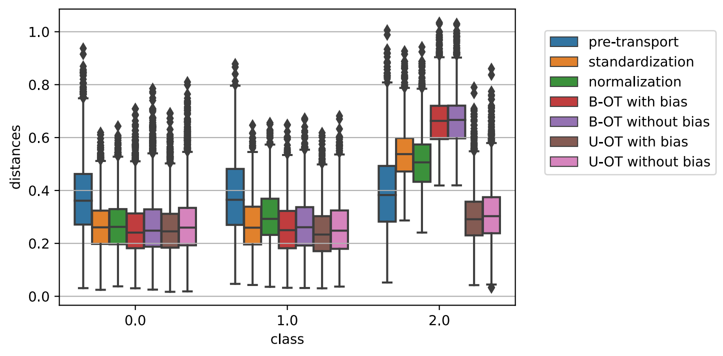
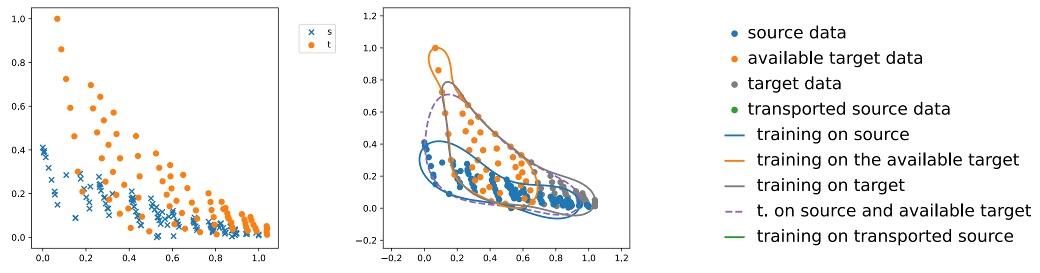


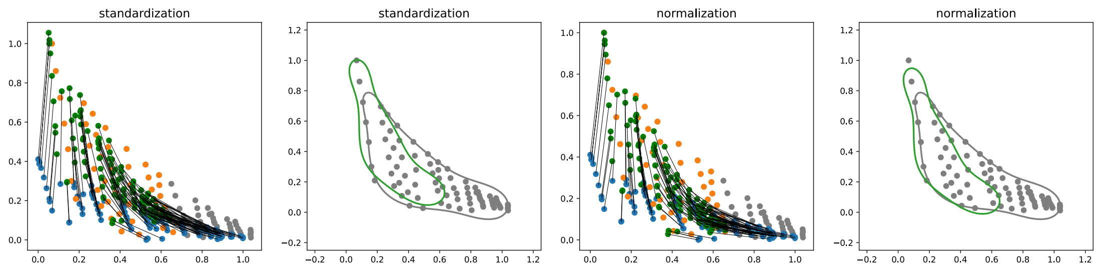
Figure 81: Multidimensional penguin dataset

## 10.5 CONCLUSIONS

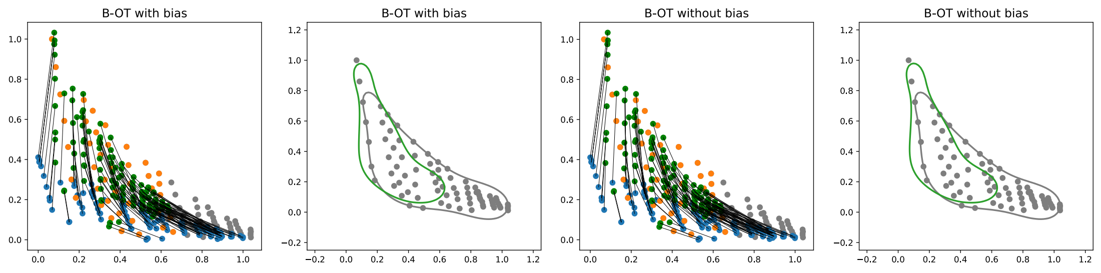
In this Chapter it has been explored the topic of adaptation between two different domains like the measurements coming from two similar but different machines. The goal was to train an anomaly detection model on a machine whose data are too few to train a reliable detector, so the training algorithm relies on previous information coming from a similar machine. The idea was to rely on Optimal Transport tools in order to transfer the knowledge between the better known machine and the new one. However existing approaches are only able to reconstruct a map between the two domains if the two datasets are somehow balanced and contain the same amount of information. To solve this issue it was proposed a new approach able to match the two domains in this more complex unbalanced scenario. As shown in the previous Section, the proposed approach beats both the baseline adaptations and other strategies based on balanced Optimal Transport in most of the tested scenarios. Possible limitations of this approach may arise when the map that links the two domains is too complex or when the similarity between the two domains is too small. Future works will address the possibility to learn nonlinear transformations and the application of this method to classification tasks.



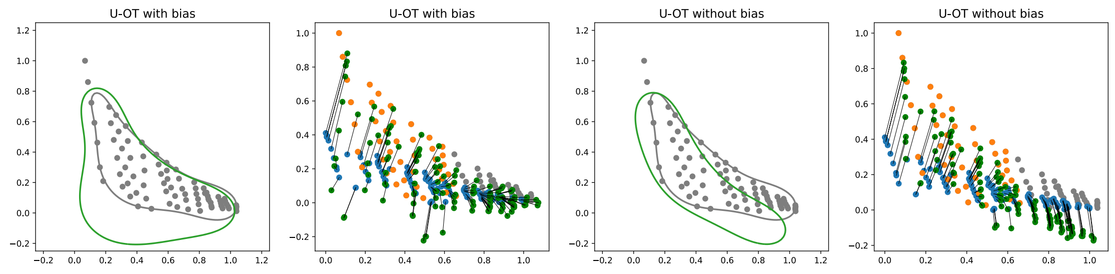
(a) Data and baseline models.



(b) Baseline adaptations.

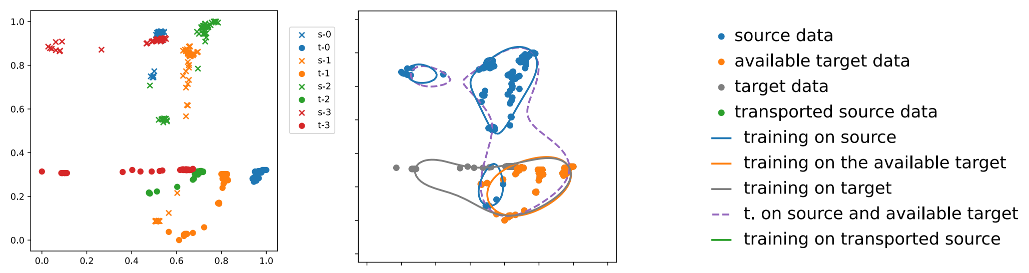


(c) Balanced OT adaptations.

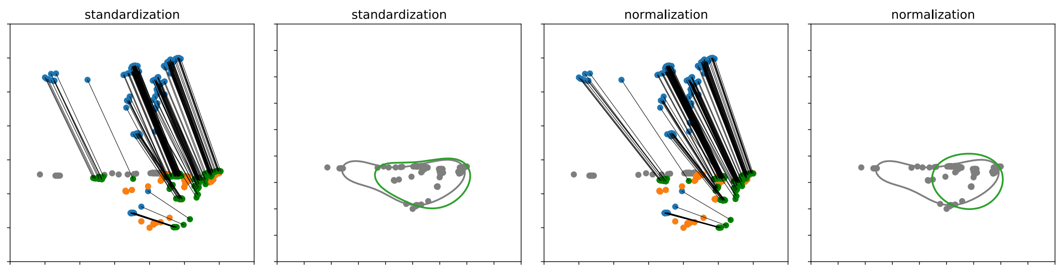


(d) Unbalanced OT adaptations.

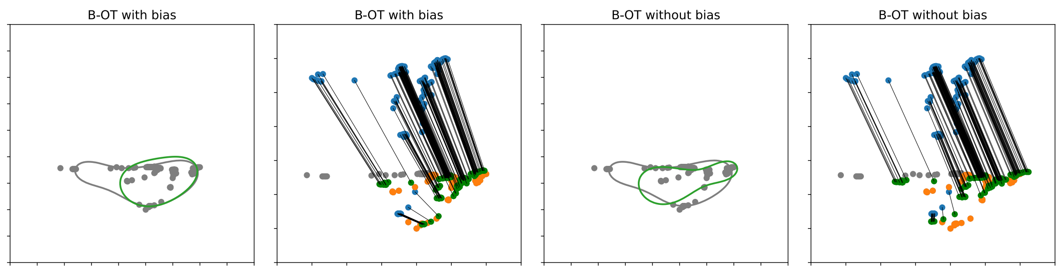
Figure 82: Multiphase flow meter



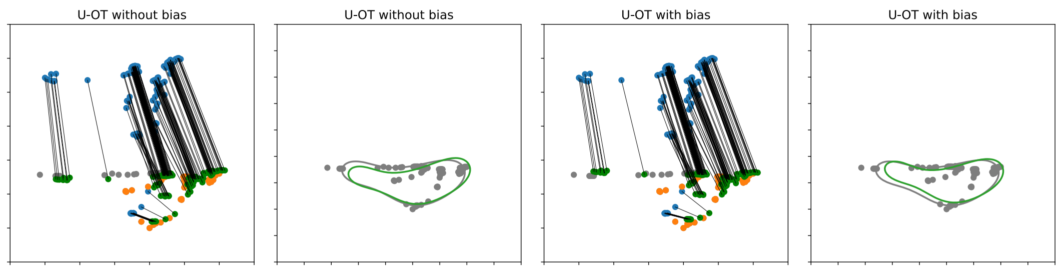
(a) Data and baseline models.



