UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

UNIVERSITÀ
DEGLI STUDI
DI BERGAMO

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

UNIVERSITÀ DEGLI STUDI DI BERGAMO

PH.D. THESIS

IN

TECHNOLOGY, INNOVATION AND MANAGEMENT

ADVANCED ANALYTICS ON MASSIVE DATASETS USING ARTIFICIAL INTELLIGENCE: INDUSTRY 4.0 SCENARIOS

ANTONIO GALLI

TUTOR: PROF. VINCENZO MOSCATO

COORDINATOR: PROF. RENATO REDONDI

XXXV CICLO

Table of contents

Summary

Artificial intelligence (AI) is fast becoming a general purpose technology with outstanding impacts in industries worldwide, thus supporting the Industry 4.0 revolution. More in detail, nowadays Deep Neural Networks (DNNs) are widely adopted in several fields, including critical systems, medicine, self-guided vehicles etc. Among the reasons sustaining this spread there are the higher generalisation ability and performance levels that DNNs usually obtain when compared to classical machine learning models.
This thesis provides an evaluation of different AI applications in Industry 4.0, highlighting the great potential in diverse areas of interest. The rise of Big Data in the late 2000s, a term referring to the collection of huge amounts of often unstructured data from diverse sources (e.g., images, text, audio, medical signals, etc.) has made the role of data center increasingly important to many applications.

Hard disk drive failures are one of the most common causes of service downtime in data centers. Predictive maintenance techniques have been adopted to extend the Remaining Useful Life (RUL) of these drives, and minimize service shortage and data loss. Several approaches based on machine and deep learning techniques have been proposed to address these issues,

mostly exploiting models based on Self-Monitoring analysis and Reporting Technology (SMART) attributes. While these models have proven to be reliable, their performance is affected by the lack of information about the proximity of disk failure in time. Moreover, many of these techniques are sensitive to the highly unbalanced nature of existing data-sets, in terms of good to failed hard disk ratio. In this thesis, a LSTM (Long Short Term Memory) based model combining SMART attributes and temporal analysis for estimating a hard drive health status according to its time to failure has been proposed. Proposed approach outperforms state-of-the-art methods when evaluated on two data-sets, one containing hourly samples from 23,395 disks and the other reporting daily samples from 29,878 disks. Experimental results showed that proposed approach is well suited to data-sets with different sampling periods, being able to predict hard drive health status up to 45 days before failure. However, the task related to predictive maintenance for components such as HDD is not the only field of application in fact , in the last decade *Anomalous Sound Detection* (ASD) , is becoming an increasingly challenging task for a plethora of applications due to the widespread diffusion of Deep Neural Networks. Nevertheless, the arise of recent cyber-physical attacks (i.e. Triton or Stuxnet), that deceive monitoring platforms, pose novel and challenging issues. For this reason, advanced predictive maintenance techniques are starting to exploit sounds generated by particular industrial equipment, whose analysis can unveil symptom of possible failures. For this kind of context, it is very easy to collect data related to normal and abnormal behaviour of a given machinery, thus several kinds of deep neural architectures can be effectively trained to predict eventual downtime situations. In this thesis, a novel deep learning-based methodology for anomalous sound detection task, having flexibility, modularity and efficiency characteristics has been proposed. The proposed methodology analyzes audio clips based on the *mel-spectogram* and ID equipment information, while a *one-hot* encoding method extracts features that are, successively, used to train an *ID Condi-*

tioned Network. In particular, the main novelty of the proposed methodology concerns the conditioning of an autoencoder by jountly analyzing the relationships between mel-spectogram and the related machine identifier through an encoder-decoder architecture for computing an anomaly score related to the input sequence. Several experiments have been made for investigating the efficiency and effectiveness of the proposed methodology on multiple instances of different industrial machines (pumps, valves, slide rails and fans), achieving low inference time and memory requirements w.r.t. the other approaches in the literature. Nowadays there are many fields, in addition to those listed above, that exploit AI advantageously. In particular, the energy sector is one of those that has taken more advantages from the implementation of AI approaches, especially Machine Learning models, for several applications, including energy performance benchmarking of buildings. However, the black-box approach could lead to a lack of result interpretability thus preventing the effective application of AI in some real-world scenarios. For this reason, eXplainable Artificial Intelligence (XAI) tools can be effectively embedded within an AI-based Energy Analytics methodology in order to enhance the explainability of the model results. In this thesis, an explainable AI-based benchmarking framework for estimating the membership to specific energy performance classes of a large set of Energy Performance Certificates (EPCs) of flats has been proposed. The classification is obtained by leveraging different black-box classifiers characterized by high accuracy, yet their inference mechanism is not human-readable. Therefore, a generalizable XAI methodology, based on the combination of a local explainer together with a clustering algorithm, is employed to explain the model results and causal effects between the predictors and target variable to better understand the model behavior, and the motivations behind correct and wrong performed classifications. The thesis provides a general methodological approach capable to exploit a limited number of instances to extract, explain and interpret inference mechanisms learnt by the model that are useful for the end-user.

The framework was tested on about 100,000 EPCs of flats located in Italy. As mentioned before, recently there has been a growing interest in XAI, a field providing tools, techniques and algorithms designed to generate interpretable explanations, comprehensible to humans, for the decisions made by a machine learning model. However, it has been demonstrated that DNNs are susceptible to Adversarial Perturbations (APs), namely procedures intended to mislead a target model by means of an almost imperceptible noise. The relation existing between XAI and AP is extremely of interest since it can help improve trustworthiness in AI-based systems. To this aim, it is important to increase awareness of the risks associated with the use of XAI in critical contexts, in a world where APs are present and easy to perform. On this line, we quantitatively analyse the impact that APs have on XAI in terms of differences in the explainability maps. Since this thesis wants to be just an intuitive proof-of-concept, the aforementioned experiments are run in a fashion easy to understand and to quantify, by using publicly available dataset and algorithms. Results show that AP can strongly affect the XAI outcomes, even in the case of a failed attack, highlighting the need for further research in this field.

Part I

The Fourth Industrial Revolution

The influence of *Artificial Intelligence* (AI) on the industry has been so significant in recent years that it has given rise to a new trend of research and applications known as "Industry 4.0" . This term refers to the use of artificial intelligence to facilitate effective data exchange and processing in manufacturing technologies, services, and transportation, laying the groundwork for what is commonly referred to as the fourth industrial revolution.

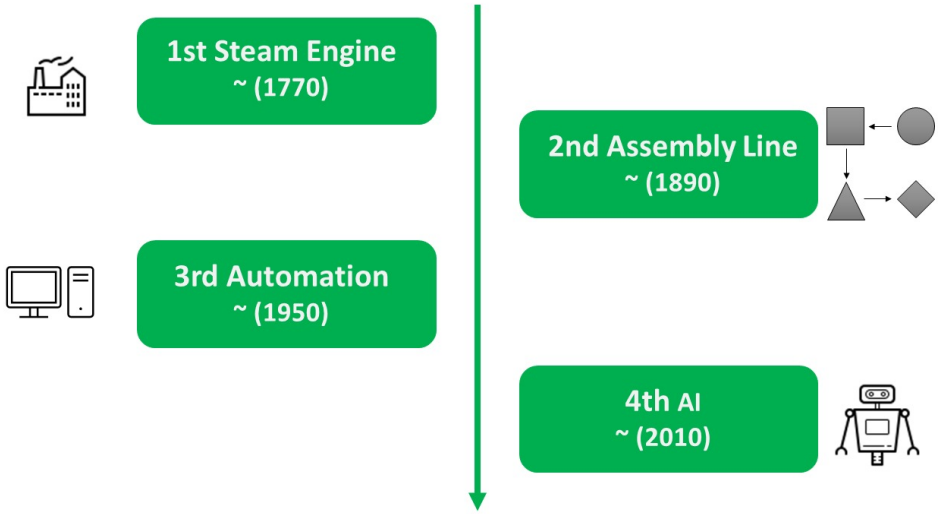


Figure 1: Timeline for the four industrial revolutions: the first, based on the invention of the steam engine; the second, thanks to the development of the assembly line by Henry Ford; the third, with the development of computers and automation; the fourth, supported by artificial intelligence.

Nowadays, many industries have been impacted by it such as: manufacturing [169], logistic [19] and transport systems [159] which represent only some of the fields in which machine learning is expected to have a very important impact in the near future.

Nevertheless, there are many contexts in which some impact are already visible:

Smart city represents what is expected to be the place where we, as human beings, will live in the near future. Although the term may suggest utopian dreams, it actually aims to design cities intended to support the well-being of citizens, such as by reducing stress, pollution and optimizing services. [111];

Smart energy harvesting and grid design are becoming extremely important, both on a local [153] and on a national [166] scale, in a world more and more relying on renewable energy rather than fossil fuels;

Automotive and avionics are probably the first use cases in which the use of machine learning has been applied for the auto-pilot [191, 24]. However, in an increasingly connected world aiming for self-driving vehicles, adequately controlling large amounts of traffic accurately and effectively becomes one of the biggest challenges to solve[179];

Telemedicine that, thanks to the ultra wide-band connectivity delivered by 5G technology [183], allow to analyse huge amounts of patients data to provide AI-based solution in several clinical contexts [117];

Robotics in industrial applications where, thanks to the advances in the computer vision field, cooperate with and assist humans in safety-critical environments [99, 68];

Security enforcement, with AI opening new scenario in user identification [39] and anti-malware protection [92];

The examples listed above are just a few applications and, unfortunately, represent only one side of the coin. In fact, the disruptive spread and success of AI also involves some negatives, which find among their most famous examples applications related to the violation of subjects' privacy, unethical

behavior, and attacks on such systems, such as user profiling [6], fake news generation [143], autonomous weapon systems [9], discriminatory advertising [47], etc.

We strongly believe that because it is only a very powerful tool, artificial intelligence is not to blame. However, we argue that in order to develop a more ethical, equitable, and safe use of artificial intelligence, all stakeholders such as users, developers, and regulators need to have a very clear idea of what AI is and what are the implications of its use in various sectors such as industry.

1 The Artificial Intelligence Era

The term Artificial Intelligence (AI) refers to the ability of a computer to perform functions similar to the typical reasoning of the human brain.

Although it is thought to have emerged recently, the first studies on the development of an artificial agent started in the early 1940s. Since the beginning, AI has been at the center of the debate between scientists and philosophers, with the former interested in theory and techniques aimed at developing algorithms that would allow machines to exhibit intelligent abilities and actions, while the latter interested in aspects related to the possible implications of considering an artifact as an intelligent entity. In the opinion of Marvin Minsky, considered one of the fathers of AI along with Alan Turing and Frank Rosenblatt, the aim of this new field is "to develop machines capable of autonomously doing things that would require intelligence if they were done by humans [126].

However, there are some problems in this definition of artificial intelligence, one of them being the unavailability of a universally agreed definition of "intelligence." That is why nowadays the most widely used definition is based on the "imitation game," a test proposed by Alan Turing in his 1950 article entitled "Computing Machine and Intelligence" [184]. The idea is to consider a machine that, hidden behind a screen and connected to the world

through an appropriate communication interface, is able to fool a human tester into believing that it is interacting with another human being, in which case it is defined as intelligent. Nowadays, despite the availability of AI models capable of competing with and, in some cases, even outperforming humans [13, 178, 127], we are still far from a general AI model that can meet the definition of an intelligent machine capable of fooling a human being. In fact, there is a characteristic of even the best-performing AI systems that they are only suitable for the task for which they were designed, mainly because of the wide variety of available "input signals" and the lack of generalization capabilities across domains and tasks. Nevertheless, the use of artificial intelligence continues to spread, not only in businesses and high-tech sectors, but also in everyday applications, with a worldwide impact on the economy (figure 1.1).

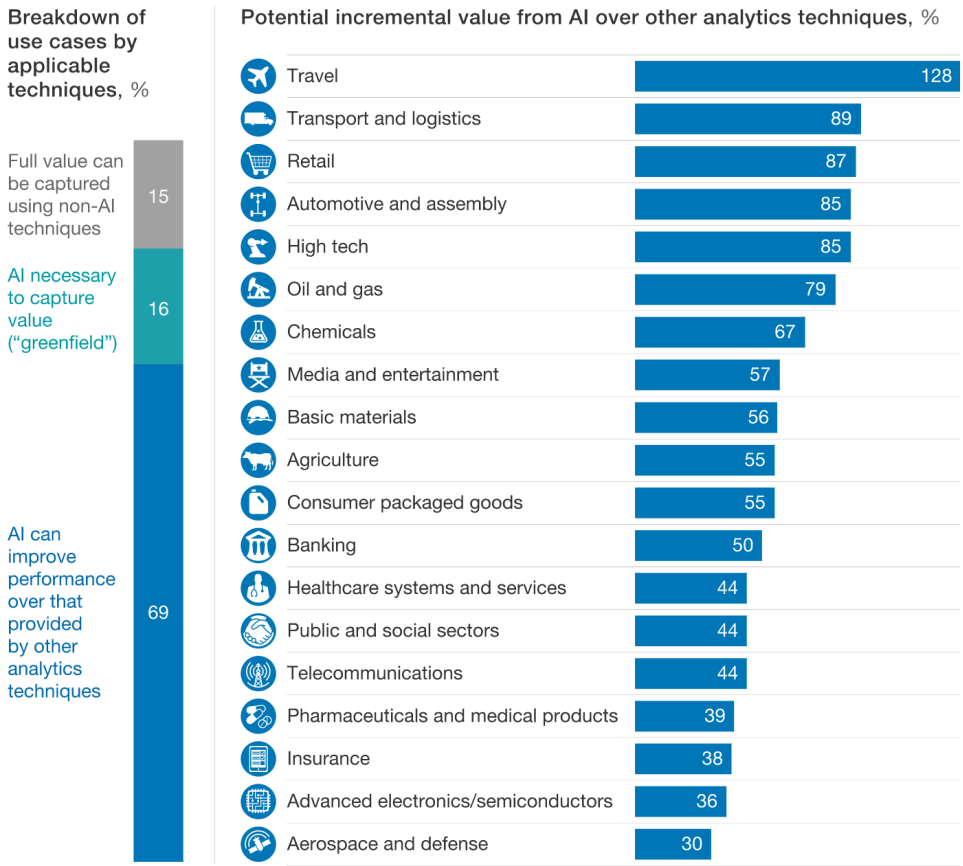


Figure 1.1: McKinsey Global Institute analysis on the potential impact that AI will have in the next future on some important industries [118].

1.1 From Shallow to Deep Neural Networks

In recent years, the term artificial intelligence has become more and more commonly used; in fact, we are increasingly dealing with smart cell phones, smart voice assistants, etc. Nowadays, the term AI is also often misused, and one of the effects of this widespread use among the mass audience is the confusion that arises with all related terms, such as "Pattern Recognition," "Machine Learning," "Deep Learning," etc. Therefore, in order to help the

reader better understand the terminology and contributions made in this thesis, the figure 1.2 gives a brief toponymy of the most commonly used terms in AI.

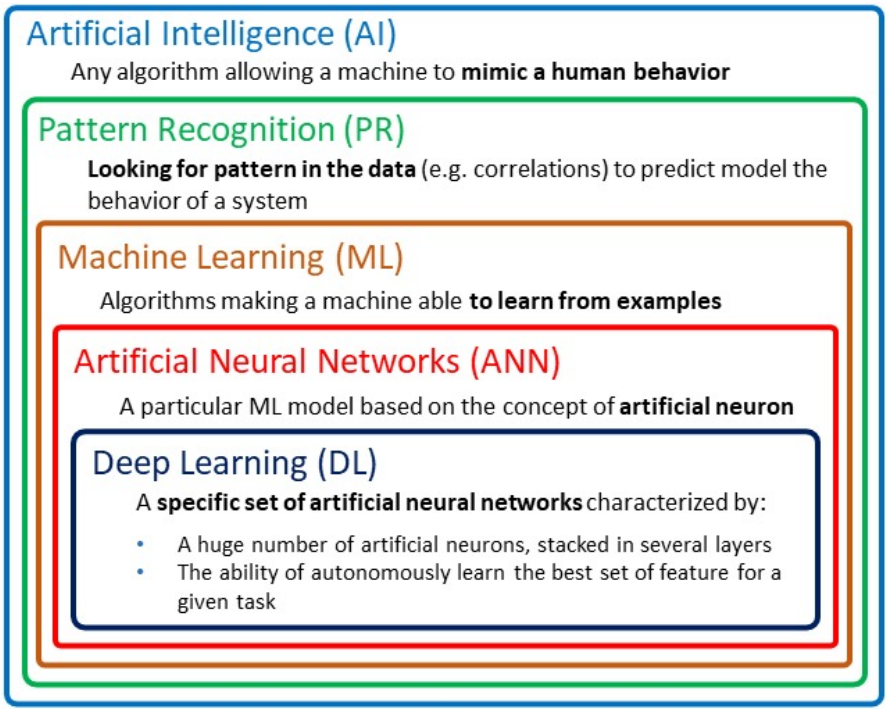


Figure 1.2: A brief toponymy of some of the most common terms related with "artificial intelligence". A box included into another represents the relation between a sub-concept and a concept.

In fact, usually what the media call AI is indeed machine learning (ML). Actually, what is capturing the attention is the ability of these types of AI systems to learn from examples, that is, to learn how to perform a task through the use of examples. This feature removes the programmer from the responsibility of writing down the sequence of operations needed to perform a given task (algorithm), consequently allowing the system to deal with complex tasks for which it would have been impossible to code a solution.

Among all ML models, Artificial Neural Networks (ANNs) are surely the branch that has received the most attention for their inspiration from the human brain, in fact similar to how the latter consists of a complex interconnected structure of biological neurons, the former is a network of artificial neurons. Specifically, an artificial neural network is a parallel structure whose elements are organized in layers and interact with each other to perform, after an appropriate training phase, the desired task.

More recently, the term Deep Learning is often used as a synonym for AI/ML. In particular, the term refers to a particular subset of ANNs characterized by a very deep structure (composed of many layers). An important key aspect of deep models is their ability to independently learn, during the training phase, the best representation of the input (or set of features) for the task under analysis. This feature, known as feature learning, has been essential to the spread of AI, as it has enabled its use even in domains lacking effective features designed by experts. Several networks have been proposed and optimized for the analysis of specific types of data , e.g., images , time series, videos etc. When it comes to elaborate images, Deep Convolutional Neural Networks (D-CNNs or simply CNNs) have shown incredible performance in a wide range of research fields, including natural image processing [196], biomedical applications [66], biometrics [41] and many others [201, 167, 83]. The core of CNNs are convolutional layers, namely layers of neurons leveraging the concept of convolution between the input and a kernel to perform the feature extraction. Unlike the human-based feature extraction, where the feature design process is fixed, convolutional layers allows CNNs to adapt the feature extraction during the training itself since the kernels used for the convolutions are learnt together with the other neurons' weights. When several convolutional layers are stacked in sequence, the whole network starts learning how to extract a hierarchical set of features, from a low to a high level of details (figure 1.5).

A Recurrent Neural Network (RNN) is a type of artificial neural network which uses either sequential data or time series data. Like feedforward and CNNs, recurrent neural networks utilize training data to learn. They are distinguished by their "memory" as they take information from prior inputs to influence both the current input and the output. While traditional deep neural networks assume that inputs and outputs are independent of each other, the output of recurrent neural networks depends on prior elements within the sequence.

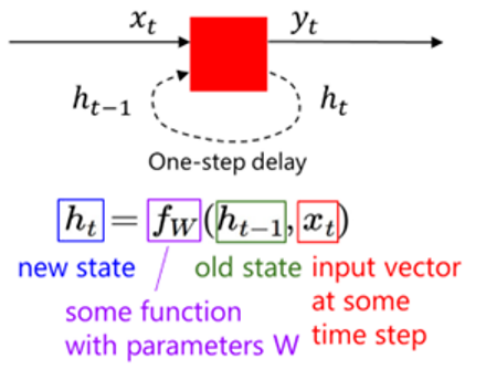


Figure 1.3: Illustration showing how a RNN works

A Long Short-Term Memory (LSTM) network is a specific type of RNN. As mentioned before for RNNs, also LSTMs excel in learning, processing, and classifying sequential data. The most popular way to train an RNN is by backpropagation through time. However, the problem of the vanishing gradients often causes the parameters to capture short-term dependencies while the information from earlier time steps decays. Long short-term memory networks aim to overcome the issue of the vanishing gradients by using the gates to selectively retain information that is relevant and forget information that is not relevant. Lower sensitivity to the time gap makes LSTM networks better for analysis of sequential data than simple RNNs. Figure 1.4 shows the

internal structure of an LSTM cell. It consists of three gates referred to as input gate, exit gate, and forget gate, governed by the following equations:

$$f_t = \sigma\left(W_f * [h_{t-1}, x_t] + b_f\right) \tag{1.1}$$

$$i_t = \sigma\left(W_i * [h_{t-1}, x_t] + b_i\right) \tag{1.2}$$

$$\widetilde{C}_t = tanh\left(W_C * [h_{t-1}, x_t] + b_C\right) \tag{1.3}$$

$$C_t = f_t \circ C_{t-1} + i_t \circ \widetilde{C}_t \tag{1.4}$$

$$o_t = \sigma\left(W_o * [h_{t-1}, x_t] + b_o\right) \tag{1.5}$$

$$h_t = o_t \circ tanh(C_t) \tag{1.6}$$

where: f_t, i_t, o_t denote, respectively, the forget gate, the input gate, and the output gate; W_f , W_i, W_C , W_o denote the weight matrices related to the relative gates; $\circ$ as the symbol for the element-wise product.