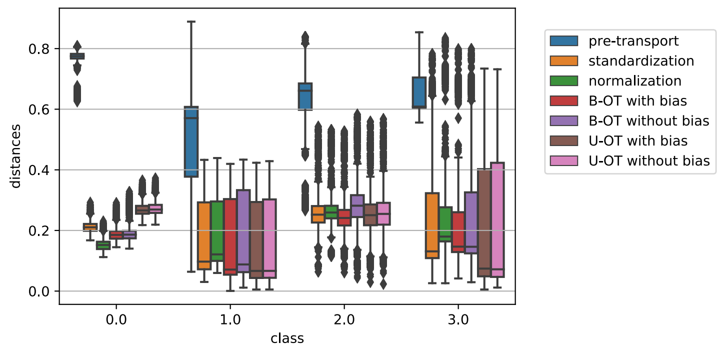
(b) Baseline adaptations.



(c) Balanced OT adaptations.



(d) Unbalanced OT adaptations.



(e) Distances

Figure 83: Gas meter

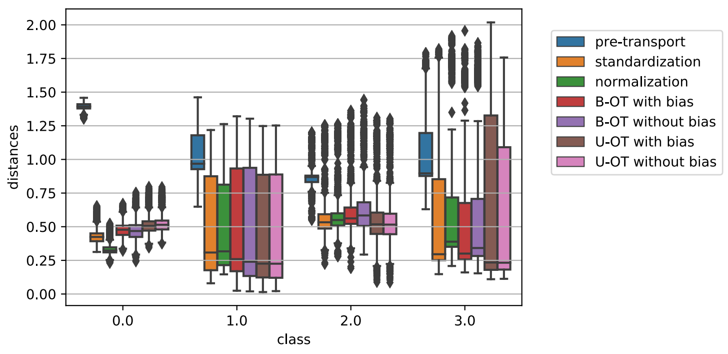


Figure 84: Multidimensional gas dataset

# 11 CONCLUSION

In the present work we have explored and developed new Machine Learning algorithms in order to enhance the reliability of industrial devices. The main use case we were inspired by is the Multiphase Flow Meter, a device able to measure the individual flow rates of oil, water and gas coming from an oil field. This instrument indeed shows many interesting challenges this thesis tried to solve, and that are applicable in many other industrial scenarios. The practical problems posed by the MPFM were described in detail in Chapter 2 and then addressed from a more theoretical point of view in the following Chapters.

Firstly the focus was on Soft Sensing techniques able to provide not only the expected estimate but also to give a confidence interval that measures its uncertainty. Results shown in Chapter 3 were in accordance with domain experts, in particular the estimated uncertainty was able to correlate with the presence of complex flow patterns.

Then, an extensive literature review concerning tree-based algorithms on anomaly detection tasks has been presented in Chapter 4, where the different models have been compared on detection strategy and performance.

In Chapter 5 an anomaly detection framework has been developed in order to apply static algorithms to an always evolving process, and to detect anomalies on the instrument behaviour more than the measured process. In this scenario it was possible to select a light algorithm that accurately looks for possible instrument malfunctions and is cheap to compute and interpret. Future studies should focus on approaches that reduce the dependence of the model on feature alignment and normalization.

Some very recent research areas have been addressed in conjunction with anomaly detection in Chapters 6,7,8,9 and 10.

The problem of computing anomaly detection models on the device has been developed in the context of Tiny Machine Learning in Chapter 6 where a procedure was proposed to compress the most useful information contained in a forest, in order to allow the model to be deployed on a micro controller. Further investigations should measure the robustness of the compression algorithm, potentially exploiting the available labels in a different way.

On the other hand Chapters 7 and 8 presented a way to introduce iteratively limited supervision in an unsupervised model in order to enhance the detection performances and to push the algorithm towards the user expected anomaly definitions. Alternative formulations of the model might improve even more the detection of anomalies and the user satisfaction.

Chapter 9 addressed the problem of explainable anomaly detection, providing a way to get interpretations on why an algorithm highlights a point as a potential anomaly. In this context, we suggest undertaking further research into the development of performance metrics to measure the goodness of proposed interpretations.

Finally, in Chapter 10 the problem of the model scalability between different machines has been addressed by means of Domain Adaptation techniques. In particular the goal was to transfer information between differently sampled machines with the aid of appropriately modified Optimal Transport tools. Further works needs to be done to improve the robustness of these approaches.

It is important to stress that even if most of the algorithms that were developed in this thesis are based on Isolation Forest, they can be easily adapted to other variants of the original algorithm like Extended Isolation Forest, Robust Random Cut Forest or Mondrian Forests. This could help achieve even better detection performance by adding new features to the target model.

As obtaining labels in industrial scenarios can be very expensive, future work should focus on Active Learning strategies to reduce the number of labelled data needed to train Soft Sensing algorithms. We suggest also that further research should be undertaken in the Online Learning and Domain Adaptation areas. The first, to allow online and continuous training on the devices that might be installed remotely but need to adapt to specific working environments, while the second to transfer the solutions between similar products without the need to make a new experimental campaign every time a new product is developed.

# BIBLIOGRAPHY

[1] *2018 reform of EU data protection rules*. European Commission, May 25, 2018. URL: https://ec.europa.eu/commission/sites/beta-political/files/data-protection-factsheet-changes_en.pdf (visited on 06/17/2019).

[2] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. "A review of uncertainty quantification in deep learning: Techniques, applications and challenges." In: *Information Fusion* (2021).

[3] Shikha Agrawal and Jitendra Agrawal. "Survey on anomaly detection using data mining techniques." In: *Procedia Computer Science* 60 (2015), pp. 708–713.

[4] Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. "Unsupervised real-time anomaly detection for streaming data." In: *Neurocomputing* 262 (2017), pp. 134–147.

[5] Subutai Ahmad and Scott Purdy. "Real-time anomaly detection for streaming analytics." In: *arXiv preprint arXiv:1607.02480* (2016).

[6] Saeed Ahmed, YoungDoo Lee, Seung-Ho Hyun, and Insoo Koo. "Unsupervised machine learning-based detection of covert data integrity assault in smart grid networks utilizing isolation forest." In: *IEEE Transactions on Information Forensics and Security* 14.10 (2019), pp. 2765–2777.

[7] ET Alger, J Kroll, EG Dzenitis, R Montesanti, J Hughes, M Swisher, J Taylor, K Segraves, DM Lord, J Reynolds, et al. "NIF target assembly metrology methodology and results." In: *Fusion Science and Technology* 59.1 (2011), pp. 78–86.

[8] Roohallah Alizadehsani, Mohamad Roshanzamir, Sadiq Hussain, Abbas Khosravi, Afsaneh Koohestani, Mohammad Hossein Zangooei, Moloud Abdar, Adham Beykikhoshk, Afshin Shoeibi, Assef Zare, et al. "Handling of uncertainty in medical data using machine learning and probability theory techniques: A review of 30 years (1991–2020)." In: *Annals of Operations Research* (2021), pp. 1–42.

[9] Raed Alsini, Abdullah Almakrab, Ahmed Ibrahim, and Xiaogang Ma. "Improving the outlier detection method in concrete mix design by combining the isolation forest and local outlier factor." In: *Construction and Building Materials* 270 (2021), p. 121396.

[10] André Altmann, Laura Toloşi, Oliver Sander, and Thomas Lengauer. "Permutation importance: a corrected feature importance measure." In: *Bioinformatics* 26.10 (2010), pp. 1340–1347.

[11] A Amin. “Evaluation of commercially available virtual flow meters (VFMs).” In: *Offshore Technology Conference*. OnePetro. 2015.

[12] Ahmad Azharuddin Azhari Mohd Amiruddin, Haslinda Zabiri, Sean Suraj Jeremiah, Weng Kean Teh, and Bashariah Kamaruddin. “Valve stiction detection through improved pattern recognition using neural networks.” In: *Control Engineering Practice* 90 (2019), pp. 63–84.

[13] Fabrizio Angiulli and Clara Pizzuti. “Fast outlier detection in high dimensional spaces.” In: *European conference on principles of data mining and knowledge discovery*. Springer. 2002, pp. 15–27.

[14] Maureen Ani, Gbenga Oluyemi, Andrei Petrovski, and Sina Rezaei-Gomari. “Reservoir uncertainty analysis: The trends from probability to algorithms and machine learning.” In: *SPE Intelligent Energy International Conference and Exhibition*. OnePetro. 2016.

[15] Mattia Antonini, Massimo Vecchio, Fabio Antonelli, Pietro Ducange, and Charith Perera. “Smart audio sensors in the internet of things edge for anomaly detection.” In: *IEEE Access* 6 (2018), pp. 67594–67610.

[16] Vincent Aravantinos and Peter Schlicht. “Making the relationship between uncertainty estimation and safety less uncertain.” In: *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE. 2020, pp. 1139–1144.

[17] Sunil Aryal, KC Santosh, and Richard Dazeley. “usfAD: a robust anomaly detector based on unsupervised stochastic forest.” In: *International Journal of Machine Learning and Cybernetics* (2020), pp. 1–14.

[18] Sunil Aryal, Kai Ming Ting, Jonathan R Wells, and Takashi Washio. “Improving iforest with relative mass.” In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer. 2014, pp. 510–521.

[19] John O Awoyemi, Adebayo O Adetunmbi, and Samuel A Oluwadare. “Credit card fraud detection using machine learning techniques: A comparative analysis.” In: *2017 international conference on computing networking and informatics (ICCNI)*. IEEE. 2017, pp. 1–9.

[20] Babak Bahrami, Sajjad Mohsenpour, Hamid Reza Shamshiri Noghabi, Nassim Hemmati, and Amir Tabzar. “Estimation of flow rates of individual phases in an oil-gas-water multiphase flow system using neural network approach and pressure signal analysis.” In: *Flow Measurement and Instrumentation* 66 (2019), pp. 28–36.

[21] Edward Balaban, Abhinav Saxena, Prasun Bansal, Kai F Goebel, and Simon Curran. “Modeling, detection, and disambiguation of sensor faults for aerospace applications.” In: *IEEE Sensors Journal* 9.12 (2009), pp. 1907–1917.