The first step performed when data enter the LSTM cell is to select the information to store. This step is regulated by equation 1.1. The x_t data array containing the values of the features at instant t is input, along with the h_{t-1} output. The result obtained is given in input to the activation function (sigmoid σ) which returns a value between 0 and 1 for each element of cell state C_{t-1}, where 0 and 1 indicate if the element can be ignored or stored in time, respectively. The second step is related to the input gate and allows the LSTM to establish which new information to store. This step is realized through two operations related to equations 1.2 and 1.3. At the end of these operations, the state of the LSTM cell is updated through equation 1.4. Finally, the last step consists of the generation of the LSTM output through equations 1.5 and 1.6 which select the part of the new state to be put in output.

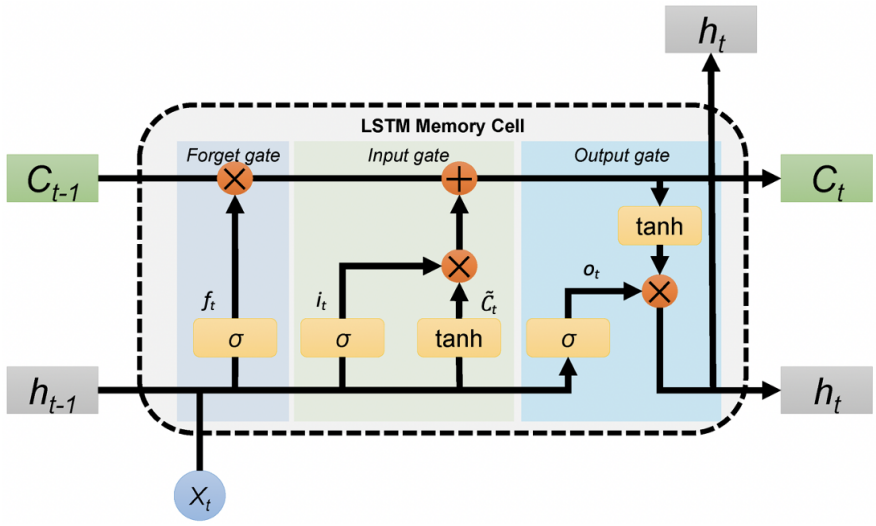


Figure 1.4: Illustration showing LSTM cell and gate.

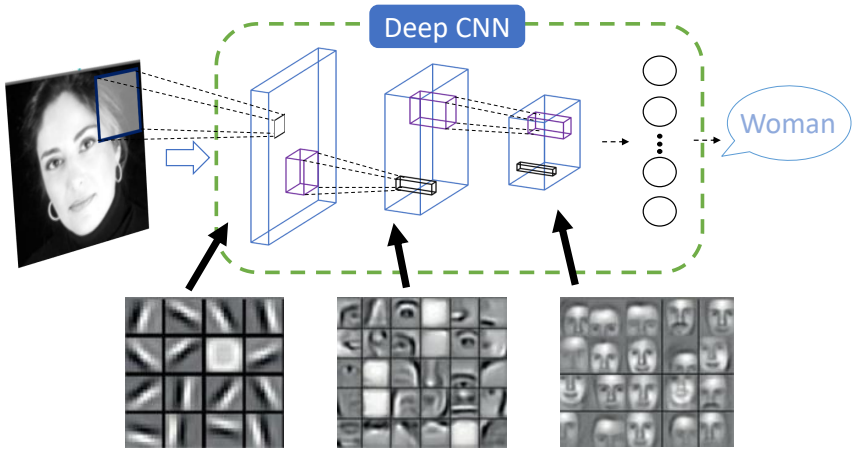


Figure 1.5: Illustration showing how a 3-layered CNN learns to describe a complex image as composed by many simpler concepts.

Several are the factors that concurred to make neural networks, and in particular deep architectures, regain popularity. Excluding some mathematical intuitions, four are the factors that contributed the most:

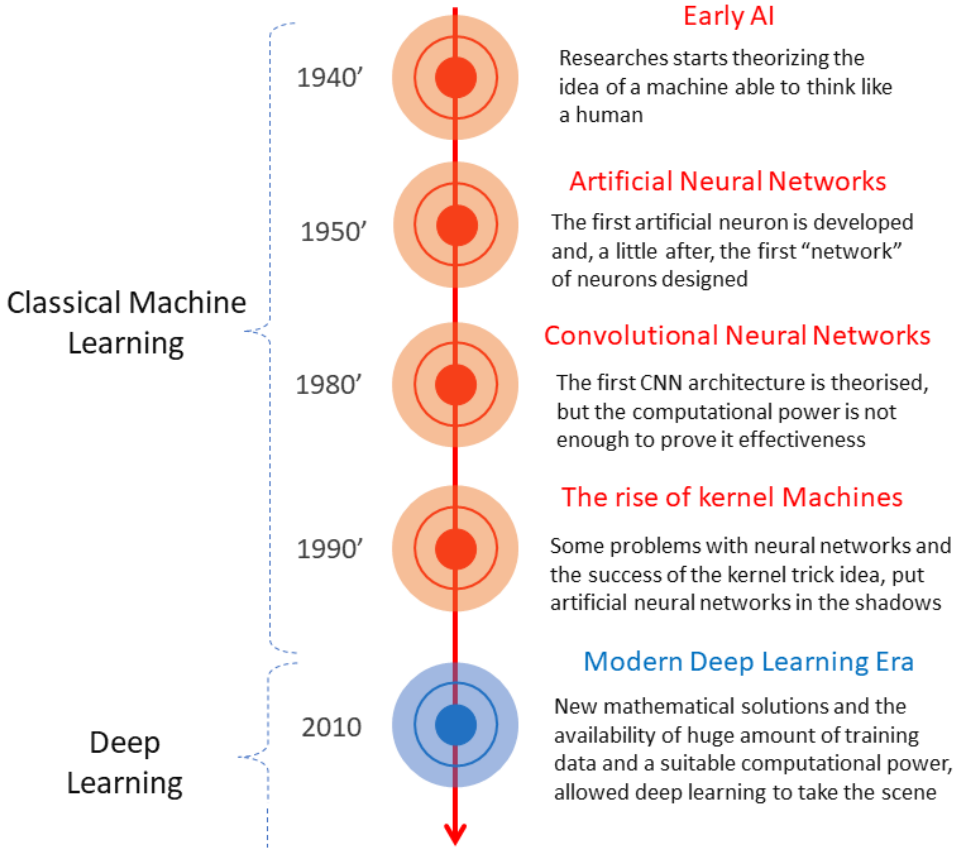


Figure 1.6: Short timeline for some of the AI most important events.

- The rise of **Big Data** in late 2000, a term referring to the collection of massive amounts of data often unstructured and coming from different sources (e.g. images, text, audio, medical signals, etc.). The availability of public and free to use collections of labelled samples started to show the limits of kernel machines while allowing researches to experiment with richer ANNs;

- The computational power needed to optimise the huge number of parameters (typical of deep architecture) and to iteratively elaborate the

involved massive training datasets has been a technical limitation for a long time. On this regards, the advances in **General-Purpose GPU (GP-GPU) computing** strongly sustained the spread of deep learning models, allowing a significant reduction in the required training times;

- The last step to do before really considering deep learning within everyone's reach was to make GPU computing accessible (since suited GPU were, and sometimes still are, relatively expensive). Some companies quickly catch this business opportunity, starting to provide easy to use web-based virtual machines intended to allow developers to use remote GPUs for a little price. In 2017, Google publicly released Colaboratory[1], a totally free web-based IDE providing remote GPU acceleration.

[1] https://colab.research.google.com/

2 Artificial Intelligence in Industry

2.1 HDDs and Predictive Maintenance

Introduced by IBM in 1956, HDDs have become a reliable, wide-spread technology for data storage. So wide-spread, in fact, that in 2017 Western Digital predicted that by 2020 70% of all data would be stored in HDDs [Coughlin]. As of March 2019 Backblaze — a pioneer data storage provider — reported 106,238 spinning hard drives in their cloud storage ecosystem spread across three data centers [87]. Even the rise of modern Solid State Drives (SDDs) — a storage technology with no moving parts which instead uses semiconductor chips with storage cells — has not affected the popularity of HDDs in data-centers. There is a simple reason for this: although SSDs consume less energy, HDDs' trade-off between storage capacity, life expectancy, and cost remains unbeaten [Bauer].

While the pervasive use of HDD technology is thus unquestionable, its reliability remains an important issue. Moreover, the increasing popularity of big data applications has made it so that storage systems are required to possess exabytes of capacity, usually resulting in millions of hard disk drives per data center [88]. Obviously, at such a scale disk failures become the norm. The fundamental concern is then not just the life-expectancy of a standard

HDD, nor the quality of information storage, but the frequency of unexpected failures. Therefore, resources have recently been focused on the search for optimal strategies to deal with such failures.

In order to come up with cost-effective strategies against data-loss, it is important to individuate the primary causes of disk failures. Being aware of these causes is the first step towards mitigating HDD failure risk, as it allows for a variety of monitoring systems that target each failure source specifically.

As disk failure in large-scale storage systems becomes unavoidable though, solutions to data-loss have increasingly been relying on redundant arrays of inexpensive disks (RAID; [64]). However, RAID recovery is a time-consuming process which requires bringing the failed system offline for repair. This results in delays on the user side — as the number of data sources decreases considerably while the damaged disk is being replaced — and in additional stress on the remaining disk drives due to the intensive read and write activities [146]. Therefore, while RAID approaches are an effective way to address the data-loss problem from a general perspective, they remain an overall unsatisfying solution.

Importantly, these kind of methods are *reactive*: they provide a safety net in case a failure occurs. Reactive storage protection approaches thus suffer from high recovery overheads, which significantly affect data availability and system performance [146]. For such reasons, recent efforts have been focused on exploring *proactive* solutions to HDD failure. If we have technologies that are able to predict disk failure in advance with a high confidence rate, then we can plan data rescue operations so that they overlap with regular storage operations — thus reducing the intervals of data-unavailability. Effectively predicting HDDs' health status becomes then the core challenge of maintenance operations.

In this sense, predictive maintenance models have become the state-of-the-art in maintenance research, by using historical logs of real-time data to predict the future failure of still operating hard disks.

In the last few years, predictive technologies have been relying more and more on machine learning techniques grounded in big data analysis in order to monitor the on-line health status of a hard drive, and improve fault prediction accuracy. Importantly though, this comes with the challenge of balancing information coming from multiple data sources (e.g., temperature, vibration), in order to take into consideration all of the possible causes of failure, and come up with appropriate maintenance strategies. Furthermore, it has been observed that hard drives often deteriorate gradually over time [170, a.o.]. Thus, it is important to be able to model the temporal dynamic of the dependencies within SMART attributes.

However, predictive systems need to strike a careful balance when extrapolating health information from these sources in order to raise failure warnings, as high false alarm rates would lead to as many overhead costs as missed disk failures. It is then crucial to find the most sensible way to exploit each HDD's health attributes, so to balance high accuracy and conservative predictions. This requires expert domain knowledge, in order to best tune the features of the model and address challenged due to feature specificity and cross-correlation between health attributes.

Moreover, hard disk data vary between manufacturers — and even between hard disk models within the same manufacturer. This implies that a previously specified model cannot be carried over to perform equally well in different data centers, and explicit feature engineering needs to be repeated multiple times based on the characteristics of each model [171].

In this thesis, we follow recent research in predictive maintenance, and present a deep learning approach addressing many of the current problems in the literature (data sparsity, need for domain knowledge, manual feature engineering).

As mentioned, HDD failure prediction plays a very important and crucial role in reducing data center downtime and significantly improving service reliability. By collecting information about the *health* conditions of HDD in

	Methodology	Pros	Cons
[129]	A multiple-instance learning framework using a naive Bayesian classifier for predicting failures	Maintaining low false alarm rate.	No information about HDD health level status. The main focus is on false alarm avoidance.
[103]	A classification and regression model based on SMART attributes for predicting failures	Maintaining low false alarm rate.	No information about HDD health level status. The main focus is on false alarm avoidance.
[198]	An RNN model based on SMART attributes for evaluating HDD status.	Performs health status assessment.	Health degree settings manually defined in terms of number and size of each interval. The prediction phase combines the prediction obtained by the analysis of the last 1 hour.
[165]	A prediction model based on a part-voting Random Forest algorithm.	The prediction model differentiates failure prediction in a coarse-grained manner	The model does not consider data correlation and historical information about HDDs.
[195]	A prediction model based on an Online Random Forest algorithm.	The model evolves with sequential arrival of data on the flying adapting to SMART distribution over the time	This model does not consider data correlation and historical information about HDDs.
[190]	A two-step approach: anomaly detection according to a sliding window and a failure prediction model.	It tries to balance the failure detection rate and false alarm.	It does not provide any information about HDD health assessment. The sliding window is manually computed based on the number of samples ignoring correlation with the timestamp.
[10]	Compares two model based respectively on Random Forest and LSTM for HDD RUL estimation.	It predicts an HDD's Remaining Useful Life.	Health degree settings manually defined in terms of number and size of each interval. Only the current snapshot of SMART attribute values's sequence are considered according to the number of samples ignoring correlation with the timestamp.
[20]	An LSTM-based prediction model for Remaining Useful Life extimation.	It performs a hyper parameter optimization with to improve the prediction phase.	Health degree settings manually defined in terms of number and size of each interval. Only the current snapshot of SMART attribute values' sequence are considered according to the number of samples ignoring correlation with the timestamp.
[205]	A method based on adversarial training and layerwise perturbation for HDD health status prediction.	They propose a Layerwise Perturbation-Based Adversarial Training method to deal with overfitting and biased fitting problems.	Health degree settings manually defined in terms of number and size of each interval.
[25]	Pipeline based on SMART attributes for disk replacement.	Statistical technique are used to correlate SMART parameters and disk replacement.	The authors consider only two classes (healthy and replaced) and they do not provide any information about the Remaining Useful Life.
[199]	A cost-sensitive ranking-based machine learning model for disk error prediction.	Combining SMART attributes with system-level signals.	The authors consider only two classes and they do not provide any information about the Remaining Useful Life.
[174]	A TCNN-based prediction model for hardware failure prediction.	TCNN model for analyzing SMART attribute distribution over time.	The authors consider only two classes and they do not provide any information about the Remaining Useful Life.

Table 2.1: Overview of State-of-the-Art approaches

real-time, SMART records have been consistently used in failure detection systems — though with remarkably low failure detection rates (FDRs).

The use of information collected by sensors online is clearly essential to efficient prediction systems. Thus, different methods have been proposed to optimize the performance of SMART based models: from Bayesian classifiers [71] and support vector machines (SVM; [129]), to classification trees [103], back-propagation neural networks (BPNNs; [208]), and recurrent neural networks (RNNs; [198]). A summary of the approaches analyzed in this Section is presented in Table 2.1.

Hamerly & Elkan [71] were among the first to use Bayesian modelling in

failure prediction. The intuition behind their work is to re-frame failure prediction as an anomaly detection problem, and then classify test data based on the probability of reading from an HDD behaving normally, and information about the HDD internal conditions. They compare two methods — a mixture of Naive Bayes sub-models trained on expectation-maximization (EM) and a Naive Bayes — and show how their models perform significantly better than previously industry level predictors.

Following this line of investigation, Murray *et al.* [129] conduct an extensive evaluation of four different methods (SVM, unsupervised clustering, rank-sum and reverse arrangements test). They then propose an algorithm based on the multiple-instance learning framework and a naive Bayesian classifier (*mi-NB*) to deal with false-alarm rate (FAR). Also interested in minimizing FAR while maximizing detection accuracy, Zhu *et al.* [208] evaluate the accuracy of an SVM and a back-propagation neural network exploiting SMART attributes *and* their change rates. They show that while the SVM model achieves the lowest FAR (0.03%), the back-propagation model has the best high detection rate (95%). A similar approach is adopted by Li *et al.* [103], which propose a surprisingly well-performing prediction model based on regression trees.

More recently, Xu et al. [198] suggested that the inefficiency of past prediction systems stems from the fact that most HDD are not simply *good* or *bad*, but they are subject to gradual decay. Thus, they use Recurrent Neural Networks (RNNs) to model gradual changes in sequential SMART attributes and better capture the shifting nature in HDDs' health status. Their model achieves better performance than previous sequence independent models *and* short-term sequence dependent models. With a similar goal in mind, Botezatu et al. [25] designed a pipeline based on SMART attributes for predicting disk replacements approximately 15 days in advance. To deal with over-fitting and biased-fitting problems, [205] propose a Layerwise Perturbation method, using adversarial training to predict HDD health status on the basis of three health levels manually defined. Furthermore, a cost-sensitive ranking-based

machine learning model, combining SMART attributes with system-level signals, has been proposed by Xu et al.[199] for predicting disk error.

Finally, it has been observed that the data available on HDDs' health status is highly unbalanced (in particular, in the ratio of healthy and failure samples) — an imbalance that is bound to affect the performance of prediction system relying on intrinsic features.

To partially address this issue, Shen *et al.* [165] propose a failure prediction model based on a part-voting random forest algorithm which compensates for data unbalance using a clustering-based under-sampling method.

Similarly, Xiao *et al.* [195] exploit an online random forest prediction model, which evolves on-the-fly with sequential arrival of data, according to the variance of SMART distribution over time. Their model addresses both the problem of labelling sequential samples gathered on-the-fly, and the high unbalance in the distribution of healthy/failing disks.

Sun et al. [174] propose the use of a temporal Convolutional Neural Network (TCNN) in order to address the high variability of delay-to-failure values in real world scenarios with sparse failure samples, while reducing sensitivity to noise in the analysis of SMART distributions over time. To address issues due to data sparsity, they extend the binary cross-entropy loss function emphasizing the loss of misclassified samples. TCNNs within this approach show superior performance than RNNs and LSTMs.

Similarly efficient methods of failure prediction have highlighted the fundamental importance of taking into account the sequential nature of the SMART attributes when modelling failure rates [190]. In this spirit then, it seems to be crucial to define models that adapt to the dynamical changes in the SMART attributes, while taking into account the unbalanced distribution of the training data [170]. In this sense, LSTMs seem to be especially up to the task, as they have been shown to be sensitive to long-term dependency and the dynamical nature of time-series data across a variety of domains [164, 204, 109].

Thus, we follow recent work on predictive models for HDD failure using neural-networks [10, 20], and propose a LSTM based model for HDDs' health level prediction task, which automatically identifies the HDD health levels.

2.2 Anomaly sound detection

A lot of work discussing deep learning approaches for anomaly detection, especially for predictive maintenance task, has been recently presented (see ([34, 108, 160, 59, 36]) for more details).

In particular, two interesting approaches for industrial applications are those proposed by ([28]) and ([81]). The first work designed a methodology that jointly performs K-means and Self-Organizing maps (SOMs) as a starting point on which a local Probability Density Distributions (PDD) algorithm has been applied for detecting anomalies in the monitoring of turbine's bearing temperature in a hydropower plant. A Convolutional Long Short-Term Memory (CLSTM) model has been then use in the second work for fault detection in rotating machinery relying on a combination of statistical and different features, extracted by performing Fast Fourier Transform (FFT) and Continuous Wavelet Transform (CWT) of raw signals.

In the last years, different approaches based on encoder architectures ([133, 132, 149]) have been developed for anomaly detection. In ([5]), the authors proposed an autoencoder-based approach using power consumption of industrial laundry assets and bearing vibrations data in order to compute an anomaly degree score of each instance. The reconstruction errors are further fed into a discriminator with the aim to discern anomalies instances from normal ones. In turn, ([77]) developed an unsupervised detection model based on the Long Short Term Memory autoencoder. The model has been trained to reconstruct input sequences for each timestamp using a sliding-window method, which are successively labeled as normal or anomalous on the basis of its reconstruction error. Finally, a majority voting mechanism has been

applied for making the final decisions, being each timestamp labeled as many time as the sliding window length.

In ([193]), the authors developed a fault attention generative probabilistic adversarial autoencoder (FGPAA) to automatically find low-dimensional manifold embedded in high-dimensional space of the signal. Yin et al. in ([202]) proposed an integrated model of the Convolutional Neural Network (CNN) and recurrent autoencoder for anomaly detection in IoT time series. In ([40]), the authors designed an unsupervised anomaly detection system for industrial robots based on a sliding-window convolutional variational autoencoder (SWCVAE), which realizes real-time anomaly detection by coping with multivariate time series data. Another strategy to deal with the anomaly detection task concerns a combination of deep and ensemble learning. A novel method, called ensemble deep autoencoders (EDAEs), has been then proposed in ([163]) for fault diagnosis of bearings. In particular, the ensemble of multiple deep autoencoders aims to overcome the low generalization ability of individual ones.

Indeed, the existing anomaly detection systems used in the industrial applications depend on the properties of sensors used to monitor an industrial machine. The majority is focused on visual anomaly detection systems, which have some drawbacks (i.e. illumination or occlusion by objects), which strongly affect the performance of the system, especially in terms of computation power needed to achieve high real time performances.

To overcome these issues, the Anomalous Sound Detection (ASD) systems [131, 187] have been developed using acoustic data features that are often processed, like signals representing air pressure values or spectrograms, often in the Mel scale[2], for supporting neural network training process.

[2]In Mel-Scale the entire frequency spectrum is separated into x evenly spaced frequencies (bins), where 'evenly spaced' means that the distance on the frequency dimension approximates the human auditory system's response more closely than the linearly-spaced frequency bands used normally in spectrograms.