[22] Colby Banbury, Chuteng Zhou, Igor Fedorov, Ramon Matas, Urmish Thakker, Dibakar Gope, Vijay Janapa Reddi, Matthew Mattina, and Paul Whatmough. "Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers." In: *Proceedings of Machine Learning and Systems* 3 (2021).

[23] Colby R Banbury, Vijay Janapa Reddi, Max Lam, William Fu, Amin Fazel, Jeremy Holleman, Xinyuan Huang, Robert Hurtado, David Kanter, Anton Lokhmotov, et al. "Benchmarking TinyML Systems: Challenges and Direction." In: *SysML 2020, Proceedings*. York. 2020.

[24] Tharindu R Bandaragoda, Kai Ming Ting, David Albrecht, Fei Tony Liu, Ye Zhu, and Jonathan R Wells. "Isolation-based anomaly detection using nearest-neighbor ensembles." In: *Computational Intelligence* 34.4 (2018), pp. 968–998.

[25] Patrick Bangert. "Machine Learning for Multiphase Flow Metering." In: *Machine Learning Applications in Subsurface Energy Resource Management*. CRC Press, pp. 337–352.

[26] Tommaso Barbariol, Filippo Dalla Chiara, Davide Marcato, and Gian Antonio Susto. "A review of tree-based approaches for anomaly detection." In: *Control Charts and Machine Learning for Anomaly Detection in Manufacturing* (2022), pp. 149–185.

[27] Tommaso Barbariol, Enrico Feltresi, and Gian Antonio Susto. "Self-Diagnosis of Multiphase Flow Meters through Machine Learning-Based Anomaly Detection." In: *Energies* 13.12 (2020), p. 3136.

[28] Tommaso Barbariol and Gian Antonio Susto. "TiWS-iForest: Isolation forest in weakly supervised and tiny ML scenarios." In: *Information Sciences* 610 (2022), pp. 126–143.

[29] David Barber. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

[30] Caroline Mattos Barbosa, Cesar Marques Salgado, Luis Eduardo Barreira Brandão, et al. "Study of photon attenuation coefficient in brine using MCNP code." In: (2015).

[31] Alessandro Beghi, Luca Cecchinato, Chiara Corazzol, Mirco Rampazzo, Francesco Simmini, and Gian Antonio Susto. "A one-class svm based tool for machine learning novelty detection in hvac chiller systems." In: *IFAC Proceedings Volumes* 47.3 (2014), pp. 1953–1958.

[32] Giovanni Betta and Antonio Pietrosanto. "Instrument fault detection and isolation: State of the art and new research trends." In: *IEEE Transactions on Instrumentation and Measurement* 49.1 (2000), pp. 100–107.

[33] Timur Bikmukhametov and Johannes Jäschke. "First principles and machine learning virtual flow metering: a literature review." In: *Journal of Petroleum Science and Engineering* 184 (2020), p. 106487.

[34] Leo Breiman. “Random forests.” In: *Machine learning* 45.1 (2001), pp. 5–32.

[35] Christopher Earls Brennen and Christopher E Brennen. *Fundamentals of multiphase flow*. Cambridge university press, 2005.

[36] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. “LOF: identifying density-based local outliers.” In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 2000, pp. 93–104.

[37] Lucas C Brito, Gian Antonio Susto, Jorge N Brito, and Marcus AV Duarte. “An explainable artificial intelligence approach for unsupervised fault detection and diagnosis in rotating machinery.” In: *Mechanical Systems and Signal Processing* 163 (2022), p. 108105.

[38] Sebastian Buschjager, Philipp-Jan Honysz, and Katharina Morik. “Randomized outlier detection with trees.” In: *International Journal of Data Science and Analytics* (2020), pp. 1–14.

[39] Guilherme O Campos, Arthur Zimek, Jörg Sander, Ricardo JGB Campello, Barbora Micenková, Erich Schubert, Ira Assent, and Michael E Houle. “On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study.” In: *Data mining and knowledge discovery* 30 (2016), pp. 891–927.

[40] Domenico Capriglione, Consolatina Liguori, Cesare Pianese, and Antonio Pietrosanto. “On-line sensor fault detection, isolation, and accommodation in automotive engines.” In: *IEEE Transactions on Instrumentation and Measurement* 52.4 (2003), pp. 1182–1189.

[41] Fabrizio Carcillo, Yann-Aël Le Borgne, Olivier Caelen, Yacine Kessaci, Frédéric Oblé, and Gianluca Bontempi. “Combining unsupervised and supervised learning in credit card fraud detection.” In: *Information sciences* 557 (2021), pp. 317–331.

[42] JA Cariño, M Delgado-Prieto, D Zurita, A Picot, JA Ortega, and RJ Romero-Troncoso. “Incremental novelty detection and fault identification scheme applied to a kinematic chain under non-stationary operation.” In: *ISA transactions* (2019).

[43] Mattia Carletti, Chiara Masiero, Alessandro Beghi, and Gian Antonio Susto. “Explainable Machine Learning in Industry 4.0: Evaluating Feature Importance in Anomaly Detection to Enable Root Cause Analysis.” In: *2019 IEEE Conference on Systems, Man, and Cybernetics*. IEEE. 2019.

[44] Mattia Carletti, Chiara Masiero, Alessandro Beghi, and Gian Antonio Susto. “Explainable machine learning in industry 4.0: Evaluating feature importance in anomaly detection to enable root cause analysis.” In: *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. IEEE. 2019, pp. 21–26.

[45] Mattia Carletti, Matteo Terzi, and Gian Antonio Susto. “Interpretable Anomaly Detection with DIFFI: Depth-based feature importance of Isolation Forest.” In: *Engineering Applications of Artificial Intelligence* 119 (2023), p. 105730.

[46] Mattia Carletti, Matteo Terzi, and Gian Antonio Susto. *Interpretable Anomaly Detection with DIFFI: Depth-based Isolation Forest Feature Importance*. 2023.

[47] Thyago P Carvalho, Fabrizzio AAMN Soares, Roberto Vita, Roberto da P Francisco, João P Basto, and Symone GS Alcalá. “A systematic literature review of machine learning methods applied to predictive maintenance.” In: *Computers & Industrial Engineering* 137 (2019), p. 106024.

[48] Laetitia Chapel, Rémi Flamary, Haoran Wu, Cédric Févotte, and Gilles Gasso. “Unbalanced Optimal Transport through Non-negative Penalized Linear Regression.” In: *Advances in Neural Information Processing Systems* 34 (2021).

[49] Fan Chen, Zicheng Liu, and Ming-ting Sun. “Anomaly detection by using random projection forest.” In: *2015 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2015, pp. 1210–1214.

[50] Gang Chen, Yuan Li Cai, and Juan Shi. “Ordinal isolation: An efficient and effective intelligent outlier detection algorithm.” In: *2011 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems*. IEEE. 2011, pp. 21–26.

[51] Zhiwen Chen, Xueming Li, Chao Yang, Tao Peng, Chunhua Yang, HR Karimi, and Weihua Gui. “A data-driven ground fault detection and isolation method for main circuit in railway electrical traction system.” In: *ISA transactions* 87 (2019), pp. 264–271.

[52] Yao Cheng, Zhiwei Wang, Bingyan Chen, Weihua Zhang, and Guanhua Huang. “An improved complementary ensemble empirical mode decomposition with adaptive noise and its application to rolling element bearing fault diagnosis.” In: *ISA transactions* (2019).

[53] Zeki Murat Çınar, Abubakar Abdussalam Nuhu, Qasim Zeeshan, Orhan Korhan, Mohammed Asmael, and Babak Safaei. “Machine learning in predictive maintenance towards sustainable smart manufacturing in industry 4.0.” In: *Sustainability* 12.19 (2020), p. 8211.

[54] Roberto Confalonieri, Ludovik Coba, Benedikt Wagner, and Tarek R Besold. “A historical perspective of explainable Artificial Intelligence.” In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 11.1 (2021), e1391.

[55] Sidsel Corneliussen, Jean-Paul Couput, Eivind Dahl, Eivind Dykesteen, Kjell-Eivind Frøysa, Erik Malde, Håkon Moestue, Paul Ove Moksnes, Lex Scheers, and Hallvard Tunheim. *Handbook of Multiphase Flow Metering*. Norwegian Society for Oil and Gas Measurement. ISBN: 82-91341-89-3. URL: https://nfogm.no/wp-content/uploads/

2014/02/MPFM_Handbook_Revision2_2005_ISBN-82-91341-89-3.pdf.

[56] Claudia Corradino, Gaetana Ganci, Giuseppe Bilotta, Annalisa Cappello, Ciro Del Negro, and Luigi Fortuna. "Smart decision support systems for volcanic applications." In: *Energies* 12.7 (2019), p. 1216.

[57] National Research Council. *Assessing the Reliability of Complex Models: Mathematical and Statistical Foundations of Verification, Validation, and Uncertainty Quantification*. Washington, DC: The National Academies Press, 2012. ISBN: 978-0-309-25634-6. DOI: 10.17226/13395. URL: https://www.nap.edu/catalog/13395/assessing-the-reliability-of-complex-models-mathematical-and-statistical-foundations.

[58] Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. "Optimal Transport for Domain Adaptation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.9 (2017), pp. 1853–1865. DOI: 10.1109/TPAMI.2016.2615921.

[59] Jovani Dalzochio, Rafael Kunst, Edison Pignaton, Alecio Binotto, Srijnan Sanyal, Jose Favilla, and Jorge Barbosa. "Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges." In: *Computers in Industry* 123 (2020), p. 103298.

[60] Mahashweta Das and Srinivasan Parthasarathy. "Anomaly detection and spatio-temporal analysis of global climate system." In: *Proceedings of the third international workshop on knowledge discovery from sensor data*. 2009, pp. 142–150.

[61] Shubhomoy Das, Weng-Keen Wong, Thomas Dietterich, Alan Fern, and Andrew Emmott. "Incorporating expert feedback into active anomaly discovery." In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE. 2016, pp. 853–858.

[62] Shubhomoy Das, Weng-Keen Wong, Alan Fern, Thomas G Dietterich, and Md Amran Siddiqui. "Incorporating feedback into tree-based anomaly detection." In: *arXiv preprint arXiv:1708.09441* (2017).

[63] Akshay J Dave and Annalisa Manera. "Inference of Gas-liquid Flowrate using Neural Networks." In: *arXiv preprint arXiv:2003.08182* (2020).

[64] Chesner Désir, Simon Bernard, Caroline Petitjean, and Laurent Heutte. "One class random forests." In: *Pattern Recognition* 46.12 (2013), pp. 3490–3506.

[65] Charlie Dickens, Eric Meissner, Pablo G Moreno, and Tom Diethe. "Interpretable Anomaly Detection with Mondrian Polya Forests on Data Streams." In: *arXiv preprint arXiv:2008.01505* (2020).

[66] Zhi-Guo Ding, Da-Jun Du, and Min-Rui Fei. "An isolation principle based distributed anomaly detection method in wireless sensor networks." In: *International Journal of Automation and Computing* 12.4 (2015), pp. 402–412.

[67] Zhiguo Ding and Minrui Fei. “An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window.” In: *IFAC Proceedings Volumes* 46.20 (2013), pp. 12–17.

[68] Zhiguo Ding, Minrui Fei, and Dajun Du. “An online anomaly detection method for stream data using isolation principle and statistic histogram.” In: *International Journal of Modeling, Simulation, and Scientific Computing* 6.02 (2015), p. 1550017.

[69] Steve Donoho. “Early detection of insider trading in option markets.” In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2004, pp. 420–429.

[70] Finale Doshi-Velez and Been Kim. “Towards a rigorous science of interpretable machine learning.” In: *arXiv preprint arXiv:1702.08608* (2017).

[71] Jiaxin Du, Guangjie Han, Chuan Lin, and Miguel Martinez-Garcia. “ITrust: An Anomaly-resilient Trust Model Based on Isolation Forest for Underwater Acoustic Sensor Networks.” In: *IEEE Transactions on Mobile Computing* (2020).

[72] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. URL: http://archive.ics.uci.edu/ml.

[73] Ricardo Dunia, S Joe Qin, Thomas F Edgar, and Thomas J McAvoy. “Identification of faulty sensors using principal component analysis.” In: *AIChE Journal* 42.10 (1996), pp. 2797–2812.

[74] Lachit Dutta and Swapna Bharali. “TinyML Meets IoT: A Comprehensive Survey.” In: *Internet of Things* (2021), p. 100461.

[75] David Duvenaud. “Automatic Model Construction with Gaussian Processes.” PhD thesis. Computational and Biological Learning Laboratory, University of Cambridge, 2014.

[76] Carlos Eiras-Franco, David Martinez-Rego, Bertha Guijarro-Berdinas, Amparo Alonso-Betanzos, and Antonio Bahamonde. “Large scale anomaly detection in mixed numerical and categorical input spaces.” In: *Information Sciences* 487 (2019), pp. 115–127.

[77] André Listou Ellefsen, Peihua Han, Xu Cheng, Finn Tore Holmeset, Vilmar Æsøy, and Houxiang Zhang. “Online fault detection in autonomous ferries: Using fault-type independent spectral anomaly detection.” In: *IEEE Transactions on instrumentation and measurement* 69.10 (2020), pp. 8216–8225.

[78] Gioia Falcone, Geoffrey Hewitt, and Claudio Alimonti. *Multiphase flow metering: principles and applications*. Vol. 54. Elsevier, 2009.

[79] Gioia Falcone, GF Hewitt, C Alimonti, B Harrison, et al. “Multiphase flow metering: current trends and future developments.” In: *SPE annual technical conference and exhibition*. Society of Petroleum Engineers. 2001.

[80] Shiwei Fan and Tinghu Yan. “Two-phase air–water slug flow measurement in horizontal pipe using conductance probes and neural network.” In: *IEEE Transactions on Instrumentation and Measurement* 63.2 (2013), pp. 456–466.

[81] Tom Fawcett and Foster Provost. “Activity monitoring: Noticing interesting changes in behavior.” In: *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. 1999, pp. 53–62.

[82] Peter A Flach and Meelis Kull. “Precision-Recall-Gain Curves: PR Analysis Done Right.” In: *NIPS*. Vol. 15. 2015.

[83] Rémi Flamary, Nicolas Courty, Alain Rakotomamonjy, and Devis Tuia. “Optimal transport with Laplacian regularization.” In: *NIPS 2014, Workshop on Optimal Transport and Machine Learning*. Montréal, Canada, Dec. 2014. URL: https://hal.archives-ouvertes.fr/hal-01103076.

[84] J Fonollosa, Luis Fernandez, Agustin Gutierrez-Galvez, Ramon Huerta, and Santiago Marco. “Calibration transfer and drift counteraction in chemical sensor arrays using Direct Standardization.” In: *Sensors and Actuators B: Chemical* 236 (2016), pp. 1044–1053.

[85] Ralph Foorthuis. “On the nature and types of anomalies: A review of deviations in data.” In: *International Journal of Data Science and Analytics* 12.4 (2021), pp. 297–331.

[86] Fiorenzo Franceschini, Maurizio Galetto, and Gianfranco Genta. “Multivariate control charts for monitoring internal camera parameters in digital photogrammetry for LSDM (Large-Scale Dimensional Metrology) applications.” In: *Precision Engineering* 42 (2015), pp. 133–142.

[87] Marguerite Frank and Philip Wolfe. “An algorithm for quadratic programming.” In: *Naval Research Logistics Quarterly* 3 (1956), pp. 95–110.

[88] Luca Frau, Gian Antonio Susto, Tommaso Barbariol, and Enrico Feltresi. “Uncertainty Estimation for Machine Learning Models in Multiphase Flow Applications.” In: *Informatics*. Vol. 8. 3. MDPI. 2021, p. 58.

[89] Rina Friedberg, Julie Tibshirani, Susan Athey, and Stefan Wager. “Local linear forests.” In: *Journal of Computational and Graphical Statistics* (2020), pp. 1–15.

[90] Daniel Fryer, Inga Strümke, and Hien Nguyen. “Shapley values for feature selection: the good, the bad, and the axioms.” In: *IEEE Access* 9 (2021), pp. 144352–144360.

[91] Yarin Gal and Zoubin Ghahramani. “Dropout as a bayesian approximation: Representing model uncertainty in deep learning.” In: *international conference on machine learning*. PMLR. 2016, pp. 1050–1059.

[92] Rongfang Gao, Tiantian Zhang, Shaohua Sun, and Zhanyu Liu. "Research and Improvement of Isolation Forest in Detection of Local Anomaly Points." In: *Journal of Physics: Conference Series*. Vol. 1237. 5. IOP Publishing. 2019, p. 052023.

[93] Suryakant Gautam, Prakash K Tamboli, Kallol Roy, Vaibhav H Patankar, and Siddhartha P Duttagupta. "Sensors incipient fault detection and isolation of nuclear power plant using extended Kalman filter and Kullback–Leibler divergence." In: *ISA transactions* (2019).

[94] Natalie Gentner, Mattia Carletti, Andreas Kyek, Gian Antonio Susto, and Yao Yang. "DBAM: Making virtual metrology/soft sensing with time series data scalable through deep learning." In: *Control Engineering Practice* 116 (2021), p. 104914.

[95] Alia Ghaddar, Lama Darwish, and Fadi Yamout. "Identifying Mass-based local anomalies using Binary Space Partitioning." In: *2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE. 2019, pp. 183–190.

[96] Seyed Hossein Ghafarian and Hadi Sadoghi Yazdi. "Functional gradient approach to probabilistic minimax active learning." In: *Engineering Applications of Artificial Intelligence* 85 (2019), pp. 21–32.

[97] Sushmito Ghosh and Douglas L Reilly. "Credit card fraud detection with a neural-network." In: *System Sciences, 1994. Proceedings of the Twenty-Seventh Hawaii International Conference on*. Vol. 3. IEEE. 1994, pp. 621–630.

[98] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. "Explaining explanations: An overview of interpretability of machine learning." In: *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*. IEEE. 2018, pp. 80–89.

[99] Nicolas Goix, Nicolas Drougard, Romain Brault, and Mael Chiapino. "One class splitting criteria for random forests." In: *Asian Conference on Machine Learning*. PMLR. 2017, pp. 343–358.

[100] Markus Goldstein and Andreas Dengel. "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm." In: *KI-2012: poster and demo track* 9 (2012).

[101] Parikshit Gopalan, Vatsal Sharan, and Udi Wieder. "Pidforest: anomaly detection via partial identification." In: *Advances in Neural Information Processing Systems* 32 (2019).

[102] Sudipto Guha, Nina Mishra, Gourav Roy, and Okke Schrijvers. "Robust random cut forest based anomaly detection on streams." In: *International conference on machine learning*. PMLR. 2016, pp. 2712–2721.

[103] Morteza Hadipour, Javad Farrokhi Derakhshandeh, and Mohsen Aghazadeh Shiran. "An experimental setup of multi-intelligent control system (MICS) of water management using the Internet of Things (IoT)." In: *ISA transactions* (2019).

[104] A Hall, D Griffin, and R Steven. “A discussion on wet gas flow parameter definitions.” In: *26th International North Sea Flow Meter Workshop, organized by the National Engineering Laboratory, East Kilbride, Scotland, UK*. 2007.

[105] Lærke Skov Hansen, Simon Pedersen, and Petar Durdevic. “Multiphase flow metering in offshore oil and gas transportation pipelines: Trends and perspectives.” In: *Sensors* 19.9 (2019), p. 2184.