In this context, [22] compared the performances of Convolutional LSTM autoencoder (Conv-LSTMAE) and sequential Convolutional Autoencoder (CAE) using sounds retrieved from Youtube videos recorded during industrial manufacturing processes. The final decision about the generation of an alert due to the possible presence of anomalies, is made on the comparison between the reconstruction error of each spectrogram and a pre-defined threshold. In [136], the authors propose a convolutional autoencoder which receives in input frames obtained by segmenting mel spectrograms extracted from an audio dataset for classifying the anomalies into three abnormal categories that are manually defined. [122] investigated two different approaches for unsupervised anomaly detection: one based on One-Class Support Vector Machine (OC-SVM) and one based on autoencoders trained in unsupervised way. In [90], a *few-Shot learNIng with ensured true-PositivE-Rate* (*SNIPER*) architecture, whose goal is to maximize the true positive rate (TPR) on already observed anomalies. In particular, it combines the anomaly score produced by an autoencoder trained in unsupervised manner and a new ad-hoc similarity score computed by a specific anomaly detector $\mathscr{S}$ on the basis of a comparison between input samples and memorized anomalous sounds. The authors, further, build $\mathscr{S}$ using an approach based on *VAE* to overcome the problem of the absence of sufficient overlooked anomalous data. In [148] two deep learning models have been proposed based respectively on a dense and convolutional architectures fed with spectrograms aiming to minimize the reconstruction error between inputs and outputs. For the former, the encoder and decoder networks consist of four fully-connected layers, followed by Batch Normalization and ReLU as the activation function while in the latter the encoder and decoder networks are composed by convolutional layers with Batch Normalization and the ReLU activation function after each convolution. A LSTM-based autoencoders architecture for ASD task has been proposed in [80], that encodes information using LSTM layers, whose output is, firstly, processed by the LSTM-based bottleneck layer and, successively,

reconstructed by a stack of sequential LSTM layers. Another interesting approach has been proposed by [72], that relies on a Transformer-based and Conformer-based autoencoder for ASD, that extracts sequence-level information from whole audio inputs using a self-attention mechanism to perform sequence-to-sequence processing.

Table 2.2 summarizes the state-of-the-art approaches, also focusing on application domains and their pros and cons.

As can be noted, all the approaches proposed above differ on the basis of the way autoencoders are built. In fact, the main idea is always to train an autoencoder to reconstruct the normal audio samples provided in input and then use it in order to detect anomalies on the basis of reconstruction error.

Nevertheless, there is additional information related to the equipment which are potentially useful in order to improve the patterns learned from the model. One of the most important information concerns the ID related to the equipment which the sound clips are retrieved. In this regard, our main idea is to use the information related to the particular equipment identifier (*ID*) from which the sound clips are retrieved in order to influence autoencoder behaviour and its capability to reconstruct the inputs.

For this reason, we propose a modular framework for different anomalous sound detection tasks (i.e. predictive maintenance or surveillance), relying on two phases: the former jointly analyzes audio clips based on mel-spectogram analysis and ID equipment information through a *one-hot* encoding method for extracting features that are, successively, used in the latter to train an *ID Conditioned Network*. In particular, this module is composed by an autoencoder and ID conditioning neural network, properly trained in a joint manner with a loss developed ad hoc.

Furthermore, the proposed framework is modular because it is possible to integrate any type of autoencoder replacing the encoder and decoder blocks as well as using other types of deep networks by customizing the interaction with the *ID conditioning network*.

Finally, our framework enables real-time analysis by analyzing continuous audio stream coming from different audio sources on the basis of a customized sliding windows.

Application Domain	Approach	Ref.	Pros	Cons
Industrial Laundry/bearings vibrations data (MFPT)	Dense AE	[5]	The discriminator avoids an iterative process of choosing the threshold.	Need to choose accurately the synthetics features needed to extract sensor data trends over time.
Production line chambers with presence sensors	LSTM AE	[77]	Capacity to capture temporal dependencies in multivariate sensor data.	Tuning needed to choose the sliding window length and hop-size.
Fault diagnosis of bearings (CWRU)	Ensemble Deep AE (EDAEs)	[163]	Overcome the low generalization ability of individual autoencoders.	High complexity of the architecture due to the number of autoencoders that must be trained.
Joint currents and joints angles of a pick-and-place robot	Sliding Window Conv VAE (SWCVAE)	[40]	Capacity to detect spatial and temporal anomalies in data, also in real-time.	Tuning needed to choose the sliding window length and hop-size.
Industrial manufacturing processes sound retrieved on-line	Con LSTM-AE and Seq-Conv-IAE	[22]	Capacity to detect temporal dependencies in data.	Longer training time due to the convolutional nature of the network
Audio data from Surface Mount Device assembly machine	Conv-AE	[136]	Capacity to extract features autonomously from audio spectrograms.	Tuning needed to choose the sliding window length and hop-size to extract spectrograms from a audio.
	LSTM AE	[137]	Capacity to detect temporal dependencies and to allow fast training.	Tuning needed to choose sequence length and hop-size used for anomaly detection.
Sounds recorded from compressor, engine, compression pump, electric drill and a condensing unit of an air-conditioner (DCASE 2016 Dataset)	VAE + AE	[90]	Capacity to never overlook an observed type of anomaly twice thanks to a component which memorizes anomalies not recognized in the past, even if there are not too much samples of them	Augmented overall architecture complexity. Necessity to pre-process audio clips related to working machine states.
Sounds recorded from industrial machines, like pumps,valves, slide rails and fans (DCASE 2020 Dataset)	Dense AE	[148]	Simplicity of the overall architectures.	Necessity to tune parameters regarding audio clips transformation process into spectrograms.
	Conv-AE	[148]	Capacity to capture temporal dependencies using audio spectrograms as inputs.	Longer training time regarding the convolutional version.
	LSTM AE	[80]	Capacity to capture temporal dependencies on audiospectrograms segments.	Tuning needed to choose the number of timesteps and the number of features.
	Transformed and Conformer based AEs	[72]	Attention resolves the inability of seq2seq architecturesto retain longer sequences, focusing on valuable part of the sequence.	Architecture complexity in terms of parameters due to the presence of attention layers.
	Conditional Dense AE, Conditional Conv-AE and Conditional LSTM AE	[72],[84],[89]	Conditioning autoencoders can recognize audio samples recorded from different machines of the same type using external information, like identifiers.	Necessity to add external layers to process these identifiers, which should also be transformed in a compatible format. A new training loss must be defined to use the identifier information

Table 2.2: Related works summary

3 The need of interpretability

The past decade has seen the rise of deep learning techniques applied to several tasks in a variety of domains (for a recent review, see [162] a.o.).

In particular, the robust coupling of artificial intelligence (AI) and energy domain knowledge proved to be effective in achieving relevant energy saving in buildings by exploiting a variety of predictive-based energy management solutions, such as energy consumption forecasting [54, 175, 154], anomaly/fault detection and diagnosis in buildings and energy systems [74, 142], advanced energy benchmarking [62, 104], load profiling [52, 110]. Predictive analytics is de facto considered a cross-sectional application of AI for enhancing energy management in buildings [207], and until now its use has been associated to the need of achieving the highest accuracy as possible of predictions at the basis of decision-making process.

Such use is extensive in many domains including the previously analyzed domain concerning the estimation of the lifetime of an HDD. In general, it is difficult to analyze the entire process of disk health deterioration and predict when disk drives will fail in the future. Due to the common lack of diagnostic information on disk failure, most approaches rely on SMART (Self-Monitoring, Analysis and Reporting Technology) data and explore statistical analysis techniques to identify the onset of disk degradation. It is important

to note that the decision-making process used by most of these techniques to arrive at a specific result is often opaque to human users.

For these reason understanding *why* a certain prediction is provided by a black-box model is becoming more and more an essential feature of predictive analytics in several modern contexts, especially when the decisions of an AI system are required to be transparent and fair (e.g, for certification aims, substitution of component). Generally speaking, such task is the main goal of *eXplainable Artificial Intelligence* (XAI) [69], which offers new opportunities for successfully embedding AI-based solutions in industrial applications where explanations of the data-driven AI models is often a mandatory requirement.

In the last decade, XAI has become for AI researchers an emerging and very challenging topic whose meaning and usefulness can be summarized through several key aspects, as reported by the first significant published studies [150, 23]. Certainly, the two most relevant concepts concern the ability of an AI system to explain its decisions in intelligible terms to humans [50, 67] (i.e., *Explainability*) and, the ability to identify the set of characteristics that mostly contribute to making a decision [1] (i.e., *Interpretability*). Therefore, XAI supports the definition of more explainable models, maintaining a high level of performances, allowing human users to understand and trust AI-based systems.

During the last years, different XAI methods and strategies have been proposed. Usually, they can be classified, according to the granularity of the related analysis, into *local* (understand a single prediction) and *global* (understand the model behaviour) approaches.

Local XAI methods aim to explain a black-box model outcome on the basis of local information around the prediction. For instance, [16] have proposed to measure local gradients to exactly identify in which ways changing the input affects the prediction. Similarly, [151] presented a feature importance method, which computes the differences between a prediction and the

obtained solution. Finally, the model-agnostic (LIME) method proposed by [150] is based on an algorithm that faithfully explains the predictions of any classifier, by approximating it locally with a fully interpretable model.

The techniques above summarized focus on local explanations to achieve an overall explanation of a model. On the other hand, other techniques explicitly try to build global explanations. The most popular methods of this latter typology rely on features importance to explain tree-based models: the global Mean Decrease in Impurity (MDI) approach [27] – which exploits splits' number of samples – and the Mean Decrease in Accuracy (MDA) technique [112] – which computes a model mean increase error on the basis of a random permutation of the features.

Directly related to the task about HDD, Xie et al. [197] presented a XAI system explicitly designed for disk failure prediction, which is able to infer the prediction rules learned by a model, in order to make the failure prediction process transparent.

Recently, also in the energy domain the concept of XAI is being introduced. [55] defined a comprehensive methodology to explain and evaluate building energy performance models. Whereas, [11] introduced a methodology that enhances the existing building benchmarking process of Energy Star by increasing accuracy and providing additional model output processing to help explaining why a building is achieving a particular energy performance score. [2] developed an explainable and interpretable Deep Neural Network (DNN) model for a Guideless Irregular Dew Point Cooler (GIDPC). The SHapley Additive exPlanations (SHAP) method was used to assess and interpret the contribution of the operating conditions on performance parameters of the system. Also [98] employed a XAI-based process for improving the interpretability of a prediction model. In particular, the study focused on the forecasting of solar PV system generation and on the use of XAI tools, such as LIME, SHAP, and ELI5, for model explanations. A further promising application of XAI in the energy and buildings field concerns with fault detec-

tion and diagnosis (FDD). In particular, XAI offers the opportunity to explain which are the boundary conditions related to the detection of a fault/anomaly during system operation and most of all provides a readable interpretation about its diagnosis. As a reference, an interesting application of XAI was proposed by [115] for enhancing FDD analysis based on machine-learning algorithms (i.e., support vector machine, artificial neural network) conducted on building Air Handling Unit (AHU).

4 Adversarial perturbation in AI application

With the spread of Deep Learning and in particular of CNNs it was only a matter of time before researchers started analysing aspects associated with their decision making process and weakness. The former is what laid the foundation for XAI [69], a set of methods and algorithms intended to shed lights on the motivations that made an AI model to take a given decision. Similarly, Adversarial Perturbations (APs) [3] are a good example for the latter, with their ability to mislead a target CNN.

More in detail, the aim of and AP is to move the target sample beyond the model decision boundary (figure 4.1). There are two possible ways of doing it:

- **Gradient-based** methods exploit the gradients information with respect to the input in order to determine the best perturbation to add to the target sample to mislead the target CNN;
- **Non-Gradient-based** that changes some values in the input data until a fitness function says that the obtained perturbation is able to mislead the target classifier.

Since, by definition, an adversarial perturbation should be as invisible as possible, the hardest part is to determine a small (imperceptible) noise that is

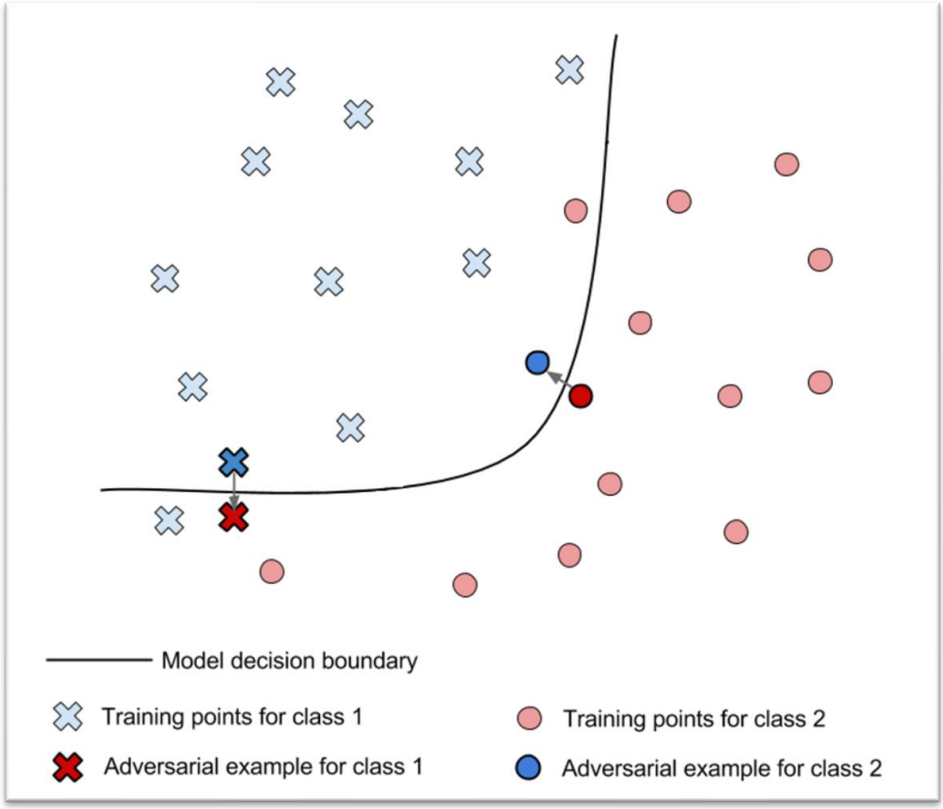


Figure 4.1: Illustration of an adversarial perturbation attack performed in the case of a 2D features space. Given a classifier (and thus identified its decision boundary), an adversarial perturbation attack aims to move correctly classified samples across the decision boundary. In the example, a dark red dot and a dark blue cross, previously corrected classified (since laying in the right subspace identified by the decision boundary) are "pushed" (i.e. modified by adding some carefully crafted adversarial noise) just enough to cross the decision boundary. The effect is that now the perturbed samples are misclassified by the target model. Image adapted from another work[3].

still able to mislead the target classifier. Of all research areas for which deep neural networks have been demonstrating overwhelming performance, *com-*

[3] https://pod3275.github.io/paper/2019/08/02/KDwithADVsamples.html

puter vision is still one of those that mostly catches the interest of researchers. For this reason, of all deep architecture, Convolutional Neural Networks (CNN) are among the most popular and used, and, as a consequence, the most "attacked". Indeed, in 2013 it has been shown that given a target CNN, it is possible to craft samples able to arbitrarily mislead it [176]. It is worth noting that the authors not only proved the existence of blind-spots in CNNs, but also introduced a method for the generation of adversarial samples based on the Limited Memory Broyden–Fletcher–Goldfarb–Shanno (LM-BFGS) algorithm and on the network loss function value.

Two years later, the Fast Gradient Sign Method (FGSM) laid the foundations for attacks exploiting the network gradient to generate an adversarial sample [65]. In short, given a victim CNN and a clean input image, the FGSM multiplies a user-defined standard deviation ε by the sign of the prediction gradient (with respect to the input class) to generate an additive perturbation. The FGSM Iterative Method [97] was proposed to perform a semi-automatic tuning of the ε value by using an iterative procedure. In this case, a small magnitude perturbation is calculated and applied several times, instead of applying a stronger noise in a single shot. DeepFool [128] made a step further by introducing an efficient iterative approach exploiting the network gradient of a locally linearized version of the loss. This allows generating a sequence of additive perturbations that move the clean sample on the edge of the classification boundaries. Finally, Momentum Iterative Method [49] introduced a momentum-based iterative FGSM like approach, resulting in a procedure able to stabilize the update directions and thus to escape from poor local maxima determined during its execution. Meantime, other researches focused their attention on the development of adversarial perturbation techniques aimed to surpass some limitations, rather than only looking for performance. The Carlini Wagner L2 method [32] proposed to construct the adversarial samples using the same basic idea of [176], but with some significant improvements considering i) three possible targeted attack (best, worse and average case),

ii) three different distance metrics between the clean and the adversarial sample (L_0, L_2, L_∞) and iii) different optimization algorithms. Finally, *the algorithm performs a greedy search to determine a discrete perturbation, to make it robust to rounding operations performed in the* $[0-255]$ *value image representation*. With the aim of modifying as few pixels as possible, the Jacobian-based Saliency Map Attack (JSMA) performs a greedy iterative procedure that i) evaluate a saliency map based on the target class classification gradient ii) to determine the most influencing pixels [135]. The algorithm iterates until the adversarial sample is generated or the number of modified pixels exceeds a fixed threshold (meaning that the attack is failed). With the same aim, the One-Pixel Attack [172] defined a totally different manner to reach the solution. The idea is to modify a very reduced number of pixels without having any prior information on the network. On a different side, the Feature-Opt method [156] approaches the adversarial perturbation problem focusing on the image representation at the internal layers of a CNN instead of considering the output of the classification layer. The aim is to generate an adversarial sample that not only causes an erroneous classification, but that also has an internal representation closer to the target class rather than to the clean one. Nowadays, many researchers are introducing XAI methods as part of the development loop of machine learning models, in applications ranging from biomedical to security. For example, in [140] the authors used Grad-CAM and Guided BackPropagation (GBP) [168] to analyse the clinical coherence of the features learned by a CNN for automated grading of brain tumours in MRI. More recently, Chen et al. [38] used Gradient Class Activation Mapping (Grad-CAM) [161] to generate the explainability map of a model, showing that it focused its attention in high-dimensional bands excited by structure resonance.

Focusing on adversarial perturbations, the vast majority of the works target the development of new perturbation strategies or the design of attacks against some critical applications. Nonetheless, in some previous works, we

showed that APs can also be leveraged to increase fairness and security with AI. In particular, in [119] we showed how to design an adversarial patch able to prevent ethnicity recognition in automatic face analysis, while in [120] we attacked a fingerprint authentication system to shed lights on light-heartedly use of CNN in security-critical applications.

As illustrated by the aforementioned example, XAI and APs have the potential to help the improvement of both accountability and reliability of CNNs. Some authors tried to make a step further by "contaminating XAI" with APs and vice-versa. In [95] the authors exploit XAI to make OnePixel [173] (a famous AP algorithm) more robust. On the other hand, in [57] the authors proposed a way to detect APs by means of XAI, by using the latter as a way to extract a "fingerprint" of the image.

Besides these experiments, the relation existing between XAI and APs is extremely of interest, since it can help increase AI reliability and trustworthiness of AI-based systems. On this line, in [203] the authors analysed the propagation of adversarial noise through CNNs layers, by measuring the similarity between the feature map of clean and of adversarially perturbed images. Similarly, in [46] the authors analyse how the "strength" of the noise affects the prediction accuracy. Finally, in [78] the authors demonstrated how XAI and APs are connected by a generalised form of hitting set duality, also proposing an algorithm to move from one to the other.

The latter three examples are the closest to this paper. However, each of them lacks some aspects, such as a "human interpretable" outcome [203], a quantitative analysis of the results [46] or experiments and proofs made on more realistic datasets [78].

Part II

AI techniques for predictive maintenance

5 HDD health assessment

5.1 Introduction

Hard disk drives (HDD) are nowadays a primary type of storage in data centers. Due to this pervasive use, HDD failure is now one of the main factor for data center downtime, unavailability, and data loss — with obvious effects on overall business costs and reliability. Thus, an important line of research has focused on developing robust predictive maintenance techniques, to reliably predict HDD failures, and timely adopt maintenance strategies to increase the *Remaining Useful Life* (RUL) of these drives.

HDD *maintenance* actions include inspection, testing, repair, and replacement — basically any action aimed at preserving the quality of the system while improving its availability and extending its life. Obviously, in order for maintenance costs for large distributed systems to remain effective, such actions have to be properly organized according to well-defined strategies. Because of the dimensions of most data-centers, properly scheduling these actions is not a trivial problem, and maintenance policies have become a fruitful topic of research [71, 170, a.o.].

Recently, the dissemination of condition monitoring equipment and the development of methods for deterioration prognosis and residual life estimation

have shifted the interest of most practitioners towards *predictive maintenance* techniques [165]. Predictive maintenance seeks to anticipate system failures in order to plan timely interventions on the system. In such frameworks, decisions are based on a system's online health prognostic information (i.e., information about the system future state), rather than on the online diagnostic information (i.e., information on the system current state) [198].

Due to this shift in focus towards predictive systems, machine learning approaches have been gaining increasing popularity [129, a.o.]. In particular, models based on *Self-Monitoring, Analysis and Reporting Technology* (SMART) have shown high accuracy levels by relying on internal attributes of HDDs as indicators of drive reliability.

Importantly, most prediction systems analyze HDD failure as a binary classification task, simply distinguishing between good hard drives and those at high risk of failure. However, the complexity of the prediction task and the unbalanced nature of the data used in training have shown how these models' performance goes down significantly when they are tested on datasets representative of real-world environments [15].

Moreover, it has been suggested that, due to the non-linear pattern dynamics found in real-world systems, the ability to model the proximity of possible failures in time (and not just the chance of failure) could fundamentally change the way maintenance strategies are optimized [20]. Following this intuition, in this thesis has been implemented a HDD health level prediction task that models the health degree of a HDD unit according to its estimated time to failure.

In particular, a framework that leverages a *Long Short Term Memory* model for HDDs' health level prediction has been designed. Its main characteristics can be summarized as follows:

- it automatically identifies the HDD health levels by considering the distribution of SMART attribute values over the time;

- it improves prediction accuracy by considering sequential dependencies in SMART attributes;

- it relies on an automated strategy for identifying the number and size of hard drive's health degree settings.

First, an automated step for HDD health level definition using a Regression Tree (RT) algorithm has been implemented. Then LSTM networks [75] is exploited to model the sequential dependencies between SMART attributes over time. LSTMs are particularly appropriate for this task, as they were explicitly designed to model long-range dependencies in temporal sequences.

While LSTM approaches to RUL estimation have been successfully explored in the past [20, 205, 93], proposed methodology proves more flexible to the highly complex nature of the data by relaxing the predefined health degree levels traditionally used in the literature in favor of dynamically generated ones. Identifying HDD health levels automatically allows to take full advantage of the information available in the training sets, and to obtain finer-grained predictions beyond what would be available through the simple binary classification task used in current systems.

In order to support the efficacy and practicality of the proposed model in real-world scenarios, its performance over two data-sets has been evaluated. For each hard drive in used data-sets, the attribute sequences from specific time windows (TW) of varying size (from 4 to 48 hours for the first data-set and from 5 to 14 days for the second ones) has been extracted, and how this approach outperforms a variety of models and methods in the previous literature [198, 103, 208, a.o.].

5.2 Methodology

Since hard drives often deteriorate gradually rather than abruptly, we argue that temporal analysis methods should be employed to model the sequential

nature of the dependencies within SMART attributes over time. Thus, an approach to estimate the Remain Useful Life (RUL) of a HDD has been proposed, by automatically identifying specific health conditions on the basis of SMART attributes values. This methodology is grounded in three main steps:

- **Hard drive health degree definition**: in which a status (or health level) is defined for each hard drive according to its time to failure;
- **Sequences extraction**: in which sequences in a specific time window are extracted for each hard drive;
- **Health Status assessment through LSTM**: in which a health level is associated to each temporal sequence.

In what follows, are described each component of proposed framework in detail.

5.2.1 Health degree definition

Hard drive failure in real-world data centers is a gradual process of deterioration. To address the gradient nature of the decay, the health status (or level) of a HDD according to its time before failure has been defined. Differently than [198], an automated step for HDD health level definition has been implemented.

More specifically, in this step has been considered only the hard drives that are going to fail, introducing for each of them an additional feature representing the *time before failure*. The data-set reports, for each hard disk, the temporally sorted sequence of SMART attributes with a specific sampling period. Denoting with m_j be the number of samples for the hard disk j, it is possible associate each sample with an index i from 0 to $m_j - 1$, representing the number of samples that follow it in the sequence describing hard disk failure. As a consequence, the sample with index $i = 0$ is the last sample

before failure. In Figure 5.1, *Time-to-failure* is the feature representing the time before failure for each hard drive whose meaning depends on sampling period while f_1, f_2, ..., f_n are the SMART attributes.

The idea is to build a Regression Tree (RT) for each SMART attribute f_i with $i = 1,2...n$, having the feature representing the time before failure as predictor and f_i as the numeric target value. Among all the resulting trees (one for each SMART attribute f_i), the one with the highest performance is selected, showing the attribute most temporally dependent. Since the selected Regression Tree (RT) presents splits only on the feature *Time-to-failure*, the latter is used to distinguish hard drive health levels according to time before failure. Figure 7.2 is an example of hard drive health levels identification by means of the Regression Tree algorithm. Each internal node represents a split on the feature Time-to-failure, resulting in the definition of four health degree levels.

As mentioned above, the automated step for health-level definition only considers those hard drives that are going to fail. A different level or status should be assigned to samples belonging to hard drives that will not fail since they have been excluded in this step. More specifically, the samples belonging to the hard drives that will not fail are labelled as *Good*.

5.2.2 Sequence extraction

To explore the temporal dependencies within the SMART features periodically collected for each hard drive, feature sequences in specific time windows (TW) have been extracted.

Let w and a^t be the time window size and the set of SMART features $(f_1, f_2...f_n)$ at time t, respectively. Proposed model aims to predict hard drive health status at time $t+1$ ($Hs(t+1)$) considering the sequence (a^{t-w+1}..., a^{t-1}, a^t). For each a^t, the health status $Hs(t)$ is defined as the Regression Tree built according to in Figure 5.4, and the feature sequence for each hard drive at time t is extracted considering the $w-1$ previous samples (cf. Figure

Hard drive ID	f_1	f_2		f_n	Time To Failure
⋮					⋮
1	value	value	.....	value	3
1	value	value	.....	value	2
1	value	value	.....	value	1
1	value	value	.....	value	0
⋮					⋮
2	value	value	.....	value	240
2	value	value	.....	value	239
2	value	value	.....	value	238
⋮					⋮
n	value	value	.....	value	2
n	value	value	.....	value	1
n	value	value	.....	value	0

Figure 5.1: *Time to failure* is a feature representing the time before failure for each hard drive sample, while f_1, f_2, ..., f_n are the SMART attributes.

5.3). Each sequence results in a bidimensional array of size $w \times n$, where n is the number of SMART features considered. For each hard drive, sequences are extracted with a stride of one. It follows that $m_j - w + 1$ sequences are extracted for each hard drive, where m_j is the number of samples for the disk j.

For each sequence (a^{t-w+1}..., a^{t-1}, a^t), the hard drive's health level is defined by the health level of the set of features a^{t+1}. The result of this step is a sequence-based data-set — a set of bidimensional arrays, each associated to a health level representing the hard drive's health condition between two consecutive samples (i.e., a^t and a^{t+1}).

5.2.3 Health Status assessment through LSTMs

Based on what established in the previous sections, it should be clear how hard drives' health level prediction consists in a multiclass classification task, where each feature sequence is assigned to one of the classes (health levels) introduced in Section 7.2.1.

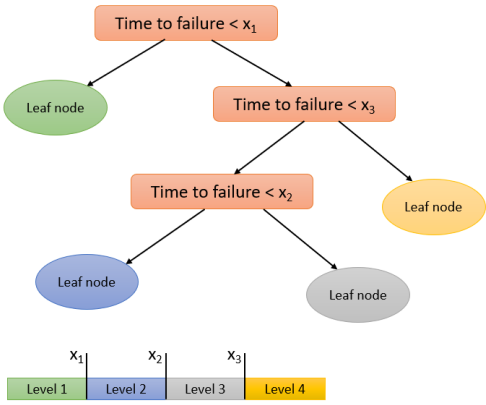


Figure 5.2: Identification of hard drive health levels by means of a Regression Tree algorithm. Each internal node represents a split on the feature Time-to-failure, resulting in the definition of four health degree levels.

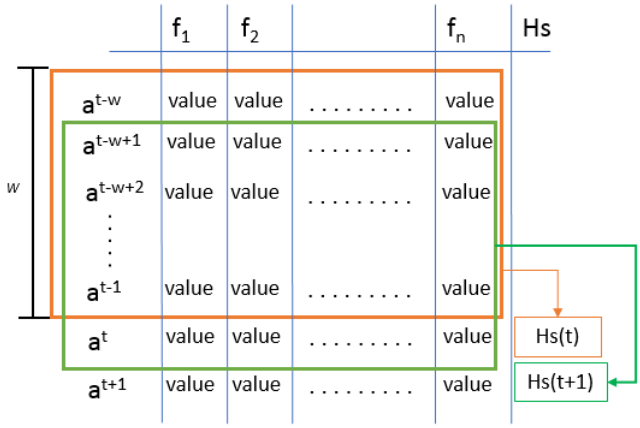


Figure 5.3: Sequence extraction step for a single hard drive.

Because of the sequential, gradiently changing nature of the SMART features, it is important that designed model is able to capture dependencies across features over time. Long Short Term Memory networks (LSTMs) are extension to recurrent neural networks, explicitly designed with the purpose of learning long-term dependencies [75].

In the proposed framework, the input to each LSTM layer is a three-dimensional data structure of size $z \times w \times n$, where:

- z is the the total number of sequences (or the batch size at each iteration);
- w is the size of each sequence — that size of a time window, in terms of time steps;
- n is the total number of features describing each time step.

Since the percentage of failed hard drives is often small compared to the percentage of good hard drives, the sequence extraction step may result in an unbalanced data-set with the majority of sequences belonging to the *Good* level. As a consequence, a data balancing step is introduced, so that the input to the network is a set of balanced data.
In particular, the sequence-based data-set is balanced by replicating the sequences belonging to the minority classes. Sequences replication is an efficient balancing strategy that avoids the polarization of the classification model on a single class without creating synthetic data or reducing the data-set size by sampling the instances beloging to the majority class. The implemented classification network has two stacked LSTM layers with 128 units, followed by a single dense layer.

5.3 Experimental Evaluation

This section aims to evaluate the effectiveness of the proposed approach. The prediction performance of the model have been tested on the two different SMART data-set, and then compared with those of three other methods explored in the literature: a Classification Tree model, a Random Forest model, and a model based on Multiclass Neural Networks.

5.3.1 Baidu data-set

The first SMART data-set used for the analysis was collected from a single running data center of Baidu Inc [4], and contains samples from 23,395 disks. All samples refer to an enterprise-class disk model of Seagate (ST31000524NS). Each disk was labeled *Good* or *Failed*, with only 433 disks in the failed class and the rest of disks (22,962) in the good class. As the ratio of *Good* vs. *Failed* disks is approximately 1 : 50, so this a highly unbalanced data-set.

SMART attribute values were read per-hour for each disk. For *Good* disks, every sample collected over a week is kept in the data-set, so every good disk is associated to 168 samples. For *Failed* disks, samples in a longer time period (20 days before actual failure) are saved, resulting in a maximum of 480 samples per disk. Note though that a specific disk could actually be associated to a smaller number of samples, if it failed during the 20 days of operation since the start of data collection. Finally, each entry in the data-set contains the 14 features listed in Table 5.3.1, with values for every attribute value scaled to the same interval $[-1, 1]$.

5.3.2 Backblaze data-set

The Backblaze data-set [5] contains daily data collected from 50,984 hard disks. Each sample consists of information about timestamp, disk serial number, disk model, disk capacity and values for 90 SMART attributes. Moreover, for each sample the feature *failure* is set to 0 if the drive is alive while it is set to 1 if the disk has been replaced the following day. All samples before February 2014 have been excluded, since more than 70% of SMART parameters had not been collected. We focused on samples belonging to Seagate ST4000DM000, since it is the most populated model in data-set (29,878 disks in total; 29,083 *good* disks and 795 *failed* disks). Among all the SMART attributes, the ones

[4] http://pan.baidu.com/share/link?shareid=189977&uk=4278294944
[5] https://www.backblaze.com/b2/hard-drive-test-data.html

SMART ID #	Attribute Name
	Serial Number
	Label
1	Raw Read Error Rate
3	Spin Up Time
5	Reallocated Sectors Count
7	Seek Error Rate
7	Power On Hours
187	Reported Uncorrectable Errors
189	High Fly Writes
194	Temperature Celsius
195	Hardware ECC Recovered
197	Current Pending, Sector Count
5	Raw Value of Reallocated Sectors Count
197	Raw Value of Current Pending Sector Count

Table 5.1: SMART attributes as features

that are shared between the Backblaze and the Baidu data-sets (see Table 5.3.1) have been selected. However, also the SMART attribute with ID 195 (*Hardware ECC Recovered*) has been excluded, since no sample had a value associated to this feature. Finally, the values for every SMART attribute were scaled to the interval $[-1, 1]$.

5.3.3 Preprocessing

Data preprocessing consisted of two main steps:

Features Selection

The feature representing disk capacity have been excluded in the Backblaze data-set. Importantly, the attributes *Label* for Baidu, *failure* for Backblaze, and *Serial Number* for both data-sets are necessary in order to distinguish

between failed and good hard drives and to create sequences for each hard drive. However, they are not taken into account during sequence classification.

For good hard drives, each sample was associated to the health degree level *Good*, while for failed hard drives, their remaining functioning time depends on the number of samples collected for said device.

Health degree computation

The way health degree levels are computed differs between the Baidu and the Backblaze data-sets.

- **Baidu data-set.** For *failed* disks, stored samples correspond to a period of 20 days before actual failure. Thus, we propose a model for predicting hard drive health status 20 days in advance. As mentioned in Section 7.2.1, hard drive health degree definition depends on the splits of the selected Regression Tree (RT) model on the feature Time-to-failure. Recall that the Baidu data-set presents samples read per-hour for each disk. For this reason, the feature Time-to-failure *Hour to failure* have been renamed. The regression tree built with the feature *Raw value of Current Pending Sector Count* reported in Figure 5.5 has been selected. Specifically, the selected Regression Tree suggests distinguishing 6 different levels of *health degree* for hard drives that will fail. We then introduced a different level for those hard drives that will not fail. This results in the definition of 7 levels, named as follows:
  - **Good**: the hard drive works properly. This level is associated to samples belonging to hard drive that will not fail;
  - **Very Fair**: the hard drive works properly, but a fault or an error may have occurred;
  - **Fair**: the health status of the disk drive is fair and the hard drive is probably going to fail in less that 332 hours (approximately 14 days);

- **Soft Warning**: the hard drive is going to fail in less that 235 hours (approximately 10 days);
- **Warning**: the hard drive is going to fail in less that 179 hours (approximately 7 days);
- **Alert**: the hard drive will fail in less than 96 hours (approximately 4 days);
- **Red Alert**: the hard drive will fail in less than 47 hours (approximately 2 days).

The levels *Good* and *Very Fair* represent HDDs still in good health conditions. They both imply that a hard drive works properly, and thus time constraints are left unspecified. The other statuses are associated with different degrees of deterioration. Therefore, we classify a hard drive as being in a *Good status*, if its health level is characterized as *Good*, *Very Fair*, or *Fair*. A hard drive is classified as being in a *Failed Status*, if its health level is in *Soft Warning*, *Warning*, *Alert*, or *Red Alert*. Figure 5.4 shows the health degree settings for a single hard drive, as used in our evaluation. By assumption, the health degree level *Good* in never assigned to failed hard drives because of the high probability of errors or faults. Thus, the distinction between good and failed hard drives is preserved.

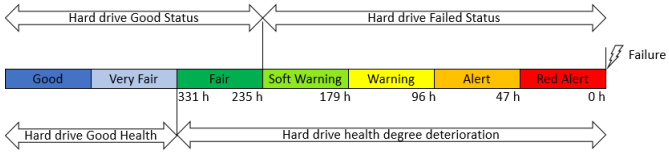


Figure 5.4: Hard drive's health degree settings for the Baidu data-set

- **Backblaze data-set.** Since samples were collected from February 2014 to December 2015, failed disks present a long observation period. As a consequence, there is a high probability that at the beginning of that

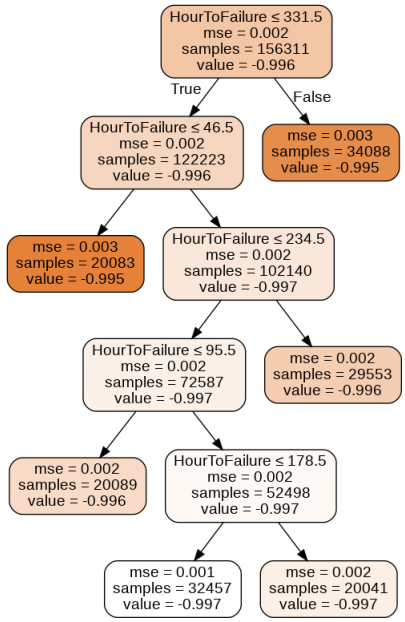
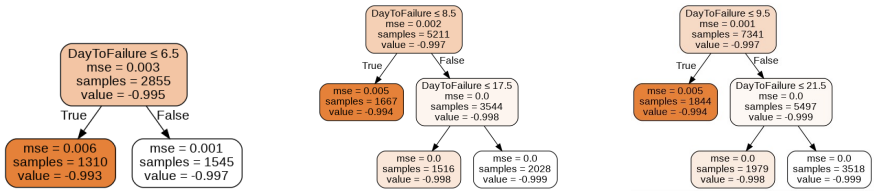


Figure 5.5: Identification of hard drive health levels by means of the Regression Tree algorithm built on the feature *RawCurrentPendingSectorCount* for the Baidu data-set. For each leaf node *mse* is the mean squared error of the samples, *samples* is the number of samples in that node, and *value* is the value of the SMART attribute f_i for the samples in that leaf. For each internal node, we report the condition on the feature *Hour to failure*.

period, the disks having samples for more than one year were *good disks*. It is not possible to determine the exact time of error occurrence. For this reason, we focused on the last q samples of each failed hard drives, where q is a *prediction window* that determines the period in which hard drive health status should be assessed. Specifically, proposed approach is able to predict hard drive health status q days before failure.

Different values for q have been explored, from 15 to 45 days. After choosing the value for q, hard drive health levels are defined according to Section 7.2.1. Since the Backblaze data-set contains daily samples for each hard drive, the feature *Time-to-failure* has been renamed *Day to failure*. We then selected the regression tree built with the feature *Raw value of Current Pending Sector Count*.

Figure 5.6 shows the regression trees obtained by selecting $q = 15$, $q = 30$ and $q = 45$. More specifically, Figure 5.6b and 5.6c suggest distinguishing 3 different levels of *health degree* for hard drives that will fail, while Figure 5.6a suggests 2 levels. Then a different level for those hard drives that will not fail has been introduced. When q is set to 30 or 45, the result is the definition of 4 levels, labelled *Alert, Warning, Very Fair* and *Good*. In turn, if q is set to 15, we define 3 levels, labelled *Alert, Warning* and *Good*. The levels *Good* and *Very Fair* represent HDDs still in good health conditions. Therefore, we classify a hard drive as being in a *Good status*, if its health level is characterized as *Good* or *Very Fair* while a hard drive is classified as being in a *Failed Status*, if its health level is in *Warning* or *Alert*. Figure 5.7 shows the health degree settings for a single hard drive, as used in our evaluation. Similary to the Baidu data-set, the health degree level *Good* is never assigned to failed hard drives due to the high probability of errors or faults.



(a) Regression tree obtained with *q=15*

(b) Regression tree obtained with *q=30*

(c) Regression tree obtained with *q=45*

Figure 5.6: Identification of hard drive health levels by means of the Regression Tree algorithm built on the feature *RawCurrentPendingSectorCount* for the Backblaze data-set. For each leaf node *mse* is the mean squared error of the samples, *samples* is the number of samples in that node, and *value* is the value of the SMART attribute f_i for the samples in that leaf. For each internal node, we report the condition on the feature *Day to failure*.

5.3.4 Experimental setup

As discussed above, an automatic step for hard drive health levels definition has been proposed, building a Regression Tree (RT) for each SMART attribute f_i, with the feature representing the time before failure as predictor. The selected trees (see Figure 5.5 and 5.6) consider the SMART attribute *Raw Value of Current Pending Sector Count* as numerical target value. The function measuring the quality of a split is the mean squared error (*mse*). The minimum number of samples required for leaf node in the Regression Tree is 20000 for the Baidu data-set and 1830, 1380, and 1200 for the Backblaze data-set with $q = 45$, $q = 30$ and $q = 15$ respectively.

Proposed approach has been evaluated with respect to three of the *sequence independent* methods most used in the literature: a Classification Tree (CT), a Random Forest (RF), and a Multiclass Neural Network (MNN) — a deep neural network with dense layers. These models are sequence independent because they generalize over input samples rather than sequences, and thus don't take the temporal dependencies of the SMART attributes into

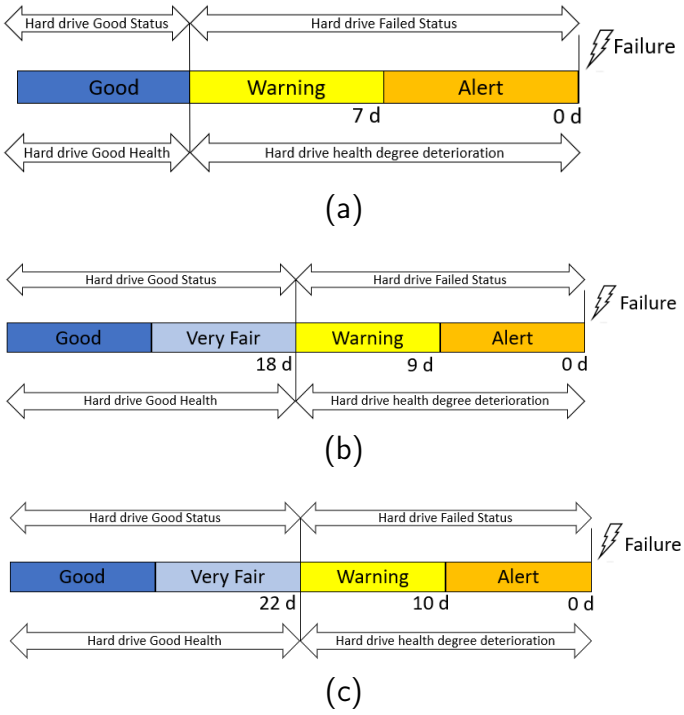


Figure 5.7: Hard drive's health degree settings for Backblaze data-set for *q=15* (figure 5.7a), *q=30* (figure 5.7b) and *q=45* (figure 5.7c).

account. Downstream of the parameters optimization, the number of trees for RF is set to 110 and 210 for the Baidu data-set and the Backblaze data-set respectively, and the minimum number of samples required for leaf node in CT is 20 for both data-sets.