[106] Yuki Hara, Yoshikazu Fukuyama, Kenya Murakami, Tatsuya Iizaka, and Tetsuro Matsui. “Fault Detection of Hydroelectric Generators using Isolation Forest.” In: *2020 59th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*. IEEE. 2020, pp. 864–869.

[107] Sahand Hariri, Matias Carrasco Kind, and Robert J Brunner. “Extended isolation forest.” In: *IEEE Transactions on Knowledge and Data Engineering* (2019).

[108] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[109] Douglas M Hawkins. *Identification of outliers*. Vol. 11. Springer, 1980.

[110] David J Hill and Barbara S Minsker. “Anomaly detection in streaming environmental sensor data: A data-driven modeling approach.” In: *Environmental Modelling & Software* 25.9 (2010), pp. 1014–1022.

[111] Tin Kam Ho. “Random decision forests.” In: *Proceedings of 3rd international conference on document analysis and recognition*. Vol. 1. IEEE. 1995, pp. 278–282.

[112] Jordan Hochenbaum, Owen S Vallis, and Arun Kejariwal. “Automatic anomaly detection in the cloud via statistical learning.” In: *arXiv preprint arXiv:1704.07706* (2017).

[113] Julia Hofmockel and Eric Sax. “Isolation Forest for Anomaly Detection in Raw Vehicle Sensor Data.” In: *VEHITS*. 2018, pp. 411–416.

[114] Viktor Holmér. *Hybrid Extended Isolation Forest: Anomaly Detection for Bird Alarm*. 2019.

[115] Allison Marie Horst, Alison Presmanes Hill, and Kristen B Gorman. *palmerpenguins: Palmer Archipelago (Antarctica) penguin data*. R package version 0.1.0. 2020. DOI: 10.5281/zenodo.3960218. URL: https://allisonhorst.github.io/palmerpenguins/.

[116] Essam H Houssein, Marwa M Emam, Abdelmgeid A Ali, and Ponnuthurai Nagaratnam Suganthan. “Deep and machine learning techniques for medical imaging-based breast cancer: A comprehensive review.” In: *Expert Systems with Applications* 167 (2021), p. 114161.

[117] Delin Hu, Jinku Li, Yinyan Liu, and Yi Li. “Flow Adversarial Networks: Flowrate Prediction for Gas–Liquid Multiphase Flows Across Different Domains.” In: *IEEE transactions on neural networks and learning systems* 31.2 (2019), pp. 475–487.

[118] Aleks Huč, Jakob Šalej, and Mira Trebar. “Analysis of Machine Learning Algorithms for Anomaly Detection on Edge Devices.” In: *Sensors* 21.14 (2021), p. 4946.

[119] Tsuyoshi Idé, Spiros Papadimitriou, and Michail Vlachos. “Computing correlation anomaly scores using stochastic nearest neighbors.” In: *Seventh IEEE international conference on data mining (ICDM 2007)*. IEEE. 2007, pp. 523–528.

[120] Boris Iglewicz and David Caster Hoaglin. *How to detect and handle outliers*. Vol. 16. Asq Press, 1993.

[121] Giulio Imbalzano, Yongbin Zhuang, Venkat Kapil, Kevin Rossi, Edgar A Engel, Federico Grasselli, and Michele Ceriotti. “Uncertainty estimation for molecular dynamics and sampling.” In: *The Journal of Chemical Physics* 154.7 (2021), p. 074102.

[122] Raihan Ul Islam, Mohammad Shahadat Hossain, and Karl Andersson. “A novel anomaly detection algorithm for sensor data under uncertainty.” In: *Soft Computing* 22.5 (2018), pp. 1623–1639.

[123] Idris Ismail, JC Gamio, SF Ahmed Bukhari, and WQ Yang. “Tomography for multi-phase flow measurement in the oil industry.” In: *Flow measurement and instrumentation* 16.2-3 (2005), pp. 145–155.

[124] Hongquan Ji, Xiao He, Jun Shang, and Donghua Zhou. “Incipient fault detection with smoothing techniques in statistical process monitoring.” In: *Control Engineering Practice* 62 (2017), pp. 11–21.

[125] Sheng-yi Jiang and Qing-bo An. “Clustering-based outlier detection method.” In: *2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery*. Vol. 2. IEEE. 2008, pp. 429–433.

[126] Hyder John and Sameena Naaz. “Credit card fraud detection using local outlier factor and isolation forest.” In: *Int. J. Comput. Sci. Eng.* 7.4 (2019), pp. 1060–1064.

[127] Daniel Jung, Kok Yew Ng, Erik Frisk, and Mattias Krysander. “Combining model-based diagnosis and data-driven anomaly classifiers for fault isolation.” In: *Control Engineering Practice* 80 (2018), pp. 146–156.

[128] Mohammad Al-Kadem, Mohammed Gomaa, Karam Al Yateem, and Abdulmonam Al Maghlouth. “Multiphase Flowmeter Health Monitoring Strategy: Maximizing the Value of Real-Time Sensors and Automation for Industrial Revolution 4.0.” In: *SPE Production & Operations* 37.03 (2022), pp. 533–542.

[129] Pawel Karczmarek, Adam Kiersztyn, and Witold Pedrycz. “Fuzzy set-based isolation forest.” In: *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE. 2020, pp. 1–6.

[130] Pawel Karczmarek, Adam Kiersztyn, and Witold Pedrycz. “n-ary Isolation Forest: An Experimental Comparative Analysis.” In: *International Conference on Artificial Intelligence and Soft Computing*. Springer. 2020, pp. 188–198.

[131] Paweł Karczmarek, Adam Kiersztyn, Witold Pedrycz, and Ebru Al. “K-Means-based isolation forest.” In: *Knowledge-Based Systems* 195 (2020), p. 105659.

[132] Peter GW Keen. “Decision support systems: a research perspective.” In: *Decision support systems: Issues and challenges: Proceedings of an international task force meeting*. 1980, pp. 23–44.

[133] Eamonn Keogh and Chotirat Ann Ratanamahatana. “Exact indexing of dynamic time warping.” In: *Knowledge and information systems* 7.3 (2005), pp. 358–386.

[134] Mohammad Rasheed Khan, Zeeshan Tariq, and Abdulazeez Abdulraheem. “Application of artificial intelligence to estimate oil flow rate in gas-lift wells.” In: *Natural Resources Research* 29.6 (2020), pp. 4017–4029.

[135] Dohyung Kim, Hyochang Yang, Minki Chung, Sungzoon Cho, Huijung Kim, Minhee Kim, Kyungwon Kim, and Eunseok Kim. “Squeezed convolutional variational autoencoder for unsupervised anomaly detection in edge device industrial internet of things.” In: *2018 international conference on information and computer technologies (icict)*. IEEE. 2018, pp. 67–71.

[136] Jonghoon Kim, Hariharan Naganathan, Soo-Young Moon, Wai KO Chong, and Samuel T Ariaratnam. “Applications of clustering and isolation forest techniques in real-time building energy-consumption data: Application to LEED certified buildings.” In: *Journal of Energy Engineering* 143.5 (2017), p. 04017052.

[137] Ahmad F Klaib, Nawaf O Alsrehin, Wasen Y Melhem, Haneen O Bashtawi, and Aws A Magableh. “Eye tracking algorithms, techniques, tools, and applications with an emphasis on machine learning and Internet of Things technologies.” In: *Expert Systems with Applications* 166 (2021), p. 114037.

[138] Edwin M Knorr and Raymond T Ng. “Algorithms for mining distance-based outliers in large datasets.” In: *VLDB*. Vol. 98. Citeseer. 1998, pp. 392–403.

[139] Martin Kopp, Tomáš Pevny, and Martin Holeňa. “Anomaly explanation with random forests.” In: *Expert Systems with Applications* 149 (2020), p. 113187.

[140] Konstantina Kourou, Themis P Exarchos, Konstantinos P Exarchos, Michalis V Karamouzis, and Dimitrios I Fotiadis. “Machine learning applications in cancer prognosis and prediction.” In: *Computational and structural biotechnology journal* 13 (2015), pp. 8–17.

[141] Farinaz Koushanfar, Miodrag Potkonjak, and Alberto Sangiovanni-Vincentelli. “On-line fault detection of sensor measurements.” In: *SENSORS, 2003 IEEE*. Vol. 2. IEEE. 2003, pp. 974–979.

[142] Manfred Krafft, Laszlo Sajtos, and Michael Haenlein. “Challenges and opportunities for marketing scholars in times of the fourth industrial revolution.” In: *Journal of Interactive Marketing* 51.1 (2020), pp. 1–8.

[143] Matjaž Kukar, Petar Vračar, Domen Košir, Darko Pevec, Zoran Bosnić, et al. “AgroDSS: A decision support system for agriculture and farming.” In: *Computers and Electronics in Agriculture* 161 (2019), pp. 260–271.

[144] Punit Kumar and Atul Gupta. “Active learning query strategies for classification, regression, and clustering: a survey.” In: *Journal of Computer Science and Technology* 35.4 (2020), pp. 913–945.

[145] Yu-Hsuan Kuo, Zhenhui Li, and Daniel Kifer. “Detecting Outliers in Data with Correlated Measures.” In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM. 2018, pp. 287–296.

[146] Miron B Kursa and Witold R Rudnicki. “Feature selection with the Boruta package.” In: *Journal of statistical software* 36 (2010), pp. 1–13.

[147] Philip A Legg. “Human-machine decision support systems for insider threat detection.” In: *Data Analytics and Decision Support for Cybersecurity*. Springer, 2017, pp. 33–53.

[148] Pirmin Lemberger and Ivan Panico. “A primer on domain adaptation.” In: *arXiv preprint arXiv:2001.09994* (2020).

[149] Julien Lesouple, Cédric Baudoin, Marc Spigai, and Jean-Yves Tourneret. “How to introduce expert feedback in one-class support vector machines for anomaly detection?” In: *Signal Processing* 188 (2021), p. 108197.