The RT, CT and RF models have been implemented using the Python scikit-learn package, and Keras with Tensorflow as the backend for LSTM and Multiclass NN models.

As standard for this kind of techniques, the original SMART data-set was divided into training, validation and test sets. More specifically, we take the 70% of the data as training set, the 15% as validation set and the remaining data as test set.

During the training phase of the LSTM and Multiclass NN models, the maximum number of epochs is set to 150, and the batch size to 500. As an optimizer, we use Adam [86] with learning rate set to 0.001.

5.3.5 Performance Evaluation

Since each sequence is associated with one of the levels presented in Section 7.2.1, the HDD's health level assessment as defined in this approach is a multiclass classification problem with multivariate input variables.

The performance of proposed approach is first evaluated in terms of accuracy, precision, and recall. Since the distinction between good and failed hard drives is preserved in the labelling of the data-set, we express the results in term of accuracy on good sequences (ACC_G) and accuracy on failed sequences (ACC_F) — respectively, the fraction of sequences correctly classified as *Good*, and the fraction of sequence classified as the health levels suggested by the regression trees. We also consider the evaluation criteria introduced in [198], and measure the accuracy of classifying good and failed sequences for a tolerance of misclassification up to one health level (ACC_G^{TOL} and ACC_F^{TOL}).

Finally, we evaluate performance in terms of failure prediction, by assessing *failure detection rate* (FDR) and *false alarm rate* (FAR) for each model. This is done by considering the levels *Good*, *Very Fair*, and *Fair* as *Hard drive good statuses*; and the levels *Soft Warning*, *Warning*, *Alert*, and *Red Alert* as *Hard drive failed statuses* (see Figure 5.4 and 5.7). Intuitively, FDR is the fraction of failed sequences that are correctly classified as failed, while FAR is the fraction of good sequences that are incorrectly classified as failed.

5.4 Results

In this thesis we proposed a methodology to perform hard drive health status assessment exploiting the temporal dependencies of SMART attributes. In

TW SIZE [hour]	*Accuracy*	*Precision*	*Recall*	ACC_G	ACC_F	ACC_G^{TOL}	ACC_F^{TOL}	*FDR*	*FAR*
48	**99.80%**	**99.1%**	**98.9%**	**99.83%**	**93.17%**	**99.89%**	**98.31%**	**98.2%**	**0.2%**
36	98.78%	98.8%	98.7%	99.80%	91.89%	99.87%	97.45%	97.37%	**0.2%**
24	99.33%	98.9%	98.8%	99.66%	91.87%	99.74%	96.97%	97.64%	**0.2%**
12	98.71%	98.8%	98.6%	99.58%	78.06%	99.68%	90.54%	92.14%	0.4%
6	98.08%	98.3%	98.1%	99.43%	65.4%	99.59%	84.35%	86.8%	0.6%
4	97.74%	98.1%	97.8%	99.28%	60.29%	99.53%	82.47%	85.08%	0.6%

Table 5.2: Performance values for the LSTM models obtained by varying TW size on the Baidu data-set.

q [day]	*TW SIZE [day]*	*Accuracy*	*Precision*	*Recall*	ACC_G	ACC_F	ACC_G^{TOL}	ACC_F^{TOL}	*FDR*	*FAR*
15	5	95.88%	96.90%	95.10%	97.28%	66.56%	97.89%	98.08%	75.53%	2.82%
15	7	95.81%	97.10%	96.00%	97.02%	70.27%	97.93%	**98.45%**	79.34%	2.70%
30	5	94.54%	96.50%	94.60%	96.38%	56.07%	97.68%	88.30%	76.03%	2.73%
30	7	93.93%	96.80%	94.40%	95.59%	59.15%	97.07%	89.37%	80.70%	3.29%
30	10	95.25%	97.40%	96.10%	96.84%	61.84%	97.59%	91.35%	85.48%	2.73%
45	5	94.45%	96.70%	94.93%	95.95%	66.16%	97.80%	90.67%	78.30%	2.50%
45	7	95.82%	97.00%	95.85%	97.28%	68.34%	98.12%	89.37%	77.75%	2.17%
45	10	96.56%	97.72%	96.82%	97.71%	75.08%	98.36%	93.30%	84.18%	1.83%
45	14	**98.45%**	**98.33%**	**98.34%**	**99.21%**	**84.49%**	**99.40%**	96.65%	**91.48%**	**0.72%**

Table 5.3: Performance values for the LSTM models obtained by varying *prediction window (q)* and TW size on the Backblaze data-set.

Model	*Accuracy*	ACC_G	ACC_F	ACC_G^{TOL}	ACC_F^{TOL}	*FDR*	*FAR*
CT	97.01%	97.01%	58.94%	99.09%	**85.77%**	84.16%	1.00%
RF	**98.13%**	98.13%	**59.44%**	**99.82%**	85.65%	**85.36%**	**0.40%**
MNN	96.24%	**98.57%**	38.99%	99.14%	69.59%	73.03%	1.20%

Table 5.4: Results of sequence independent models on the Baidu data-set.

Model	*Accuracy*	ACC_G	ACC_F	ACC_G^{TOL}	ACC_F^{TOL}	*FDR*	*FAR*
CT	83.80%	83.87%	56.31%	95.63%	88.46%	63.58%	4.69%
RF	85.77%	85.77%	**71.75%**	93.68%	**93.82%**	**80.66%**	6.49%
MNN	**96.17%**	**99.15%**	39.78%	**99.88%**	69.20%	85.75%	**0.95%**

Table 5.5: Results of sequence independent models on the Backblaze data-set.

Metric	*Good*	*Very Fair*	*Fair*	*Soft Warning*	*Warning*	*Alert*	*Red Alert*
Accuracy	99.49%	75.10%	63.17%	41.39%	72.60%	47.44%	61.88%
Precision	100.00%	58.40%	50.90%	57.10%	46.60%	59.20%	60.10%
Recall	99.30%	75.06%	63.20%	41.40%	72.40%	47.30%	61.90%

Table 5.6: Results of best model on the Baidu data-set detailed by each class.

order to asses the effectiveness of the proposal, this section reports the perfor-

Metric	*Good*	*Very Fair*	*Warning*	*Alert*
Accuracy	99.21%	87.80%	78.10%	84.42%
Precision	99.90%	69.40%	64.70%	73.10%
Recall	98.80%	87.80%	78.10%	84.40%

Table 5.7: Results of best model on the Backblaze data-set detailed by each class.

Model	*Accuracy*	ACC_G	ACC_F	ACC_G^{TOL}	ACC_F^{TOL}	*FDR*	*FAR*
our approach	**98.45%**	99.21%	**84.49%**	99.40%	96.65%	**91.48%**	**0.72%**
K-Means [25]	93.90%	99.70%	61.76%	99.20%	88.10%	74.47%	2.30%
Smote [199]	97.37%	**99.86%**	51.87%	**99.88%**	83.68%	58.43%	0.80%

Table 5.8: Results obtained by varying different balancing methods on the Backblaze data-set

Model	*Accuracy*	ACC_G	ACC_F	ACC_G^{TOL}	ACC_F^{TOL}	*FDR*	*FAR*
our approach	**99.80%**	99.83%	**93.17%**	99.89%	**98.31%**	**98.2%**	**0.2%**
K-Means [25]	92.32%	99.91%	38.26%	99.95%	68.36%	68.77%	1.09%
Smote [199]	98.03%	**99.95%**	51.55%	**99.97%**	77.19%	76.40%	0.29%

Table 5.9: Results obtained by varying different balancing methods on the Baidu data-set

Dataset	*Method*	*ACC*	ACC_G	ACC_F	ACC_G^{TOL}	ACC_F^{TOL}	*FDR*	*FAR*
Backblaze	manual	97.54%	98.78%	75.26%	99.06%	93.31%	90.63%	0.89%
Backblaze	automatic	**98.45%**	**99.21%**	**84.49%**	**99.40%**	**96.65%**	**91.48%**	**0.72%**
Baidu	manual	99.15%	98.95%	92.38%	99.29%	97.74%	97.82%	0.37%
Baidu	automatic	**99.80%**	**99.83%**	**93.17%**	**99.89%**	**98.31%**	**98.20%**	**0.20%**

Table 5.10: Results obtained by varying different methods to define hard drive health levels

mance of proposed methodology, and a comparison with several state-of-art approaches.

Firstly, we report the results for both sequence dependent and sequence independent approaches. In particular, Table 5.4 and 7.4 show results of the LSTM based approach on the Baidu and Backblaze data-sets, respectively. Performance is reported for different sizes of the time window (TW) used

Author	Methods	ACC_G	ACC_F	ACC_G^{TOL}	ACC_F^{TOL}
Xu et al. [198]	Multiclass NN	99.19%	16.01%	99.40%	43.34%
Xu et al. [198]	CRF	99.57%	28.51%	99.59%	61.30%
Xu et al. [198]	RNN	99.73%	41.05%	**99.93%**	64.86%
Our Approach	LSTM	**99.83%**	**93.17%**	99.89%	**98.31%**

Table 5.11: Comparison of our best model (LSTM - 48h) on the Baidu data-set with previously proposed models on the hard drive health status assessment task

Author	Methods	*FDR*	*FAR*
Xu et al.[198]	Multiclass NN	83.21%	0.60%
Xu et al.[198]	CRF	85.50%	0.22%
Xu et al.[198]	RNN	87.79%	**0.004%**
Li et al.[103]	CT	95.49%	0.09%
Zhu et al.[208]	BP NN	94.62%	0.48%
Shen et al.[165]	RF	97.67%	0.017%
Our Approach	LSTM	**98.20%**	0.20%

Table 5.12: Comparison of our best model (LSTM - 48h) on the Baidu data-set with previously proposed models on the hard drive failure prediction task.

Author	Methods	Accuracy	Precision	Recall
Zhang et al.[205]	LPAT+All	92.6%	89.30%	88.70%
Sun et al.[174]	TCNN	—	75.00%	67.00%
Basak et al.[20]	LSTM	—	84.35%	72.00%
Our Approach	LSTM	**98.45%**	**98.33%**	**98.34%**

Table 5.13: Comparison of our best model (LSTM - $TW = 14$ days and $q = 45$ days) on the Backblaze data-set with previously proposed models on the hard drive health status assessment task.

in the sequence extraction step. We explored time window sizes from 4 to 48 hours for the Baidu data-set, and from 5 to 15 days for the Backblaze data-set. For the latter, we considered a *prediction window (q)* varying from 15 to 45 days. As expected given the ability of LSTMs to learn long-distance

Author	Methods	*FDR*	*FAR*
Shen et al.[165]	RF	94.89%	0.44%
Xiao et al.[195]	ORF	98.08%	0.66%
Our Approach	LSTM	**98.20%**	**0.20%**

Table 5.14: Comparison of our best model (LSTM - $TW = 14$ days and $q = 45$ days) on the Backblaze data-set with previously proposed models on the hard drive failure prediction task.

dependencies, the best results are obtained with time windows of 48 hours and 15 days for the Baidu and Backblaze data-sets, respectively.

Tables 5.4 and 7.2a report results for the sequence independent models. More in details, such models take hourly samples as input rather than sequences. The best results in terms of accuracy on failed sequences are obtained with RF for the Baidu data-set, and MNN for the Backblaze data-set. Results show that a sequence dependent approach provides higher performance than a sequence independent methodology, since the former is able to capture the SMART attribute temporal dependencies. For completeness, Table 5.4 and 7.4 report the performance of obtained best models detailed by each class.

In order to evaluate the effect of the automatic health degree definition step (as detailed in Section 7.2.1), Table 5.4 compares the performance of the model using automatically *or* manually selected health levels. For the manual set-up, hard drive health levels were split only considering the features *Hour to failure* and *Day to failure* — respectively for the Baidu and Backblaze data-set. Specifically, we define weekly (seven-day long) intervals. The only exception being the first interval, which is defined as relative to three days before failure. This comparison clearly highlights the effectiveness of automatically detected degree levels, as the automatic approach consistently outperforms the manual split the variety of evaluation metrics considered.

Furthermore, we evaluate how the balancing method affects performance, by comparing proposed approach with respect to the methods in [25] and

[199]. In particular, Botezatu *et al.* [25] selected a representative subset of healthy disks by means of a K-means clustering algorithm, while Xu *et al.* [199] applied an over-sampling technique (SMOTE) to balance the minority classes. Table 5.4 and Table 5.4 report the results obtained by varying the balancing method, and show that the best results on both data-sets are obtained using proposed method.

Finally, a comparison between proposed methodology and some other proposals in the literature has been performed, which had also been tested on the SMART data-set. Tables 5.4, 5.4, 7.4a and 7.4 compare obtained best results on the Baidu and Backblaze data-sets with different approaches for hard drive health status assessment and hard drive failure prediction tasks. Results for the first experiment are shown in Table 5.4. The performance of proposed approach on the Baidu data-set have been compared against a method based on Recurrent Neural Networks (RNN) (Xu *et al.*[198]), a model based on a Multiclass Neural Network (Mutliclass NN), and one based on Conditional Random Fields (CRFs) for hard drive health status assessment.

In Table 5.4, are considered once more the models in Xu et al.[198], which were adapted to the hard drive failure prediction task by implementing a voting rule mapping different health levels to two separate classes. Then, the models in *Li et al.* [103] , Zhu *et al.* [208] and Shen *et al.* [165] have been considered — respectively, a Classification Tree (CT) model, a Backpropagation (BPNN) and a Random Forest (RF) model.

Lastly, Table 7.4a and Table 7.4 compare obtained best result on the Backblaze data-set with other state-of-the-art methods in the literature: Zhang *et al.* [205], a method based on adversarial training and layerwise perturbation (LPAT); Sun *et al.* [174], a temporal convolutional neural network for failure prediction; Basak *et al.* [20], an LSTM-based prediction model for RUL estimation; Shen *et al.* [165] and Xiao *et al.* [195], a prediction model based on part-voting Random Forest and Online Random Forest.

To summarize, proposed approach outperforms all these models in terms of accuracy on failed sequences, FDR, and FAR both for hard drive health status assessment and hard drive failure prediction tasks. Importantly, experimental results demonstrate that proposed approach is feasible for HDD health status assessment task due to the pre-processing phase and the definition of a specific model (LSTM) relying on temporal sequence. Crucially, by showing how proposed model outperforms exiting methods based on LSTMs and CNNs (Table 7.4a), these comparisons highlight the essential contribution of proposed approach.

6 Anomalous Sound Detection

6.1 Introduction

During the last two decades, *Anomalous Sound Detection* (ASD) is becoming a more and more challenging task for a plethora of applications. Generally speaking, ASD aims at identifying whether the sound emitted from a given object is normal or anomalous, and in different cases the early detection of such anomalies can help to prevent several critical problems [131, 28].

Just to cite several examples, modern surveillance systems integrate audio and video streams to automatically recognize in a more effective way suspicious events occurring within a given area [37], advanced predictive maintenance techniques are starting to exploit the sound generated by an equipment to understand when some downtime situations could arise, eventually, the sound analysis is increasingly leveraged to mitigate the effects of cyber-physical attacks [194] on different kinds of systems.

Traditional approaches adopt different kinds of supervised machine learning models to accomplish classification or regression tasks on labelled data [131, 17, 202]. Alternatively, unsupervised models have been used to distinguish between normal and abnormal situations with and without any a-priori knowledge [182, 43, 193].

Surely, more recently, the most diffused machine learning techniques are represented by deep learning approaches that have been successfully exploited in different and heterogeneous contexts, such as medical, surveillance, finance and predictive maintenance applications as demonstrated by several recent surveys [56, 106, 188, 82, 18, 45]. Nevertheless, the arise of recent cyber-physical attacks (i.e. Triton or Stuxnet), that deceive monitoring platforms, pose novel and challenging issues. For this reason, in this thesis, we focus on the predictive maintenance task, using sounds generated by particular industrial equipment (e.g., rotating machinery, pump or slide rail), whose analysis can unveil symptom of possible failure [70]. For this kind of context, it is very easy to collect sound data related to normal and abnormal behaviour of a given machinery, thus several types of deep neural networks can be effectively trained to predict eventual downtime situations and choose the best deep architecture for a specific problem.

In this thesis, a novel deep learning-based methodology for anomalous sound detection task having flexibility and efficiency characteristics that can be considered essential for real scenarios have been proposed.

In particular, proposed approach is flexible from two different point of views: firstly, it can be easily applied to multiple instances of the same equipment as well as different machines and, successively, it is possible to instantiate any type of autoencoder, replacing the encoder and decoder blocks by using other types of deep networks. Finally, has been analyzed how proposed methodology, considering both LSTM and CNN-based autoencoders, can be applied in real scenario by processing continuous audio streams coming from different audio sources in real-world factories on the basis of a customized sliding windows.

Summarizing, the main novelties of the proposed approach concern:

- the design of a general methodology, relying on two different instances of autoencoders (Long-Short Term Memory and Convolutional Neural Network), for unsupervised anomaly detection;

- the encoding of machine identifier using one hot-encoding strategy in order to correlate each machine with relative mel-spectogram produced by audio analysis;

- the conditioning of an autoencoder by jointly analyzing the relationships between mel-spectogram and the related machine identifier through an encoder-decoder architecture for computing an anomaly score related to the input sequence;

- the flexibility of the proposed methodology and its application in real scenario have been evaluated on audio streams recorded from multiple instances of different machine types (pumps, valves, slide rails and fans) of MIMII dataset [145], achieving low inference time and memory requirements.

6.2 Methodology

One of the main challenging task for predictive maintenance problem concerns the *Anomaly Detection*, defined as a set of techniques for identifying some anomalous warning or failure states of an examined industrial machine aiming to improve maintenance activities' scheduling [132]. Several approaches [160], mainly based on the analysis of sensors data, have been proposed in literature to deal with this task although the arise of recent cyber-physical attacks (i.e. Triton or Stuxnet), that deceive monitoring platforms, pose novel and challenging issues. For this reason, novel features have to be investigated, as well as equipment sounds, that can be used by external systems in order to address these types of attacks. In this section, proposed approach is described in terms of addressed task (Section 6.2.1) and designed methodology (Section 6.2.2), composed by an Offline Training and Online Operation Phases, that is respectively discussed in Section 6.2.2 and 6.2.2.

6.2.1 Task Definition

Proposed approach is focused on the anomaly sound detection (*ASD*) task under unsupervised settings, whose aim is to compute an anomaly score by inferring features from a sequence of K audio signals $S = \{x_1, x_2, ..., x_K\}$, recorded from different versions of industrial machine M (i.e. industrial pumps or valve) over a time horizon T. In particular, proposed approach aims to learn normal behaviors of each machine in order to compute an anomaly score τ_{x_i}, representing the reconstruction error made by the model, to the examined sequence x_i.

6.2.2 Methodology Description

The proposed *ASD* methodology is composed by two main phases: an offline phase (Figure 6.1), aiming to train an autoencoder model on the basis of extracted features from a pre-collected normal audio clips (Figure 6.2), and an online operation phase (Figure 6.3), that supports analysis and detection in real scenario.

Offline Training Phase

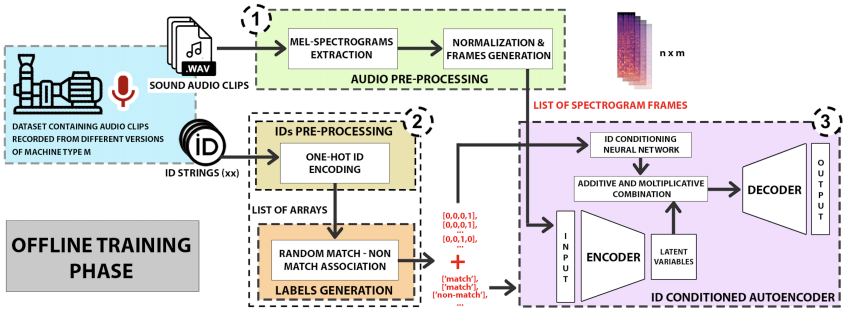


Figure 6.1: Overview of offline training phase of the proposed ASD system.

As shown in Figure 6.1, the offline training phase relies on three modules, starting from the *Audio Pre-processing* and *IDs Pre-processing*, whose aims

concern, respectively, features extraction from audio signals and encoding of *ID* string code of each machine version, until to *ID Conditioned autoencoder*, that jointly analyzes outputs of two previous modules through an encoder-decoder architecture for computing an anomaly score related to the input sequence. In particular, the *Audio Pre-processing* module is composed by two different components: *Mel-Spectrogram extractor*, that produces $n \times q$ images in log-mel-scale (*Mel-Spectrograms*) from the audio signal as input, and *Normalization and Frames generator*, which performs a segmentation in $n \times m$ overlapping frames of the generated Mel-spectrograms, that have, firstly, been normalized.

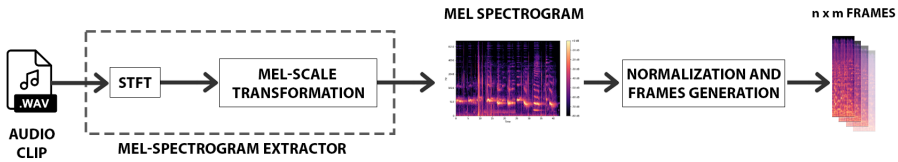


Figure 6.2: Features extraction block

The most important parameters involved into this transformation process, whose graphical abstraction is shown in Figure 6.2, are the length of the window (n_fft) used for Short-Time Fourier Transform (STFT) to produce spectrum of audio signal, the length of the overlap between two successive windows (hop_length) to produce the spectrogram and the number of bins used for the transformation into the Mel scale (n).

The *IDs Pre-processing* aims to encode this *ID* string, whose length depends by the number of different machine versions, through an *One-Hot Encoder*. For sake of simplicity, let V be the number of versions of the generic machine M – identified as $ID00$, $ID01$, $ID02$, $ID03$ – the one-hot encoder converts these strings to $[0,0,0,1]$, $[0,0,1,0]$, $[0,1,0,0]$ and $[1,0,0,0]$, respectively. In particular, this block must ensure that all frames of the same spectrogram are associated to the same binary sequence because each spectrogram is segmented in frames. The main idea concerns the use of

audio signal's IDs to enable the encoder-decoder architecture to distinguish sound signals even if there is one model trained on data belonging to different versions of the machine M. For this reason, the IDs binary sequences must be involved into the training process, as can be seen into the *ID Conditioned Autoencoder* module.

This module is composed by two components that receive in input a set of spectrogram frames, one-hot-encoded ID arrays and strings produced by *Label Generation*. The *autoencoder* relies on encoder-decoder architecture, in which the former encodes the input spectograms ($E : \mathscr{X} \to \mathscr{Z}$) that are, successively, reconstructed by the latter ($D : \mathscr{Z} \to \mathscr{X}$), where E is the domain related to the latent representation (Z) of input (X) produced by encoder and D represents the domain of reconstructed latent samples performed by decoder. The encoder and the decoder can be created with different type of layers, like convolutional, LSTM or fully-connected layers. The layer in between the encoder and the decoder is a latent or encoded representation of the input X also named Z. Differently from conventional autoencoders, in this architecture decoder input is not Z, but its mathematical combination with the output of conditioning functions: $H(Z,l) = H_\gamma(l) \cdot Z + H_\beta(l)$. In fact, the *ID Conditioning Neural Network* module, firstly performed the one-hot encoded of ID binary sequences as input (H_γ and $H_\beta : \mathscr{Y} \to \mathscr{Z}$) in order to map it into $H_\gamma(l)$ and $H_\beta(l)$, with the same size as code from $\mathscr{Z}$. The conditioning functions can be realized, for example, using dense layers and activation functions. In conclusion the output of the entire autoencoder is $D(H(E(X),l))$. The goal of ID conditioning is to inform the model about the presence of different machines of the type M, for the recognition of their different normal behaviours. The concatenation has been introduced to reduce the number of false negatives because normal sound of a machine M with ID w could be different from normal sound of a machine M with ID z this could generate some false negatives. It happens because autoencoder is unable to separate different machine versions normal behavior. Therefore,

this concatenation strategy has been chosen for a twofold reason: on the one hand, the use of one-hot vector concatenation brings a loss of information by encoding this choice in binary although it turns out to be more efficient and on the other hand, the latent representation has been used in order to improve the context representation related to each machine. In general, anomalous sound of a machine M with ID x could be similar to normal sound of machine M with ID y and this could generate some false negative (FN), because the autoencoder is unable, in this case, to distinguish this anomalous behaviours from normal one on which it has been trained on. The key concept is that the autoencoder must be trained to reconstruct normal audio spectrograms in input only if the provided ID is correct. With this assumptions, after the training, if a normal test sample is placed in input, a low reconstruction error (in terms of mean absolute error or mean squared error) is expected, while if there is an anomalous one, an high reconstruction error is generated, even if this anomalous behavior is similar to a normal behaviour of another machine. The similarity problem is so resolved by the presence of the ID.
Nevertheless, the training process needs to be revised for supporting the autoencoder in recognizing the relationships between machine identifiers (IDs) and audio signal, because the ID conditioning in latent space is not enough. For this reason, the *Label Generation* module randomly changes with a probability $1-\alpha$ the correct *ID* binary sequence associated to an audio signal with another one available. In particular, it adds the string *match* or *not-match* (corresponding to the output of the *Random Match - Non Match association* module) for each frame associated to the same audio clip on the basis of decisions. For instance, considering 100 audio signals and α equals to 0.75, we assign 75 audio signals to the correct ID whilst the remaining 25 will be associated to a random one from those available (so that the frame belonging to the same spectrogram have the same ID binary sequence and the same string).

Furthermore, a new loss must be used and tuned in the training process because the classical difference between the encoder input and the decoder output is not enough because the association between ID and audio signal may not be correct. In fact, the loss function must be calculated as $||D(H(E(X),l)) - X||$ if the label is *match*, in other words if the association between frame and corresponding ID is correct, while it assumes an arbitrary value C in the calculation of the loss $||D(H(E(X),l)) - C||$ otherwise (the label is *not-match*). In this way, the autoencoder learns the differences between machine versions, because it understand which is the correct ID string associated to a clip using the vector C with the aim to reduce the false positive (FPR) and the false negative rate (FNR).

Online Operation Phase

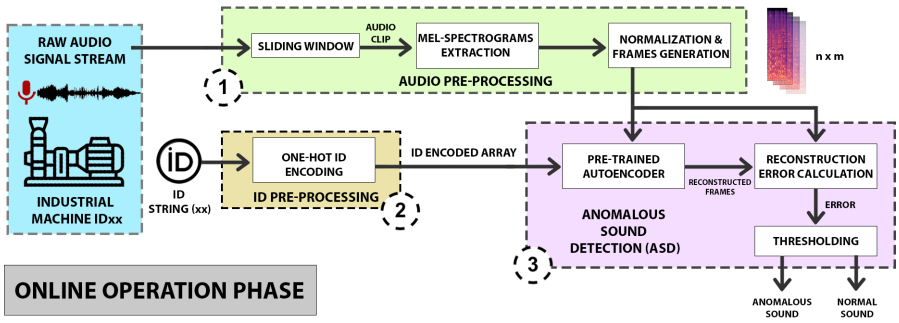


Figure 6.3: Overview of online operating phase of the proposed ASD system.

In operational phase, proposed approach should be able to work with a continuous audio stream coming from machines, which are analyzed on the basis of a sliding windows. Figure 6.3 shows the architecture for online operation phase, whose left side analyzes raw audio signal stream using sliding windows. In particular, the sliding window component takes in input the stream and samples the last T seconds from it every h seconds. The obtained audio signals is processed through mel-spectrogram extractor and the

frame generator, whose details are yet described in the offline operation phase. Finally, the *Anomalous Sound Detection* aims to identify a sound signal as normal or anomalous one. In particular, it is composed by three components: the *Pre-trained Autoencoder*, the outcome of the offline architecture, the *Reconstruction Error Calculator*, computing the difference between the input and the output of the autoencoder, and *Thresholding*, whose aim is to classify a sample in anomalous or normal ones. The threshold value could be chosen using reconstruction errors of training samples or different approaches based on ROC curves, as detailed in Section 4.6.

6.3 Experimental Evaluation

The aim of this approach is to investigate the sound analysis to deal with the more recent cyber-physical attacks, whose aim is to deceive monitoring platform affecting the performance of classical predictive maintenance tools.

In according to addressed research question, the aim is to design an experimental protocol able to evaluate how proposed strategy is capable of recognizing in an effective and efficient way possible anomalous situations by considering sound provided by different machinery.

In this section the experimental evaluation of the proposed approach is described in terms of efficacy and efficiency on the *DCASE* dataset, whose characterization is shown in Section 6.3.1. We further discussed pre-processing phase for generating images from audio signals (see Section 6.3.2) and autoencoder structure, also optimized the related hyperparameter (see Section 6.3.4). Finally, performance metrics are described in Section 6.3.5.

Two types of autoencoders are considered in order to investigate the ID conditioning effects, also analyzing its compatibility with different encoding and decoding processes. According to their autoencoder's models, the overall architectures are identified as *ID Conditioned LSTM Autoencoder* (*IDC-LSTM-AE*) and *ID Conditioned Convolutional Autoencoder* (*IDCCAE*), that

are implemented and trained on four machines available in *DCASE* dataset. Specifically, one model for each industrial machine has been trained aiming to improve the learning of specific patterns, although this choice reduces the amount of sample investigated by proposed model. In particular, the main idea is supported by the evidence that each equipment performs own operations and suffers of different anomalous conditions.

The proposed approach has been deployed on *Google Colab*[6], on which we used the following technological stack composed by *Tensorflow*[7], *Keras*[8] and *SciKit-Learn*[9].

6.3.1 Dataset and Recording Procedure

Proposed methodology has been evaluated on the *Unsupervised Detection of Anomalous Sounds for Machine Condition Monitoring* dataset, provided by DCASE 2020 TASK 2 belonging to MIMII dataset [145]. In particular, it contains audio clips recorded from four different machine types (pumps, valves, slide rails and fans), each one composed by four different versions. Table 6.1 shows information about machines' operations and possible failures that could occur. In particular, clips are recorded by a circular microphone array so that single-channel-based or multi-channel-based approaches can be evaluated.

This challenge is only based on the first channel of multi-channel recordings and the sampling rate of all signals has been down-sampled to 16 kHz. It is worth to note the presence of real factory environmental background noise mixed with the target machines sounds. Table 6.2 provides some details about machines and the number of audio clips found in training and test sets.

In conclusion, four models have been trained, one for each machine type, using training and test sets of all available IDs.

[6]https://research.google.com/colaboratory/
[7]https://www.tensorflow.org
[8]www.keras.io
[9]https://scikit-learn.org/stable/

Machine Type	Operations	Anomalous Conditions
Pump	Suction from/discharge to a water pool.	Leakage, contamination, clogging, etc.
Fan	It works to provide a continuous flow or gas of air in factories	Unbalanced, voltage change, clogging, etc.
Slide Rail	Slide repeat at different speeds	Rail damage, loose belt, no grease, etc
Valve	Open/close repeat with different timing	More than two kinds of contamination

Table 6.1: Machine descriptions with some anomalous conditions.

	ID00		ID02		ID04		ID06	
Machine Type	train	test	train	test	train	test	train	test
PUMP	906	243	905	211	602	200	936	202
FAN	911	507	916	549	933	448	915	461
SLIDER	968	456	968	367	434	278	434	189
VALVE	891	219	608	220	900	220	892	220

Table 6.2: Number of training and test samples.

6.3.2 Pre-Processing Phase

This section describe the pre-processing operations on the dataset for performing the experimental analysis, also discussing about parameters selections regarding mel-spectrograms extraction, normalization, frames generation and IDs pre-processing. In particular, the same parameters are used for all machines in the mel-spectrograms extraction task, that has been performed by using the *Librosa* library[10]: the number of bins (*n_mels*) is 128, the STFT window (n_fft) is 1024 and the *hop_length* is 512.

Due to the duration of each clip (10 seconds) and the above mentioned parameters, each mel-spectrogram has the dimension of 128×313 while the frames generation specifications are shown in Table 6.3.

[10]https://librosa.org/

	IDCCAE		IDC-LSTM-AE	
Machine	**Num. Frames**	**Hop-Size**	**Num. Frames**	**Hop-Size**
Pump	15	20	12	25
Fan	15	20	12	25
Valve	15	20	16	18
Slider	21	14	22	13

Table 6.3: Frame generation details.

In particular, the column *Num. Frames* reports the number of frames extracted from each spectrogram for each machine, while the *Hop-Size* represents the segmentation time-window shift, whose difference with respect to the the window's length determines the overlap between two successive frames according to their timestamps. Regarding the normalization, a *Z-Score* is applied on spectrograms sets extracted for each ID of each machine's type before the frame generation. For both instances of the proposed methodology, frames fed into autoencoders for training have the size of $128x32$. In conclusion, the ID strings used to identify each machine type are 00, 02, 04 and 06 (as can be seen in Table 6.2) and they are encoded respectively to $[0,0,0,1]$, $[0,0,1,0]$, $[0,1,0,0]$ and $[1,0,0,0]$.

Finally, four models for each architecture type must be trained to detect eventual anomalies. Moreover, for *match* and *not-match* transformations an $\alpha = 0.75$ is chosen, while the vector C is chosen equal to 5, after optimization.

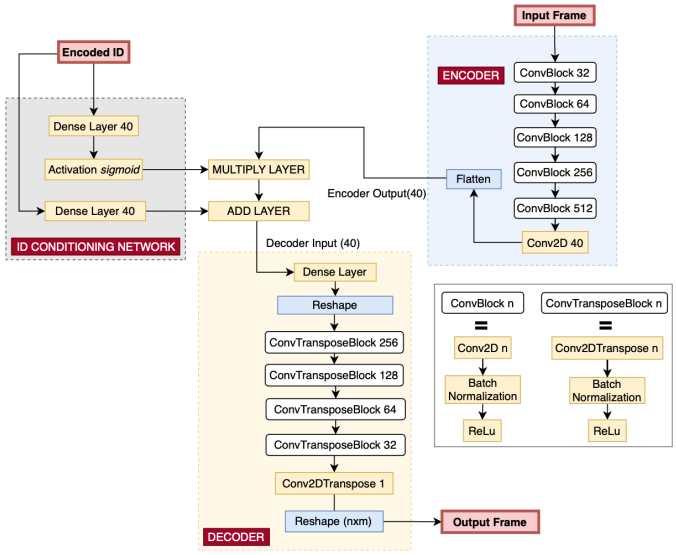
6.3.3 Autoencoder Structure

This section describes the architecture of *IDCCAE* and *IDC-LSTM-AE* (see Figure 6.4), whose conditioning phase is a sequence of mathematical operations, in which encoder output and ID conditioning network outputs are involved, as seen in Section 6.2. In particular, the *Encoded ID* is analyzed through a dense and an activation layer to produce the *ID conditioning* network first output, which is multiplied with encoder output. The second output

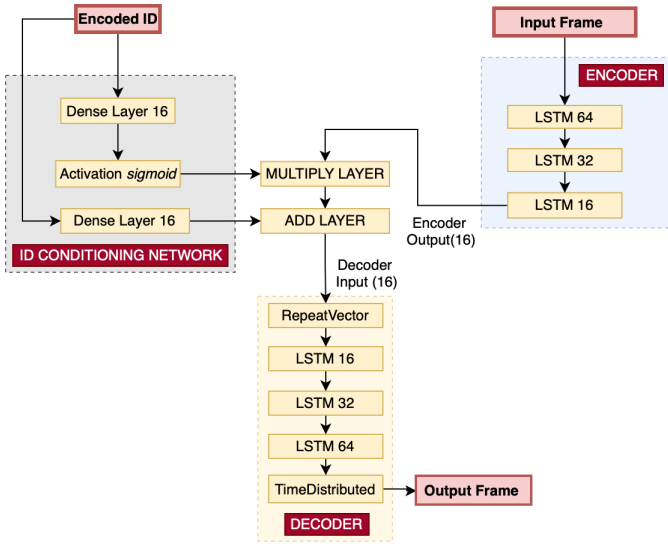
is further computed by using a dense layer with the same *Encoded ID* provided input while the final representation (decoder input) is computed by adding the multiplication output to the second output.

The Figure 6.4a reports a detailed view of encoder and decoder block in the convolutional instance of the proposed methodology. The encoder network consists in a stack of five hidden layers with convolutional filters of 32, 64, 128, 256, and 512. In particular, each component of the encoder is a block composed by a stack of different layers: a convolutional layer followed by batch normalization and the ReLU activation function. The bottleneck consists of a layer with 40 convolutional filters, reducing the encoder feature maps to a 40-dimensional encoded representation of the input. Finally, the decoder network is composed of a fully-connected layer which reshapes its input to the shape of the last layer of the encoder and five ConvTransposeBlock (Conv2DTranspose layers followed by batch normalization layers and ReLu activation functions) mirror the encoder. Conditioning operations are those explained in previously.

In turn, the Figure 6.4b describes the architecture of the LSTM based autoencoder. Encoder is composed by three LSTM layers with a decreasing number of units (64,32 and 16), which indicate the dimensionality of their output space. In this architecture, the 128x32 frames placed in input are seen as time-series of 32 timesteps each characterized by 128 features, which correspond to the frequency amplitudes (the *n_mels* bins). The decoder is the reversed version of the encoder, where at the beginning there is a RepeatVector layer which repeats its input n times. Specifically, the input is repeated 32 times like the number of timesteps. This architecture tries to capture the temporal relationship between sequential frequency amplitudes through time, in order to learn a better function to reconstruct the inputs. Finally, encoder output is a 16-dimensional representation of its input, while the conditioning operations are those explained before.



(a) ID Conditioned Convolutional Autoencoder



(b) ID Conditioned LSTM Autoencoder

Figure 6.4: Architectural details of the two designed architectures.

6.3.4 Hyperparameters Tuning

In this section the hyperparameter tuning strategy of the proposed model is described. In particular, the following model parameters have been explored: the constant vector C, that must be reconstructed by the autoencoder when the provided ID is wrong, α, being the percentage of correct frame-ID couples in training set. We performed a grid search setting α and C to {0.9, 0.75, 0.5} and {0, 2.5, 5, 10}, respectively. During the training phase other parameters are optimized for improving the performances of the proposed methodology. In addition to those seen for *IDCCAE* and *IDC-LSTM-AE*, we also optimized batch size ($\{64, 128, 256, 512\}$), number of epochs (in the [50, 200] range) and learning rate ($\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$) as autoencoder-independent hyperparameters using ADAM as optimizer. Table 6.4 reports the best hyperparameters found at the end of the optimization process, together with those seen at the beginning of this paragraph.

	IDCCAE			**IDC-LSTM-AE**		
Machine	**BS**	**EP**	**LR**	**BS**	**EP**	**LR**
Pump	256	100	0.0001	512	100	0.001
Fan	512	100	0.0001	256	100	0.001
Valve	64	100	0.0001	512	100	0.001
Slider	12	100	0.0001	512	100	0.001

Table 6.4: Batch size (BS), learning rate (LR) and number of epochs (EP) chosen. These parameters are used to get final results of the experimental part of this text.

Finally, Mean Squared Error (MSE) has been chosen to evaluate reconstruction errors for all models and all machine types.

6.3.5 Evaluation and Performance Metrics

In this section are described the metrics used to evaluate the performances of trained models. The metrics used for models evaluation are the area under

the receiver operating characteristic (ROC) curve (AUC) and the partial-AUC (pAUC). The ROC curve shows the trend of the true positive rate (TPR) in function of the false positive rate (FPR) at the variation of a parameter, the pAUC is calculated as the AUC over a low FPR range $[0, p]$, with $p = 0.1$. The proposed approaches has been evaluated in terms of the following metrics:

$$AUC = \frac{1}{N_- N_+} \sum_{i=1}^{N_-} \sum_{j=1}^{N_+} \mathscr{H}(A_\theta(x_j^+) - A_\theta(x_i^-)) \tag{6.1}$$

$$pAUC = \frac{1}{\lfloor pN_- \rfloor N_+} \sum_{i=1}^{\lfloor pN_- \rfloor} \sum_{j=1}^{N_+} \mathscr{H}(A_\theta(x_j^+) - A_\theta(x_i^-)) \tag{6.2}$$

where $A_\theta(\cdot)$ is the anomaly score generated by the autoencoder, $\cdot$ is the flooring function and $\mathscr{H}$ returns 1 when $x > 0$ and 0 otherwise. Here, $\{x_i^-\}_{i=1}^{N_-}$ and $\{x_j^+\}_{j=1}^{N_+}$ are normal and anomalous test samples, respectively, and have been sorted so that their anomaly scores are in descending order. Here, N_- and N_+ are the number of normal and anomalous test samples, respectively.

According to the above formulas, anomaly scores of normal test samples are used as thresholds. The anomaly score associated to a test sample is calculated taking the reconstruction errors average over all frames extracted from it and after the application of normalization. The pAUC is defined because it is especially important to increase the TPR under low FPR conditions, in that if an ASD system gives false alerts frequently we cannot trust it.
In conclusion, because the results produced with a GPU are generally non-deterministic, means and standard deviations are calculated from 10 independent trials. In particular, once trained a model is evaluated on test sets generating for each ID AUC and pAUC values. Moreover, mean values of AUC and pAUC are calculated from those obtained for each ID. Mean values are used to calculate mean and standard deviation over independent trials.

6.3.6 Threshold Definition

In this section an online version of the proposed methodology is shown.

The online architecture uses the autoencoder trained with offline procedures to reconstruct the spectrograms placed in input. Successively, a reconstruction error evaluation on the basis of a threshold is performed.

This section explains a possible way to calculate an optimal threshold, but it is not the only one, since different approaches about threshold definition and its optimization process can be found in literature.
To calculate an optimal threshold the concept of optimal must be firstly defined. In fact, once defined a threshold ε and a test set is evaluated, a confusion matrix could be calculated, from which true positive rate (TPR, where positive means anomalous), false positive rate (FPR) and other important measures can be extracted. The threshold goodness is related to the weights and the importance associated to these measures. In anomaly detection task is important to have an high TPR and as low as possible value of FPR.