[150] Julien Lesouple and Jean-Yves Tourneret. “Incorporating User Feedback Into One-Class Support Vector Machines for Anomaly Detection.” In: *2020 28th European Signal Processing Conference (EUSIPCO)*. IEEE. 2021, pp. 1608–1612.

[151] Filippo Leveni, Luca Magri, Giacomo Boracchi, and Cesare Alippi. “PIF: Anomaly detection via preference embedding.” In: ().

[152] Changgen Li, Liang Guo, Hongli Gao, and Yi Li. “Similarity-Measured Isolation Forest: Anomaly Detection Method for Machine Monitoring Data.” In: *IEEE Transactions on Instrumentation and Measurement* 70 (2021), pp. 1–12.

[153] Shutao Li, Kunzhong Zhang, Puhong Duan, and Xudong Kang. "Hyperspectral anomaly detection with kernel isolation forest." In: *IEEE Transactions on Geoscience and Remote Sensing* 58.1 (2019), pp. 319–329.

[154] You Li, Zhouzheng Gao, Zhe He, Yuan Zhuang, Ahmed Radi, Ruizhi Chen, and Naser El-Sheimy. "Wireless fingerprinting uncertainty prediction based on machine learning." In: *Sensors* 19.2 (2019), p. 324.

[155] Liefa Liao and Bin Luo. "Entropy isolation forest based on dimension entropy for anomaly detection." In: *International Symposium on Intelligence Computation and Applications*. Springer. 2018, pp. 365–376.

[156] Zi Lin, Xiaolei Liu, and Maurizio Collu. "Wind power prediction based on high-frequency SCADA data along with isolation forest and deep learning neural networks." In: *International Journal of Electrical Power & Energy Systems* 118 (2020), p. 105835.

[157] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. "Explainable ai: A review of machine learning interpretability methods." In: *Entropy* 23.1 (2020), p. 18.

[158] Datong Liu, Jingyue Pang, Ge Song, Wei Xie, Yu Peng, and Xiyuan Peng. "Fragment anomaly detection with prediction and statistical analysis for satellite telemetry." In: *IEEE Access* 5 (2017), pp. 19269–19281.

[159] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. "Isolation Forest." In: *2008 Eighth IEEE International Conference on Data Mining*. 2008, pp. 413–422. DOI: 10.1109/ICDM.2008.17.

[160] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. "Isolation forest." In: *2008 eighth ieee international conference on data mining*. IEEE. 2008, pp. 413–422.

[161] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. "On detecting clustered anomalies using SCiForest." In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2010, pp. 274–290.

[162] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. "Isolation-based anomaly detection." In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 6.1 (2012), pp. 1–39.

[163] Jeremiah Liu, John Paisley, Marianthi-Anna Kioumourtzoglou, and Brent Coull. "Accurate uncertainty estimation and decomposition in ensemble learning." In: *Advances in neural information processing systems* 32 (2019).

[164] Jie Liu, Jianlin Guo, Philip Orlik, Masahiko Shibata, Daiki Nakahara, Satoshi Mii, and Martin Takáč. "Anomaly detection in manufacturing systems using structured neural networks." In: *2018 13th World Congress on Intelligent Control and Automation (WCICA)*. IEEE. 2018, pp. 175–180.

[165] Wenqian Liu, Jingsha He, Song Han, Fangbo Cai, Zhenning Yang, and Nafei Zhu. "A Method for the Detection of Fake Reviews Based on Temporal Features of Reviews and Comments." In: *IEEE Engineering Management Review* 47.4 (2019), pp. 67–79.

[166] Yang Liu, Guoping Yang, Shaojie Qiao, Meiqi Liu, Lulu Qu, Nan Han, Tao Wu, Guan Yuan, and Yuzhong Peng. "Imbalanced data classification: Using transfer learning and active sampling." In: *Engineering Applications of Artificial Intelligence* 117 (2023), p. 105621.

[167] Zhen Liu, Xin Liu, Jin Ma, and Hui Gao. "An optimized computational framework for isolation forest." In: *Mathematical Problems in Engineering* 2018 (2018).

[168] Bo Lu, John Stuber, and Thomas F Edgar. "Integrated online virtual metrology and fault detection in plasma etch tools." In: *Industrial & Engineering Chemistry Research* 53.13 (2013), pp. 5172–5181.

[169] Scott M Lundberg and Su-In Lee. "A Unified Approach to Interpreting Model Predictions." In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Curran Associates, Inc., 2017, pp. 4765–4774.

[170] Simin Luo, Le Luan, Yiping Cui, Xueyan Chai, Zhuzhu Wang, and Yiming Kong. "An Attribute Associated Isolation Forest Algorithm for Detecting Anomalous Electro-data." In: *2019 Chinese Control Conference (CCC)*. IEEE. 2019, pp. 3788–3792.

[171] Yanxia Lyu, Wenjie Li, Yue Wang, Siqi Sun, and Cuirong Wang. "RMHSForest: Relative Mass and Half-Space Tree Based Forest for Anomaly Detection." In: *Chinese Journal of Electronics* 29.6 (2020), pp. 1093–1101.

[172] Haoran Ma, Benyamin Ghojogh, Maria N Samad, Dongyu Zheng, and Mark Crowley. "Isolation Mondrian forest for batch and online anomaly detection." In: *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2020, pp. 3051–3058.

[173] Marco Maggipinto, Alessandro Beghi, and Gian Antonio Susto. "A Deep Learning-based Approach to Anomaly Detection with 2-Dimensional Data in Manufacturing." In: *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*. Vol. 1. IEEE. 2019, pp. 187–192.

[174] Konstantin L Malanchev, Alina A Volnova, Matwey V Kornilov, Maria V Pruzhinskaya, Emille EO Ishida, Florian Mondon, and Vladimir S Korolev. "Use of Machine Learning for Anomaly Detection Problem in Large Astronomical Databases." In: *DAMDID/RCDL*. 2019, pp. 205–216.

[175] Wei Mao, Xiu Cao, Tong Yan, Yongkang Zhang, et al. "Anomaly detection for power consumption data based on isolated forest." In: *2018 International Conference on Power System Technology (POWERCON)*. IEEE. 2018, pp. 4169–4174.

[176] *Market Report: Oil 2021*. `https://www.iea.org/reports/oil-2021`.

[177] Pierre-François Marteau, Saeid Soheily-Khah, and Nicolas Béchet. "Hybrid Isolation Forest-Application to Intrusion Detection." In: *arXiv preprint arXiv:1705.03800* (2017).

[178] Vimala Mathew, Tom Toby, Vikram Singh, B Maheswara Rao, and M Goutham Kumar. "Prediction of Remaining Useful Lifetime (RUL) of turbofan engine using machine learning." In: *2017 IEEE International Conference on Circuits and Systems (ICCS)*. IEEE. 2017, pp. 306–311.

[179] Lorenzo Meneghetti, Matteo Terzi, Simone Del Favero, Gian Antonio Susto, and Claudio Cobelli. "Data-Driven Anomaly Recognition for Unsupervised Model-Free Fault Detection in Artificial Pancreas." In: *IEEE Transactions on Control Systems Technology* (2018).

[180] Antonella Mensi and Manuele Bicego. "A novel anomaly score for isolation forests." In: *International Conference on Image Analysis and Processing*. Springer. 2019, pp. 152–163.

[181] Antonella Mensi and Manuele Bicego. "Enhanced anomaly scores for isolation forests." In: *Pattern Recognition* 120 (2021), p. 108115.

[182] Antonella Mensi, David MJ Tax, and Manuele Bicego. "Detecting Outliers from Pairwise Proximities: Proximity Isolation Forests." In: *Pattern Recognition* (2023), p. 109334.

[183] Pooyan Mobtahej, Xulong Zhang, Maryam Hamidi, and Jing Zhang. "Deep Learning-based Anomaly Detection for Compressors Using Audio Data." In: *2021 Annual Reliability and Maintainability Symposium (RAMS)*. IEEE. 2021, pp. 1–7.

[184] Elmy Johana Mohamad, RA Rahim, Mohd Hafiz Fazalul Rahiman, HLM Ameran, SZM Muji, and OMF Marwah. "Measurement and analysis of water/oil multiphase flow using Electrical Capacitance Tomography sensor." In: *Flow Measurement and Instrumentation* 47 (2016), pp. 62–70.

[185] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[186] Florian Mouret, Mohanad Albughdadi, Sylvie Duthoit, Denis Kouamé, Guillaume Rieu, and Jean-Yves-Tourneret. "Reconstruction of Sentinel-2 Derived Time Series Using Robust Gaussian Mixture Models — Application to the Detection of Anomalous Crop Development." In: *Computers and Electronics in Agriculture* 198 (2022), p. 106983.

[187] Felix Musil, Michael J Willatt, Mikhail A Langovoy, and Michele Ceriotti. "Fast and accurate uncertainty estimation in chemical machine learning." In: *Journal of chemical theory and computation* 15.2 (2019), pp. 906–915.

[188] TUV NEL. "Well Testing-An Evaluation of Test Separators and Multiphase Flow Meters." In: *Well Testing* 26 (2010), 29th.

[189] Roar Nybø and Dan Sui. "Closing the integration gap for the next generation of drilling decision support systems." In: *SPE Intelligent Energy Conference & Exhibition*. OnePetro. 2014, pp. 497–506.

[190] Colin O'Reilly, Alexander Gluhak, Muhammad Ali Imran, and Sutharshan Rajasegarar. "Anomaly detection in wireless sensor networks in a non-stationary environment." In: *IEEE Communications Surveys & Tutorials* 16.3 (2014), pp. 1413–1432.

[191] Cheong Hee Park and Jiil Kim. "An explainable outlier detection method using region-partition trees." In: *The Journal of Supercomputing* 77.3 (2021), pp. 3062–3076.

[192] Michaël Perrot, Nicolas Courty, Rémi Flamary, and Amaury Habrard. "Mapping estimation for discrete optimal transport." In: *Advances in Neural Information Processing Systems* 29 (2016).

[193] Michaël Perrot, Nicolas Courty, Rémi Flamary, and Amaury Habrard. "Mapping Estimation for Discrete Optimal Transport Supplementary Material." In: ().

[194] Tomáš Pevny. "Loda: Lightweight on-line detector of anomalies." In: *Machine Learning* 102.2 (2016), pp. 275–304.

[195] PietroFiorentini. *(access October 13th, 2019), MPFM FloWatch HS datasheet*. Pietro Fiorentini, 2011. URL: https://www.fiorentini.com/ww/en/product/components/mpfm_eng/flowatchhs/.

[196] Boshra Pishgoo, Ahmad Akbari Azirani, and Bijan Raahemi. "A hybrid distributed batch-stream processing approach for anomaly detection." In: *Information Sciences* 543 (2021), pp. 309–327.

[197] *ProLabNL*. http://www.prolabnl.com.

[198] Luca Puggini and Seán McLoone. "Feature selection for anomaly detection using optical emission spectroscopy." In: *IFAC-PapersOnLine* 49.5 (2016), pp. 132–137.

[199] Luca Puggini and Seán McLoone. "An enhanced variable selection and Isolation Forest based methodology for anomaly detection with OES data." In: *Engineering Applications of Artificial Intelligence* 67 (2018), pp. 126–135.

[200] Hongchun Qu, Zonglan Li, and Jingjing Wu. "Integrated Learning Method for Anomaly Detection Combining KLSH and Isolation Principles." In: *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE. 2020, pp. 1–6.

[201] Tareq Aziz AL-Qutami, Rosdiazli Ibrahim, Idris Ismail, and Mohd Azmin Ishak. "Radial basis function network to predict gas flow rate in multiphase flow." In: *Proceedings of the 9th International Conference on Machine Learning and Computing*. 2017, pp. 141–146.

[202] G Madhukar Rao and Dharavath Ramesh. "A Hybrid and Improved Isolation Forest Algorithm for Anomaly Detection." In: *Proceedings of International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications*. Springer. 2021, pp. 589–598.

[203] Carl Edward Rasmussen. "Gaussian processes in machine learning." In: *Summer school on machine learning*. Springer. 2003, pp. 63–71.

[204] Shebuti Rayana. *ODDS Library*. 2016. URL: http://odds.cs.stonybrook.edu.

[205] Haoyu Ren, Darko Anicic, and Thomas A Runkler. "Tinyol: Tinyml with online-learning on microcontrollers." In: *2021 international joint conference on neural networks (IJCNN)*. IEEE. 2021, pp. 1–8.

[206] Mohammad Riazi, Osmar Zaiane, Tomoharu Takeuchi, Anthony Maltais, Johannes Günther, and Micheal Lipsett. "Detecting the onset of machine failure using anomaly detection methods." In: *International Conference on Big Data Analytics and Knowledge Discovery*. Springer. 2019, pp. 3–12.

[207] Tita Ristanto and Roland Horne. "Machine learning applied to multiphase production problems." In: *MS thesis* (2018).

[208] GH Roshani, E Nazemi, and MM Roshani. "Intelligent recognition of gas-oil-water three-phase flow regime and determination of volume fraction using radial basis function." In: *Flow Measurement and Instrumentation* 54 (2017), pp. 39–45.

[209] Peter J Rousseeuw and Christophe Croux. "Alternatives to the median absolute deviation." In: *Journal of the American Statistical association* 88.424 (1993), pp. 1273–1283.

[210] Felipe Pfeifer Rubin, Paulo Silas Severo de Souza, Wagner dos Santos Marques, Rômulo Reis de Oliveira, Fábio Diniz Rossi, and Tiago Ferreto. "Evaluating energy and thermal efficiency of anomaly detection algorithms in edge devices." In: *2020 International Conference on Information Networking (ICOIN)*. IEEE. 2020, pp. 208–213.

[211] Lukas Ruff, Jacob R. Kauffmann, Robert A. Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G. Dietterich, and Klaus-Robert Müller. "A Unifying Review of Deep and Shallow Anomaly Detection." In: *Proceedings of the IEEE* 109.5 (2021), pp. 756–795. DOI: 10.1109/JPROC.2021.3052449.

[212] Takaya Saito and Marc Rehmsmeier. "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets." In: *PloS one* 10.3 (2015), e0118432.

[213] Rodrigo Barbosa de Santis and Marcelo Azevedo Costa. “Extended Isolation Forests for Fault Detection in Small Hydroelectric Plants.” In: *Sustainability* 12.16 (2020), p. 6421.

[214] Ryoma Sato, Makoto Yamada, and Hisashi Kashima. “Fast unbalanced optimal transport on a tree.” In: *Advances in neural information processing systems* 33 (2020), pp. 19039–19051.

[215] J. Schneible and A. Lu. “Anomaly detection on the edge.” In: *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)* (2017), pp. 678–682.

[216] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. “Estimating the support of a high-dimensional distribution.” In: *Neural computation* 13.7 (2001), pp. 1443–1471.

[217] Jonas Herskind Sejr and Anna Schneider-Kamp. “Explainable outlier detection: What, for Whom and Why?” In: *Machine Learning with Applications* 6 (2021), p. 100172.

[218] Burr Settles. “Active Learning Literature Survey.” In: *Science* 10.3 (1995), pp. 237–304.

[219] Burr Settles. “Active learning literature survey.” In: (2009).

[220] Yanhui Shen, Huawen Liu, Yanxia Wang, Zhongyu Chen, and Guanghua Sun. “A novel isolation-based outlier detection method.” In: *Pacific Rim International Conference on Artificial Intelligence*. Springer. 2016, pp. 446–456.

[221] Durga L Shrestha and Dimitri P Solomatine. “Machine learning approaches for estimation of prediction interval for the model output.” In: *Neural Networks* 19.2 (2006), pp. 225–235.

[222] Timo Speith. “A review of taxonomies of explainable artificial intelligence (XAI) methods.” In: *2022 ACM Conference on Fairness, Accountability, and Transparency*. 2022, pp. 2239–2250.

[223] Guillaume Staerman, Pavlo Mozharovskyi, Stephan Clémençon, and Florence d’Alché-Buc. “Functional isolation forest.” In: *Asian Conference on Machine Learning*. PMLR. 2019, pp. 332–347.

[224] Jakob Sternby, Erik Thormarker, and Michael Liljenstam. “Anomaly Detection Forest.” In: ().

[225] Ljiljana Stojanovic, Marko Dinic, Nenad Stojanovic, and Aleksandar Stojadinovic. “Big-data-driven anomaly detection in industry (4.0): An approach and a case study.” In: *2016 IEEE international conference on big data (big data)*. IEEE. 2016, pp. 1647–1652.

[226] Hongyu Sun, Qiang He, Kewen Liao, Timos Sellis, Longkun Guo, Xuyun Zhang, Jun Shen, and Feifei Chen. “Fast anomaly detection in multiple multi-dimensional data streams.” In: *2019 IEEE International Conference on Big Data (Big Data)*. IEEE. 2019, pp. 1218–1223.

[227] Gian Antonio Susto and Alessandro Beghi. “A virtual metrology system based on least angle regression and statistical clustering.” In: *Applied Stochastic Models in Business and Industry* 29.4 (2013), pp. 362–376.

[228] Gian Antonio Susto, Alessandro Beghi, and Seán McLoone. “Anomaly detection through on-line isolation forest: an application to plasma etching.” In: *2017 28th Annual SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*. IEEE. 2017, pp. 89–94.

[229] Gian Antonio Susto, Angelo Cenedese, and Matteo Terzi. “Time-series classification methods: Review and applications to power systems data.” In: *Big data application in power systems*. Elsevier, 2018, pp. 179–220.

[230] Gian Antonio Susto, Andrea Schirru, Simone Pampuri, Alessandro Beghi, and Giuseppe De Nicolao. “A hidden-gamma model-based filtering and prediction approach for monotonic health factors in manufacturing.” In: *Control Engineering Practice* 74 (2018), pp. 84–94.

[231] Gian Antonio Susto, Matteo Terzi, and Alessandro Beghi. “Anomaly detection approaches for semiconductor manufacturing.” In: *Procedia Manufacturing* 11 (2017), pp. 2018–2024.

[232] Gian Antonio Susto, Matteo Terzi, Chiara Masiero, Simone Pampuri, and Andrea Schirru. “A Fraud Detection Decision Support System via Human On-Line Behavior Characterization and Machine Learning.” In: *2018 First International Conference on Artificial Intelligence for Industries (AI4I)*. IEEE. 2018, pp. 9–14.

[233] Reed T Sutton, David Pincock, Daniel C Baumgart, Daniel C Sadowski, Richard N Fedorak, and Karen I Kroeker. “An overview of clinical decision support systems: benefits, risks, and strategies for success.” In: *NPJ digital medicine* 3.1 (2020), pp. 1–10.

[234] Yemada Taitel and Abe E Dukler. “A model for predicting flow regime transitions in horizontal and near horizontal gas-liquid flow.” In: *AIChE journal* 22.1 (1976), pp. 47–55.

[235] Swee Chuan Tan, Kai Ming Ting, and Tony Fei Liu. “Fast anomaly detection for streaming data.” In: *Twenty-Second International Joint Conference on Artificial Intelligence*. 2011.

[236] Yanghui Tan, Chunyang Niu, Hui Tian, Yejin Lin, and Jundong Zhang. “Decay detection of a marine gas turbine with contaminated data based on isolation forest approach.” In: *Ships and Offshore Structures* (2020), pp. 1–11.

[237] Andreas Theissler. “Detecting known and unknown faults in automotive systems using ensemble-based anomaly detection.” In: *Knowledge-Based Systems* 123 (2017), pp. 163–173.

[238] Suresh N Thennadil, Mark Dewar, Craig Herdsman, Alison Nordon, and Edo Becker. “Automated weighted outlier detection technique for multivariate data.” In: *Control Engineering Practice* 70 (2018), pp. 40–49.