To this purpose, the Youden's index J is defined [22]:

$$J = Sensitivity + Specificity - 1 \tag{6.3}$$

$$Sensitivity = TPR = \frac{TP}{TP+FN} \tag{6.4}$$

$$\begin{aligned} Specificity = TNR = \frac{TN}{TN+FP} &= 1 - FPR \\ &= 1 - \frac{FP}{TN+FP} \end{aligned} \tag{6.5}$$

$$J = TPR + (1 - FPR) - 1 = TPR - FPR \tag{6.6}$$

The higher J is, better the threshold is, according to this definition of optimum, and to achieve the best, the threshold that corresponds to the max value of J must be found.
Practically, the optimal threshold has been calculated using following steps:

1. Reconstruction errors calculated from test set samples (anomaly scores) are collected.

2. Using scikit-learn function *metrics.roc_curve* FPR, TPR and Thresholds, are calculated and also used to visualize ROC curve. FPR, TPR and Thresholds are three arrays of the same length.

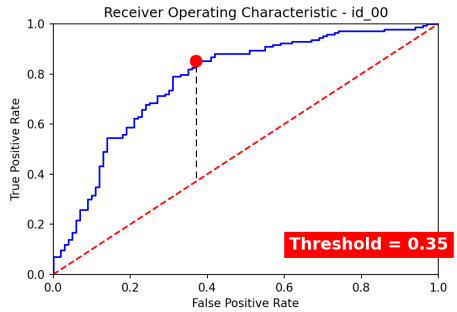
3. The optimal threshold corresponds to the element of Thresholds array at the index on which there is a maximum value of $TPR - FPR$. In other way, a vector of the differences between TPR and FPR can be calculated and then the index of the maximum difference in this vector is the index of the optimal threshold in Threshold array.

Following, Figure 6.5 shows the ROC curves calculated for *IDCCAE* architecture trained on audio clips recorded from pumps. It shows the ROC curves in blue and the bisector lines in red, while black dashed lines indicate the values of J. Moreover, red dots are used to mark the FPR and TPR which correspond to optimal thresholds, also numerically reported.
In conclusion, in online detection phases there are two alternatives:

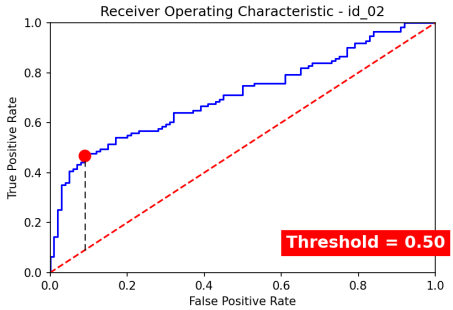
- Calculate and use a different threshold for each ID string (or machine type), even if the model used for prediction is one;

- Calculate and use only one threshold, regardless of the different machine versions (last ROC curve in Figure 6.5).

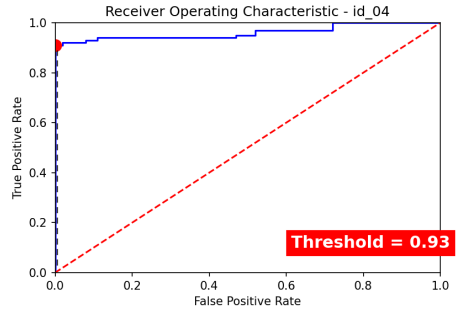
The first option implies that during detection the architecture is able to select the right threshold based on the ID associated to audio clip placed in input.



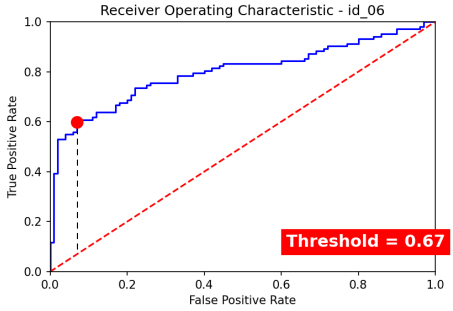
(a) ROC curve for machine with id_{00}



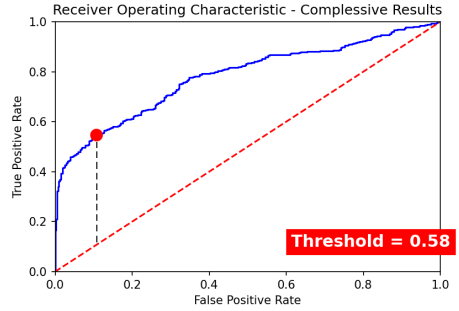
(b) ROC curve for machine with id_{02}



(c) ROC curve for machine with id_{04}



(d) ROC curve for machine with id_{06}



(e) ROC curve for all machines

Figure 6.5: ROC curves obtained using *IDCCAE* model on pumps test audio clips for the machine described in Table 3. The image shown below is the ROC curve calculated using the predictions of all four kinds of pump, with the threshold calculated.

Model	Pump		Fan		Valve		Slider		Memory
	Training	Inference	Training	Inference	Training	Inference	Training	Inference	(MiB)
IDCCAE	1h 26min 51s	3,19s	1h 58min 29s	3,04s	1 h 21 min 51s	2,94s	2h 2min 40s	2,78s	8
IDC-LSTM-AE	3min 49s	15,4s	6min 44s	14,7s	5 min 9s	15,1s	5min 56s	14,6s	64

Table 6.5: Efficiency evaluation of *IDCCAE* and *IDC-LSTM-AE* models with respect to *IDCAE* ones in terms of training and inference time and used memory.

6.4 Results

This section discusses about the results obtained by the *IDCCAE* and *IDC-LSTM-AE* models with respect to different competitors in terms of efficiency and efficacy analysis.

The efficiency of the proposed models has been evaluated by investigating their training and inference time, also considering their used memory. Specifically, the two proposed models based on LSTM and CNN respectively have been compared. The experiments have been performed using the entire dataset by averaging on ten independent trials.

Table 6.5 shows that the *IDC-LSTM-AE* achieves best results in terms of training although the inference time is highest, also requires a large amount of memory (64 MiB). On the other hand, *IDCCAE* requires a very high training time whilst achieving very good results in terms of both inference time (on average 2.98s) and used memory (8 MiB).

However, despite the high demand for training time of *IDCCAE*, the obtained results suggest its applicability in a real-world scenario due to low inference times and low memory usage, also allowing its use in edge computing.

In turn, the efficacy analysis has been evaluated comparing the proposed models with respect to different competitors in terms of *AUC* and *pAUC*. In particular, the *IDCCAE* architecture has been compared with a similar version of the architecture without the ID conditioning mechanism (*CAE* [148]).

In the same way, the *IDC-LSTM-AE* has been evaluated using the results obtained by a similar LSTM autoencoder without conditioning [80].

Both are also compared with results of the baseline model provided by DCASE authors [84]. It consists of a dense autoencoder composed by three layers in both the encoder and decoder components with 128 units, a latent space with 8 units and finally the ReLU activation function.

Finally, proposed approaches has been compared with another approach based on id_conditiong methodology applied to autoencoder [84].

Table 6.6 shows results obtained by all models for each type of machinery. For pump machinery, *IDC-LSTM-AE* achieves the best result in terms of AUC (78.29%) while *IDCAE* [84] shows an increase of 0.66% in terms of pAUC w.r.t proposed approach. In turn, for fan machinery *IDCAE* and *IDCCAE* are the best models in terms of AUC and pAUC respectively (77.45% and 70.33%). In turn, for slider and valve *IDCCAE* achieves the highest results for pAUC metric 84.14%, while *CAE* results the best model for AUC metric reporting an increase of 0.78% and 4.1% w.r.t. *IDCCAE*.

In summary, *IDCCAE* proposed model achieves the best performance on three types of machinery in terms of pAUC. In turn, *IDC-LSTM-AE* is the best model in terms of AUC only for pump machinery. Remarkable are the results obtained by *IDCCAE* for the pAUC metric because it means there are higher values of true positive and lower values of false positive, this is very important due to if an ASD system gives false alerts frequently we cannot trust it.

Finally, the results show that the proposed methodology, which leverages *ID Conditioning*, *Mel-Spectogram* and novel loss function for improving the model's performance, achieves the best value in terms of AUC and pAUC for all machines considered.

Model	**Pump**				**Fan**			
	AUC		pAUC		AUC		pAUC	
	Mean	Std.Dev	Mean	Std.Dev	Mean	Std.Dev	Mean	Std.Dev
Baseline	72.89%	0.70%	59.99%	0.77%	65.83%	0.53%	52.45%	0.21%
CAE [148]	72.07%	-	60.96%	-	66.78%	-	52.63%	-
LSTM [80]	73.94%	-	61.01%	-	67.32%	-	52.05%	-
IDCAE [84]	77.29%	-	**70.33%**	-	**77.45%**	-	70.32%	-
IDCCAE	76.63%	1.87%	67.90%	1.87%	71.05%	0.72%	**70.33%**	**0.55%**
IDC-LSTM-AE	**78.29%**	**2.21**%	69.67%	2.44%	67.66%	2.29%	65.83%	1.12%

Model	**Slider**				**Valve**			
	AUC		pAUC		AUC		pAUC	
	Mean	Std.Dev	Mean	Std.Dev	Mean	Std.Dev	Mean	Std.Dev
Baseline	84.76%	0.29%	66.53%	0.62%	66.28%	0.49%	50.98%	0.15%
CAE [148]	**91.77%**	-	76.20%	-	**78.83%**	-	53.10%	-
LSTM [80]	84.99%	-	67.47%	-	67.82%	-	51.07%	-
IDCAE [84]	80.04%	-	68.25%	-	78.26%	-	55.80%	-
IDCCAE	90.99%	4.30%	**84.14%**	**6.46%**	74.73%	5.00%	**61.18%**	**5.07%**
IDC-LSTM-AE	82.62%	1.90%	74.48%	2.64%	62.98%	2.99%	59.71%	1.53%

Table 6.6: Mean and std.dev. of AUC and pAUC for convolutional architectures on 10 independent trials. Results found in [148] are reported for comparison. Best results for each metric are marked in bold.

Part III

Interpretability and security in AI applications

7 An Explainable Artificial Intelligence methodology for HDD fault prediction

7.1 Introduction

IT infrastructures can be affected by data center's equipment failures, whose downtime costs have been growing significantly — ranging, for instance, from $5,600/minute$ in 2010 to $8,851/minute$ in 2016 [144]. As Hard Disk Drives (HDDs) have become a primary type of storage in data centers, HDD failure-rate is now one of the main factor for data center downtime, unavailability, and data loss — with obvious effects on overall business costs [157, 25]. Additionally, HDDs' reliability is affected by the complex interaction of a variety of factors (i.e., temperatures, workloads), which are difficult to address directly. Monitoring HDD's internal status is thus fundamental to reduce overhead costs due to downtime scheduling maintenance on the basis of self-monitoring, analysis, and reporting technology (SMART, [7]) for improving its availability and extending its life. While efficient planning of maintenance operations is clearly valuable, more modern approaches have been focused on *proactive analysis*: predictive strategies to identify HDDs' status in terms of binary classification (healthy or faulted).

Obviously, a plethora of approaches have been proposed with this aim, most of them using SMART attributes to predict disk replacement ahead of failure [25, 116]. However, health assessment of HDDs based on SMART statistics is not a trivial task, in particular if we are interested in estimating how much functioning time a specific HDD has left (remaining Useful Life, RUL). In particular, the historical data available about HDDs is highly imbalanced, with the majority of information available only describing healthy hard drives. Thus, any efficient solution to health status prediction will have to deal with this issue.

In this thesis, we propose a framework that predicts HDD health status, exploiting the peculiarities of LSTMS to take advantage of the variation is SMART attribute values over time. In doing so, we address one of the biggest challenges to the use of deep learning approaches in predicting HDD health status: namely, the imbalanced nature of the available data-sets. Moreover, we show how *explainable artificial intelligence* tools can we used to probe the results of proposed model, and support practitioners in their decision-making.

7.2 Methodology

Since hard drives often deteriorate gradually rather than abruptly, we argue that temporal analysis methods are more appropriate than methods that do not consider time when modeling the sequential nature of the dependencies within SMART attributes. Thus, we suggest an approach to estimate and explain the RUL of a HDD, by automatically identifying specific health conditions on the basis of SMART attributes values. This methodology is grounded in four main steps (Fig. 7.1): i) *Hard drive health degree definition*; ii) *Sequences extraction*; iii) *Health Status assessment through LSTM*; iv)*Explainer of HDD Health Status via XAI tools*.

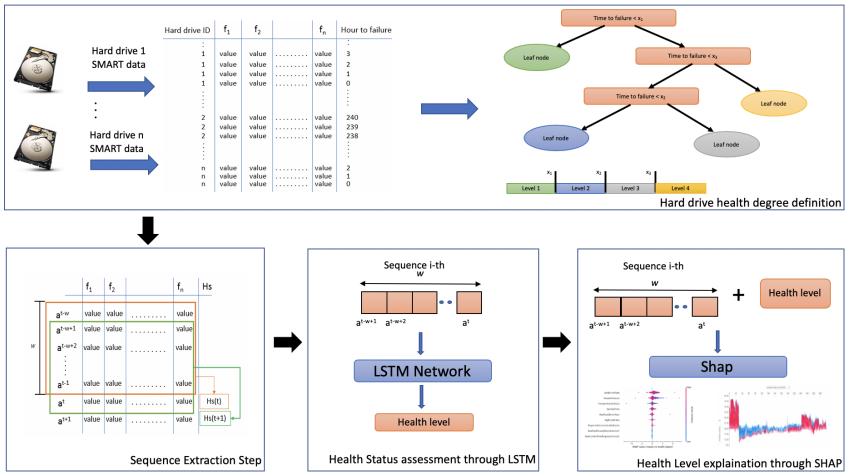


Figure 7.1: The proposed schema: Hard drive health degree definition; Sequences extraction; Health Status assessment through LSTM; XAI Explainer of HDD Health Status.

7.2.1 Health degree definition

In this step we consider only the hard drives that are going to fail, introducing for each of them an additional feature representing the *time before failure*. Denoting with m_j be the number of samples for the hard disk j, it is possible associate each sample with an index i from 0 to $m_j - 1$, representing the number of samples that follow it in the sequence describing hard disk failure. As a consequence, the sample with index $i = 0$ is the last sample before failure. *Time-to-failure* is the feature representing the time before failure for each hard drive whose meaning depends on sampling period while f_1, f_2, ..., f_n are the SMART attributes. The main idea is to build a Regression Tree (RT) for each SMART attribute f_i with $i = 1,2...n$, having the feature representing the time before failure as predictor and f_i as the numeric target value. Among all the resulting trees (one for each SMART attribute f_i), the one with the highest performance is selected, showing the attribute most temporally dependent. Since the selected Regression Tree (RT) presents splits only on the feature *Time-to-failure*, the latter is used to distinguish hard drive

health levels according to time before failure. In Figure 7.1 is reported an example of hard drive health levels identification by means of the Regression Tree algorithm. Each internal node represents a split on the feature Time-to-failure, resulting in the definition of four health degree levels. The samples belonging to hard drives that will not fail are labelled as *Good* by default.

7.2.2 Sequence extraction

We extract feature sequences over specific time windows (TW), to explore the temporal dependencies within the SMART features periodically collected for each hard drive. Let w and a^t be the time window size and the set of SMART features $(f_1, f_2 ... f_n)$ at time t, respectively. Proposed model aims to predict hard drive health status at time $t+1$ ($Hs(t+1)$) considering the sequence (a^{t-w+1}..., a^{t-1}, a^t). For each a^t, the health status $Hs(t)$ is defined, and the feature sequence for each hard drive at time t is extracted considering the $w-1$ previous samples. Each sequence results in a bi-dimensional array of size $w \times n$, where n is the number of SMART features considered. For each hard drive, sequences are extracted with a stride of one. It follows that $m_j - w + 1$ sequences are extracted for each hard drive, where m_j is the number of samples for the disk j. For each sequence (a^{t-w+1}..., a^{t-1}, a^t), the hard drive's health level is defined by the health level of the set of features a^{t+1}. The result of this step is a sequence-based data-set. More specifically, the data-set consist of bi-dimensional arrays, each associated to a health level representing the hard drive's health condition between two consecutive samples (i.e., a^t and a^{t+1}).

7.2.3 Health Status assessment through LSTMs

This step consists of a multiclass classification task, where each feature sequence is assigned to one of the classes (health levels) introduced in Section 7.2.1. Due to the sequential, gradually changing nature of the SMART

features, it is important that proposed model is able to capture dependencies across features over time. Long Short Term Memory networks (LSTMs) are extension to recurrent neural networks, explicitly designed with the purpose of learning long-term dependencies. They are widely used nowadays, as they work tremendously well on a large variety of problems. In the proposed framework, the input to each LSTM layer is a three-dimensional data structure of size $z \times w \times n$, where: i) z, is the the total number of sequences (or the batch size at each iteration); ii) w is the size of each sequence — i.e., the size of a time window in terms of time steps; iii) n is the total number of features describing each time step. The implemented network has two stacked LSTM layers with 128 units, followed by a single dense layer.

7.2.4 Explainer of HDD Health Status through SHAP

Finally, each extracted sequence is explained by means a model-agnostic XAI tool: *SHapley Additive exPlanations* (SHAP, [113]) assigning to each feature of a model, an importance value for a specific prediction. All testing sequences are classified by the LSTM model, delivering a confusion matrix. Denoting by i and j the row and column indexes — $i = 1,2,...,n$ and $j = 1,2,...,n$ with n the number of classes — respectively, each element is identified as a_{ij} and represents the number of instances of i class classified as j. Let S_{i_j} be the sequences of the test set belonging to class i but classified as j, the aim is to explain this misclassification, to understand each predicted class characteristic.

7.3 Evaluation

We test the prediction performance of the model on Backblaze SMART dataset[11], and then compare its performances against three popular methods in

[11]https://www.backblaze.com/b2/hard-drive-test-data.html

the existing literature: a Classification Tree (CT) model, a Random Forest (RF) model, and a model based on Multiclass Neural Networks (MNN). In particular, we used the Backblaze data-set that contains daily data collected from 50,984 hard disks. We focused on samples belonging to Seagate ST4000DM000, since it is the most populated model in data-set (29,878 disks in total; 29,083 *good* disks and 795 *failed* disks). Among all SMART attributes, the most influential attributes have been selected after a feature selection phase[12]. Finally, the values for every SMART attribute were scaled to the interval $[-1,1]$. Data pre-processing consisted of two main steps: *Features Selection* and *Health degree computation*. In the first step, the features *Reallocated Sectors Count* and *Current Pending, Sector Count* were removed in order to preserve their raw values — that is the features *Raw Value of Reallocated Sectors Count* and *Raw Value of Current Pending Sector Count* — since the latter seem more sensitive to the health condition of hard drives. We also excluded the feature representing disk capacity. Importantly, the attributes *failure* and *Serial Number* are necessary in order to distinguish between failed and good hard drives and to create sequences for each hard drive. However, they are not taken into account during sequence classification. For good hard drives, each sample was associated to the health degree level *Good*, while for failed hard drives, their remaining functioning time depends on the number of samples collected for said device. In the Health degree computation step, we focused on the last q samples of each failed hard drives, where q is a *prediction window* that determines the period in which hard drive health status should be assessed. Specifically, proposed approach is able to predict hard drive health status q days before failure. We explored different values for q, from 15 to 45 days. After choosing the value for q, hard drive health levels are defined according to Section 7.2.1. Since the Backblaze data-set contains daily samples for each hard drive, the feature *Time-to-failure* has been renamed *Day to failure*. We then selected the regression tree built with

[12]https://www.backblaze.com/blog/hard-drive-smart-stats/

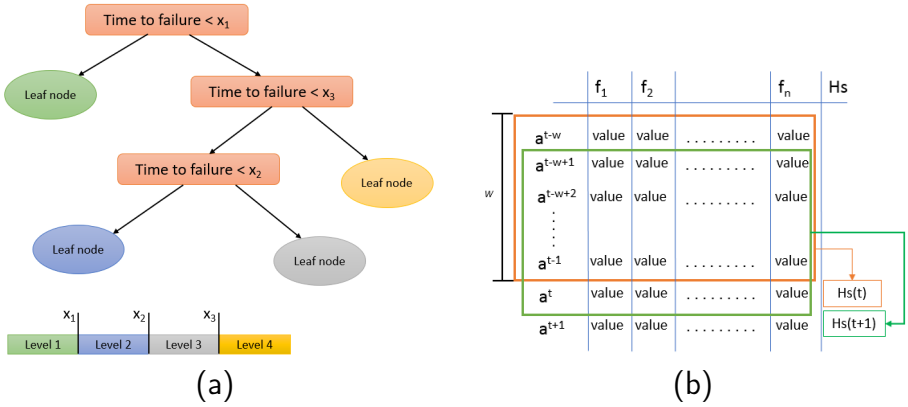


Figure 7.2: (a) HDD health levels by means of a Regression Tree. Each internal node represents a split on the feature time-to-failure, resulting in the definition of four health degree levels. (b) Sequence extraction step for a single hard drive.

the feature *Raw value of Current Pending Sector Count*. We then introduced a different level for those hard drives that will not fail. When q is set to 30 or 45, the result is the definition of 4 levels, labelled *Alert, Warning, Very Fair* and *Good*. In turn, if q is set to 15, we define 3 levels, labelled *Alert, Warning* and *Good*. The levels *Good* and *Very Fair* represent HDDs still in good health conditions. Therefore, we classify a hard drive as being in a *Good status*, if its health level is characterized as *Good* or *Very Fair* while a hard drive is classified as being in a *Failed Status*, if its health level is in *Warning* or *Alert*.