[239] Richard Thorn, Geir A Johansen, and Bjørn T Hjertaker. “Three-phase flow measurement in the petroleum industry.” In: *Measurement Science and Technology* 24.1 (2012), p. 012003.

[240] Yuan Tian, Ruihao Yuan, Dezhen Xue, Yumei Zhou, Xiangdong Ding, Jun Sun, and Turab Lookman. “Role of uncertainty estimation in accelerating materials development via active learning.” In: *Journal of Applied Physics* 128.1 (2020), p. 014103.

[241] Kai Ming Ting, Guang-Tong Zhou, Fei Tony Liu, and James Swee Chuan Tan. “Mass estimation and its applications.” In: *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2010, pp. 989–998.

[242] Kai Ming Ting, Guang-Tong Zhou, Fei Tony Liu, and Swee Chuan Tan. “Mass estimation.” In: *Machine learning* 90.1 (2013), pp. 127–160.

[243] Maurras Ulbricht Togbe, Yousra Chabchoub, Aliou Boly, Mariam Barry, Raja Chiky, and Maroua Bahri. “Anomalies Detection Using Isolation in Concept-Drifting Data Streams.” In: *Computers* 10.1 (2021), p. 13.

[244] Mikhail Tokovarov and Paweł Karczmarek. “A Probabilistic Generalization of Isolation Forest.” In: *Information Sciences* (2021).

[245] Phuong Hanh Tran, Cédric Heuchenne, and Sébastien Thomassey. “An anomaly detection approach based on the combination of LSTM autoencoder and isolation forest for multivariate time series data.” In: *Proceedings of the 14th International FLINS Conference on Robotics and Artificial Intelligence (FLINS 2020)*. World Scientific. 2020, pp. 18–21.

[246] Yu-Lin Tsou, Hong-Min Chu, Cong Li, and Shao-Wen Yang. “Robust distributed anomaly detection using optimal weighted one-class random forests.” In: *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2018, pp. 1272–1277.

[247] Kévin Vaysse and Philippe Lagacherie. “Using quantile regression forest to estimate uncertainty of digital soil mapping products.” In: *Geoderma* 291 (2017), pp. 55–64.

[248] Stefan Wager, Trevor Hastie, and Bradley Efron. “Confidence intervals for random forests: The jackknife and the infinitesimal jackknife.” In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1625–1651.

[249] Yan-Bo Wang, Ding-Ge Chang, Shao-Rui Qin, Yu-Hang Fan, Hai-Bao Mu, and Guan-Jun Zhang. "Separating multi-source partial discharge signals using linear prediction analysis and isolation forest algorithm." In: *IEEE Transactions on Instrumentation and Measurement* 69.6 (2019), pp. 2734–2742.

[250] Zumin Wang, Jiyu Tian, Hui Fang, Liming Chen, and Jing Qin. "Light-Log: A lightweight temporal convolutional network for log anomaly detection on the edge." In: *Computer Networks* (2021), p. 108616.

[251] Hong Wang-jian. "Optimal input design for multi UAVs formation anomaly detection." In: *ISA transactions* (2019).

[252] Marc Weber, Simon Klug, Eric Sax, and Bastian Zimmer. "Embedded hybrid anomaly detection for automotive CAN communication." In: *9th European Congress on Embedded Real Time Software and Systems (ERTS 2018)*. 2018.

[253] Albert Weckenmann, X Jiang, K-D Sommer, Ulrich Neuschaefer-Rube, Jörg Seewig, Laura Shaw, and T Estler. "Multisensor data fusion in dimensional metrology." In: *CIRP annals* 58.2 (2009), pp. 701–721.

[254] René Wetzig, Anton Gulenko, and Florian Schmidt. "Unsupervised Anomaly Alerting for IoT-Gateway Monitoring using Adaptive Thresholds and Half-Space Trees." In: *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*. IEEE. 2019, pp. 161–168.

[255] Weng-Keen Wong, Andrew W Moore, Gregory F Cooper, and Michael M Wagner. "Bayesian network anomaly pattern detection for disease outbreaks." In: *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. 2003, pp. 808–815.

[256] Ke Wu, Kun Zhang, Wei Fan, Andrea Edwards, and S Yu Philip. "Rs-forest: A rapid density estimator for streaming anomaly detection." In: *2014 IEEE International Conference on Data Mining*. IEEE. 2014, pp. 600–609.

[257] Tong Wu, Ying-Jun Angela Zhang, and Xiaoying Tang. "Isolation forest based method for low-quality synchrophasor measurements and early events detection." In: *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE. 2018, pp. 1–7.

[258] Haolong Xiang, Zoran Salcic, Wanchun Dou, Xiaolong Xu, Lianyong Qi, and Xuyun Zhang. "OPHiForest: Order Preserving Hashing Based Isolation Forest for Robust and Scalable Anomaly Detection." In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020, pp. 1655–1664.

[259] Min Xu, Jeanne M David, Suk Hi Kim, et al. "The fourth industrial revolution: Opportunities and challenges." In: *International journal of financial research* 9.2 (2018), pp. 90–95.

[260] Yongjian Xue and Pierre Beauseroy. “Transfer learning for one class SVM adaptation to limited data distribution change.” In: *Pattern Recognition Letters* 100 (2017), pp. 117–123.

[261] Yong Yan, Lijuan Wang, Tao Wang, Xue Wang, Yonghui Hu, and Quansheng Duan. “Application of soft computing techniques to multiphase flow measurement: A review.” In: *Flow Measurement and Instrumentation* 60 (2018), pp. 30–43.

[262] Qibo Yang, Jaskaran Singh, and Jay Lee. “Isolation-Based Feature Selection for Unsupervised Outlier Detection.” In: *Annual Conference of the PHM Society*. Vol. 11. 1. 2019.

[263] Chengfei Yao, Xiaoqing Ma, Biao Chen, Xiaosong Zhao, and Gang Bai. “Distribution Forest: An Anomaly Detection Method Based on Isolation Forest.” In: *International Symposium on Advanced Parallel Processing Technologies*. Springer. 2019, pp. 135–147.

[264] Yuan Yao, Abhishek Sharma, Leana Golubchik, and Ramesh Govindan. “Online anomaly detection for sensor systems: A simple and efficient approach.” In: *Performance Evaluation* 67.11 (2010), pp. 1059–1075.

[265] Xiao Yu, Lu An Tang, and Jiawei Han. “Filtering and refinement: A two-stage approach for efficient and effective anomaly detection.” In: *2009 Ninth IEEE International Conference on Data Mining*. IEEE. 2009, pp. 617–626.

[266] Huitaek Yun, Hanjun Kim, Young Hun Jeong, and Martin BG Jun. “Autoencoder-based anomaly detection of industrial robot arm using stethoscope based internal sound sensor.” In: *Journal of Intelligent Manufacturing* (2021), pp. 1–18.

[267] Chunkai Zhang, Ao Yin, Yepeng Deng, Panbo Tian, Xuan Wang, and Lifeng Dong. “A novel anomaly detection algorithm based on trident tree.” In: *International Conference on Cloud Computing*. Springer. 2018, pp. 295–306.

[268] Haifeng Zhang, Yiyuan Yang, Ming Yang, Likun Min, Yi Li, and Xiangyuan Zheng. “A Novel CNN Modeling Algorithm for the Instantaneous Flow Rate Measurement of Gas-liquid Multiphase Flow.” In: *Proceedings of the 2020 12th International Conference on Machine Learning and Computing*. 2020, pp. 182–187.

[269] Xuyun Zhang, Wanchun Dou, Qiang He, Rui Zhou, Christopher Leckie, Ramamohanarao Kotagiri, and Zoran Salcic. “LSHiForest: A generic framework for fast tree isolation based ensemble anomaly analysis.” In: *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE. 2017, pp. 983–994.

[270] Yi Zhang, Peng Peng, Chongdang Liu, and Heming Zhang. “Anomaly detection for industry product quality inspection based on Gaussian restricted Boltzmann machine.” In: *2019 IEEE international conference on systems, man and cybernetics (SMC)*. IEEE. 2019, pp. 1–6.

[271] Chaojie Zhao, Guozhu Wu, Haifeng Zhang, and Yi Li. “Measurement of water-to-liquid ratio of oil-water-gas three-phase flow using microwave time series method.” In: *Measurement* 140 (2019), pp. 511–517.

[272] Yong-Ping Zhao, Gong Huang, Qian-Kun Hu, and Bing Li. “An improved weighted one class support vector machine for turboshaft engine fault detection.” In: *Engineering Applications of Artificial Intelligence* 94 (2020), p. 103796.

[273] Yue Zhao, Zain Nasrullah, and Zheng Li. “PyOD: A python toolbox for scalable outlier detection.” In: *Journal of Machine Learning Research* 20.96 (2019), pp. 1–7.

[274] Shisheng Zhong, Song Fu, Lin Lin, Xuyun Fu, Zhiquan Cui, and Rui Wang. “A novel unsupervised anomaly detection for gas turbine using isolation forest.” In: *2019 IEEE International Conference on Prognostics and Health Management (ICPHM)*. IEEE. 2019, pp. 1–6.

[275] Jianlong Zhou, Amir H Gandomi, Fang Chen, and Andreas Holzinger. “Evaluating the quality of machine learning explanations: A survey on methods and metrics.” In: *Electronics* 10.5 (2021), p. 593.

[276] Xiaojin Zhu and Andrew B Goldberg. “Introduction to semi-supervised learning.” In: *Synthesis lectures on artificial intelligence and machine learning* 3.1 (2009), pp. 1–130.

[277] Tiago Zonta, Cristiano André Da Costa, Rodrigo da Rosa Righi, Miromar Jose de Lima, Eduardo Silveira da Trindade, and Guann Pyng Li. “Predictive maintenance in the Industry 4.0: A systematic literature review.” In: *Computers & Industrial Engineering* 150 (2020), p. 106889.