7.3.1 Experimental setup

We propose an automatic step for hard drive health levels definition, building a Regression Tree (RT) for each SMART attribute f_i, with the feature representing the hours before failure as predictor. The selected tree consider the SMART attribute *Raw Value of Current Pending Sector Count* as numerical target value. The function measuring the quality of a split is the mean squared error (*mse*). The minimum number of samples required for leaf node

in the Regression Tree is 1830, 1380, and 1200 with $q = 45$, $q = 30$ and $q = 15$ respectively. We evaluate proposed model with respect to three of the *sequence independent* methods most used in the literature: CT, a RF, and a MNN. These models are sequence independent because they generalize over input samples rather than sequences, and thus don't take the temporal dependencies of the SMART attributes into account. We implement the RT, CT and RF models using the Python scikit-learn package, and we use Keras with Tensorflow as the backend for LSTM and Multiclass NN models. As standard for this kind of techniques, the original SMART data-set was divided into training, validation and test sets. More specifically, we take the 70% of the data as training set, the 15% as validation set and the remaining data as test set. Downstream of the parameters optimization, the number of trees for RF is set to 210 and the minimum number of samples required for leaf node in CT is 20. During the training phase of the LSTM and Multiclass Neural Network models, the maximum number of epochs is set to 10, and the batch size to 500. We use Adam as an optimizer, with learning rate set to 0.001. The performance of proposed approach is first evaluated in terms of accuracy, precision, and recall. Since the distinction between good and failed hard drives is preserved in the labelling of the data-set, we express the results in term of accuracy on good sequences (ACC_G) and accuracy on failed sequences (ACC_F) — respectively, the fraction of sequences correctly classified as *Good*, and the fraction of sequence classified as the health levels suggested by the regression trees. We also measure the accuracy of classifying good and failed sequences for a tolerance of misclassification up to one health level (ACC_G^{TOL} and ACC_F^{TOL}). Finally, we evaluate performance in terms of failure prediction, by assessing *failure detection rate* (FDR) and *false alarm rate* (FAR) for each model. This is done by considering the levels *Good*, *Very Fair* as *Hard drive good statuses*; and the levels *Warning*, *Alert* as *Hard drive failed statuses*. Intuitively, FDR is the fraction of failed sequences that are

q [day]	*TW SIZE [day]*	*Accuracy*	*Precision*	*Recall*	ACC_G	ACC_F	ACC_G^{TOL}	ACC_F^{TOL}	*FDR*	*FAR*
15	5	95.88%	96.90%	95.10%	97.28%	66.56%	97.89%	98.08%	75.53%	2.82%
15	7	95.81%	97.10%	96.00%	97.02%	70.27%	97.93%	**98.45%**	79.34%	2.70%
30	5	94.54%	96.50%	94.60%	96.38%	56.07%	97.68%	88.30%	76.03%	2.73%
30	7	93.93%	96.80%	94.40%	95.59%	59.15%	97.07%	89.37%	80.70%	3.29%
30	10	95.25%	97.40%	96.10%	96.84%	61.84%	97.59%	91.35%	85.48%	2.73%
45	5	94.45%	96.70%	94.93%	95.95%	66.16%	97.80%	90.67%	78.30%	2.50%
45	7	95.82%	97.00%	95.85%	97.28%	68.34%	98.12%	89.37%	77.75%	2.17%
45	10	96.56%	97.72%	96.82%	97.71%	75.08%	98.36%	93.30%	84.18%	1.83%
45	14	**98.45%**	**98.33%**	**98.34%**	**99.21%**	**84.49%**	**99.40%**	96.65%	**91.48%**	**0.72%**

Table 7.1: Performance values for the LSTM models obtained by varying *prediction window (q)* and TW size on the Backblaze data-set.

correctly classified as failed, while FAR is the fraction of good sequences that are incorrectly classified as failed.

7.4 Results

Table 7.4 shows results of proposed LSTM based approach. Performance is reported for different sizes of the time window (TW) used in the sequence extraction step. We explored time window sizes from 5 to 15 days.

For the latter time-interval, we considered a *prediction window (q)* varying from 15 to 45 days. As expected given the ability of LSTMs to learn long-distance dependencies, the best results are obtained with a time window spanning 15 days. Table 7.2a reports results for a set of sequence independent models previously explored in the literature, taking hourly samples as input rather than sequences. The best results in terms of accuracy on failed sequences are obtained with MNN. Overall though, these results show that sequence dependent approaches provide higher performance than a sequence independent methodologies.

Finally, Table 7.4 reports the performance of obtained best models detailed by each class.

Proposed methodology has been compared to other sequence dependent approaches, which had been tested on the SMART data-set. Tables 7.4a

Model	*Accuracy*	ACC_G	ACC_F	ACC_G^{TOL}	ACC_F^{TOL}	*FDR*	*FAR*
CT	83.80%	83.87%	56.31%	95.63%	88.46%	63.58%	4.69%
RF	85.77%	85.77%	**71.75%**	93.68%	**93.82%**	**80.66%**	6.49%
MNN	**96.17%**	**99.15%**	39.78%	**99.88%**	69.20%	85.75%	**0.95%**

(a)

Metric	*Good*	*Very Fair*	*Warning*	*Alert*
Accuracy	99.21%	87.80%	78.10%	84.42%
Precision	99.90%	69.40%	64.70%	73.10%
Recall	98.80%	87.80%	78.10%	84.40%

(b)

Table 7.2: (a) Results of sequence independent models on the Backblaze data-set. (b) Results of best model on the Backblaze data-set detailed by each class.

Metric	*Good*	*Very Fair*	*Warning*	*Alert*
Accuracy	99.21%	87.80%	78.10%	84.42%
Precision	99.90%	69.40%	64.70%	73.10%
Recall	98.80%	87.80%	78.10%	84.40%

Table 7.3: Results of best model on the Backblaze data-set detailed by each class.

Author	Methods	Accuracy	Precision	Recall
Zhang et al.[205]	LPAT+All	92.6%	89.3%	88.7%
Basak et al.[20]	LSTM	—	84.35	72.0%
Our Approach	LSTM	**98.45%**	**98.33%**	**98.34%**

(a)

Author	Methods	*FDR*	*FAR*
Shen et al.[165]	RF	94.89%	0.44%
Xiao et al.[195]	ORF	98.08%	0.66%
Our Approach	LSTM	**98.20%**	**0.20%**

(b)

Table 7.4: Comparison of obtained best model (LSTM - $TW = 14$ days and $q = 45$ days) on the Backblaze data-set with previously proposed models on the hard drive health status assessment and failure prediction tasks ((a) (b) respectively).

Author	Methods	*FDR*	*FAR*
Shen et al.[165]	RF	94.89%	0.44%
Xiao et al.[195]	ORF	98.08%	0.66%
Our Approach	LSTM	**98.20%**	**0.20%**

Table 7.5: Comparison of our best model (LSTM - $TW = 14$ days and $q = 45$ days) on the Backblaze data-set with previously proposed models on the hard drive failure prediction task.

and 7.4 compare obtained best results with different approaches for hard drive health status assessment and hard drive failure prediction tasks. In particular, Table 7.4a and 7.4 compare obtained best result with some other state-of-the-art methods in the literature: Zhang *et al.* [205], a method based on adversarial training and layerwise perturbation (LPAT); Basak *et al.* [20], an LSTM-based prediction model for RUL extimation; Shen *et al.* [165] and Xiao *et al.* [195], a prediction model based on part-voting Random Forest and Online Random Forest. Proposed approach outperforms all these models in terms of accuracy on failed sequences, FDR, and FAR both for hard drive health status assessment and hard drive failure prediction tasks. Importantly, experimental results demonstrate that proposed approach is feasible for HDD health status assessment task due to the pre-processing phase and the definition of a specific model (LSTM) relying on temporal sequence. Finally, SHAP tools are used to explain each extracted sequence. To minimize the number of false negative alarms, we are interested in explaining why samples are not placed in the damaged class — that is, Alert. We focused on the damaged class because HDDs are the main cause of downtime and unavailability for a data center. Since the cost of their replacement has a significant impact on the business continuity and financial resources of a company, so an accurate analysis of these cases is not only desirable, but necessary.

Figure and 7.3a Figure 7.3b show summary plots combining feature importance with feature impact on model output. Specifically, the y-axes shows features ordered by importance, and and the x-axis shows the related

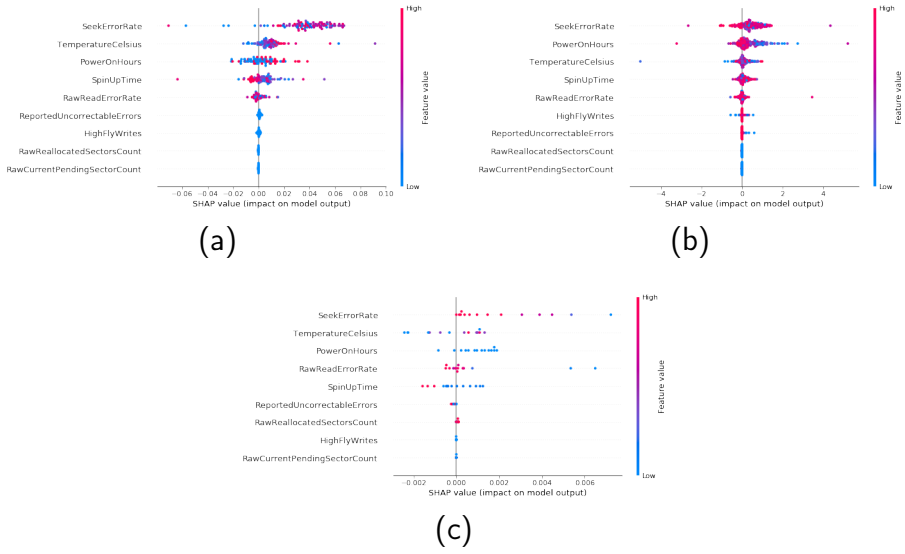


Figure 7.3: Summary plot of Good sequences classified as Good.

SHAP values. Moreover, each point is characterized by a color representing the feature values from low to high. *Temperature Celsius* (TC), *Seek Error Rate*(SER), *Power on Hours* (PoH) and *Spin Up Time* (SUT) are the most important features. As can be seen from the plots above, the most important features for the sequences classified as part of the Good class are: i) TC, which is almost always low; ii) PoH, with high value; iii) SER, which often assumes low values; iv) SUT, with low values. On the other hand, for the sequences classified as belonging to the Alert class the most important features are: i) SER, which always assumes high values; ii) PoH with low values; iii) TC, which often assumes high values; iv) SUT that takes high values. Finally, Figure 7.3c reports an example of a false negative sequence with true class Alert, and classified with Good. This plot highlights the causes that led to an incorrect classification: SER and Raw Read Error Rate contain some outliers (low values) which have a greater impact on the output than the other (high values); moreover, SUT often assumes low values. Overall, these results show the advantages of employing XAI techniques in conjunction

with deep learning models. In the case of HDD failure detection, the insights gained from XAI-based analysis helps identify false negatives cases. It should be possible to use this additional information to define better maintenance plans, this allowing companies to optimize operating costs, and increase the reliability of the provided services.

8

Bridging the gap between complexity and interpretability of a data analytics-based process for benchmarking energy performance of buildings

8.1 Introduction

The building sector is recognised as one of the largest primary energy consumers worldwide. According to the International Energy Agency (IEA) among the EU member countries, buildings are responsible for about the 21% of total final energy consumption [123]. Specifically, more than the 50% of this energy amount is used by heating and cooling systems installed in residential buildings [123]. As a consequence, the building sector is currently one of the most strategic targets for decreasing overall energy demand, improving energy efficiency to achieve demanding decarbonisation objectives.

In this context, energy benchmarking systems play a key role in the evaluation of the energy performance of buildings supporting different stakeholders (public and private) in the process of energy management and planning for achieving energy saving objectives. Cities around the world began benchmarking their building stock after realizing the potential of energy benchmarking

systems and recorded an energy saving up to 8% in a reference period of 3-4 years from their implementation [12, 60].

From the technical point of view, the main goal of a benchmarking system is to evaluate, in a systematic way, the divergence between the energy performance of a building/system and a reference baseline. Four types of baselines can be considered in existing benchmarking methods: previous performance of similar buildings (i.e., external benchmarking), current/intended performance of similar buildings (i.e., external benchmarking), previous performance of the same building (i.e., internal benchmarking), and intended performance of the same building (i.e., internal benchmarking) [107]. The first two types of baselines are used by regulators, public authorities, or private building portfolio managers to encourage owners to improve energy efficiencies of their buildings [42]. On the other hand, internal benchmarking techniques are exploited at single building level for energy performance tracking and continuous commissioning purpose.

According to the modeling approach considered, benchmarking systems can be further classified in calculation-based and data-driven ones [189]. The calculation-based benchmarking system compares the observed energy consumption with a simulated benchmark, representing an archetype or a theoretical energy performance [101]. Simulation tools, belonging to the so-called white box methods, are by now the main instrument to assess the energy performance of buildings and to evaluate the possible scenarios for energy retrofit [53, 76, 121, 100, 177]; they also provide the most reliable results at the design stage of a building [4]. This approach was however of limited use for large building stocks because it is time-consuming, labour intensive [58], and it requires detailed building information which is not always easily available at large scale [206]. On the other hand, the data-driven benchmarking process compares the observed energy consumption with a benchmark value obtained from actual energy consumption data. The most common data-driven benchmarking processes, proposed in the literature,

are performed through statistical models [102], data analytics techniques [141, 30, 61] and simple normalization of the energy consumption with respect to floor area and/or volume as a way to compute the mean or median value [189].

With the rapid growth of stored and open data in building sector and the necessity to extract knowledge from these large data sets to improve the building performance, data-driven benchmarking systems are more and more emerging [134, 200, 152]. The choice of the most suitable strategy (simple normalization, statistical models or data analytics techniques) to develop a benchmarking process mainly depends on the quantity and the quality of the available information and on properties of the considered dataset.

In the last decades, instruments, such as, Energy Performance Certificates (EPCs) have emerged as a key tool for driving the definition of energy efficiency policies for the building sector. As a reference, under the Energy Performance Buildings Directive (EPBD) (2002/91/EC), EPCs have become compulsory in EU Member States. The EPBD allows member states to define the actual implementation of its directives. In Italy the EPBD is currently implemented by various national legislative decrees and technical standards, but there are different rating schemes developed in local areas (regions and autonomous provinces). EPCs provide theoretical measure of building performance if they are operated in standard conditions. However, the performance gap, i.e. the difference between estimated and actual energy performance could be significant. For instance, [138] stated that for the Swedish EPCs data-set the performance gap is about the 20% for energy consumption assessments. An EPC is therefore not fully representative of the actual performance during operation but makes it possible to conduct comparisons between a building and its peers.

As emerged from the scientific literature, EPCs data sets represent today great sources of information and a growing number of researchers are using them for addressing different tasks in the context of building energy manage-

ment [138] including advanced benchmarking analysis [14]. The interest in energy performance assessment is increased especially to estimate how the combination of different features affects the energy needs in buildings [11]. In fact, from the design point of view, it is crucial to determine the effect of the building features on its future energy performance in the early design phase. Similarly, for existing buildings, it could be useful to evaluate the feasibility and impact of a refurbishment plan. Regardless the scope to be pursued, estimating building energy performance in a quick and reliable way, for different combinations of building features, is essential for different actors such as building owners, designers, facility managers and public authorities [14, 29]. Despite this, building professionals are typically suspicious towards the prediction results of data-driven processes because they cannot always fully interpret the model inference mechanism. In fact, what not-expert users need in practice is not only the result obtained through a single prediction, but also explanations for improving the awareness of the decision-making process. In this perspective it is becoming more and more important to develop predictive analytics tools capable of providing feedbacks about the reasons behind a certain prediction with robust indication of the supporting and conflicting evidences towards it [55, 125, 12].

Explainable artificial intelligence, also called XAI, is an emerging subject in the field of big data analytics. It aims to provide methods and tools to enhance the model usability breaking the trade-off between model complexity and model interpretability [55]. Considering the practical difficulties faced by building professionals in utilized advanced supervised learning techniques, XAI is very promising to fully exploit the potential of advanced machine learning techniques in the building application field [158, 125, 12].

According to the aforementioned motivations, the objective of this work is twofold. Firstly, the work proposes a data-driven process capable to estimate, for a large set of EPCs of flats, the membership to specific energy performance classes for benchmarking purpose. The classification task is performed

through different ML classifiers characterised by high accuracy but whose inference mechanism, despite in some cases is human-readable, can not be easily interpreted by the end-user. Secondly, a XAI-based explanation process is introduced to provide insight about the behaviour of classification models used to benchmark the energy performance of buildings and to understand the motivations behind correct and wrong classifications (this information can be very helpful for e.g., certification entities or technical figures). To this aim, the explanation process combines the XAI tool called LIME together with k-means clustering method for providing local but representative explanations of model predictions. The proposed methodology was then tested on an EPC data-set related to about 100,000 flats located in Piedmont (north-western region in Italy).

In the light of the objective of this activity, the following Section **??** reports and discuss the literature concerning the implementation of XAI-based processes in different fields of research including the one of energy and buildings. The main contributions to the literature, an the novelty introduced with this study, are presented and discussed in 8.2.

8.2 Novelty and contribution of the work

The work presented in this thesis aims to introduce a novel approach in explanation analysis leveraging local XAI tools, such as LIME, for providing insights about the behaviour of classification models used for benchmarking the energy performance of buildings.

The approach combines different advanced data analytics techniques with the aim of maintaining the output of an external building energy benchmarking process human-readable and interpretable while providing accurate and reliable results. This aspect is extremely valuable for such kind of benchmarking systems because it is usually employed by regulators, public authorities, or managers involved in the decision-making process of large building portfolio

energy management. For this reason the proposed approach was tested on an EPCs dataset related to the energy performance evaluated for about 100,000 flats located in Piedmont (north-western region in Italy).

Despite the spread of XAI techniques in several domains, to the best of authors' knowledge, the proposed approach represents the first attempt to investigate the problem of explainability in the Energy Analytics domain for the automatic estimation of building energy performances using an EPCs dataset. In particular, different ML models were firstly developed in order to solve the classification task under analysis (estimation for a new instance of its membership to an energy performance class). Successively, a XAI-based process was used to probe the rationale behind model decisions in order to understand the motivations behind right and wrong classifications.

In this context the main innovative aspects introduced by the present approach can be summarised as follows:

- The proposed framework makes it possible to employ the best classifier for energy benchmarking (in terms of achieved accuracy) regardless to its level of interpretability. From the energy point of view, the interpretability of the data-driven model used for benchmarking building energy performance is often considered a constraint in the selection of the prediction model to be used and can have repercussion on the final achievable accuracy [31, 30, 14]. Following the proposed approach the end-user has the possibility to easily probe the rationale behind model decisions following an agnostic approach and then to select the model that can better extract the main patterns from data, achieving the best accuracy.

- In this study a detailed analysis was performed for better understanding the behaviour of the model in handling classifications in border areas across adjacent energy performance classes. The analysed dataset is particularly dense and includes about 100,000 EPCs of flats. As a consequence, each energy performance class can not be considered well

separated from the adjacent ones. In this context the main targets of the explanation analysis are the instances misclassified due to border effects in order to understand the model behavior and assess the trustworthiness of its predictions in such particular cases.

- The explanation analysis has been conceived both to support the end-user during the deployment phase of the benchmark model (i.e., explanation at single prediction level) and to guide the analyst in extracting the macro-behaviors of the classification models under particular conditions (i.e., misclassification of border objects). The latter objective has been pursued by coupling the local explanation algorithm, i.e., LIME with a k-means clustering analysis, in order to firstly recognise the most significant groups of similar instances in the dataset and then explain predictions with reference to prototype objects (i.e., cluster centroids) that have been intended as representative of groups of instances. In this way, the analyst is provided with a set of reference explanations to assess some key feature combinations that could lead to more certain/uncertain predictions.

The rest of the proposed approach is organized as follows. Section 8.3 provides an overview and a brief theoretical description of the data analytics methods used for conducting the analysis. Section 8.4 presents and describes the case study considered for the analysis. Section 8.5 introduces the methodological framework behind the analysis performed. Eventually, Sections 9.3 and 8.7 present and discuss the results obtained while in Section **??** the concluding remarks and future research perspectives are reported.

8.3 Materials and Methods

In this section, the data analytics methods employed in this work are briefly described. The method descriptions are not intended to be exhaustive, but they

are aimed to underline the main model features according to the objectives of this study. In particular, the classification algorithms used for developing the energy benchmarking model based on EPCs are described. Successively, a brief introduction to k-means clustering technique is provided. Eventually, the main theoretical principles of the LIME explanation algorithm are reported.

8.3.1 Classification Algorithms

As well known, classification is related to a predictive modeling problem where a class label has to be predicted starting from labelled input data. There exists a plethora of classification algorithms that can be conveniently used depending on the dataset features. Below, the five algorithms exploited in the present study (i.e., Decision tree, Random Forest, Extremely Randomized Tree, Bagging classifier, MultiLayer Perceptron) were introduced and described.

A *Decision Tree* (DT) [147] is a supervised learning algorithm that fits well with many kinds of classification problems. Given a set of labelled data, a decision tree produces a sequence of IF-THEN rules that can be used to classify the data. It works like a flow chart, separating data points into two similar categories at a time from the "tree trunk" to "branches", to "leaves", where the categories become more finitely similar. DT requires little data preparation, and can handle both numerical and categorical data. However, it can create complex trees that do not generalize well, and can be unstable because small variations in the data might result in a completely different tree being generated.

The *Random Forest* (RF) [27] algorithm is an expansion of the DT concept. The term "forest" is referred to an ensemble of DTs, usually trained with the "bagging" method. The general idea of the bagging method is that a combination of learning models increases the overall result. Specifically, it develops a number of DTs on various sub-samples of the dataset and uses average to improve the predictive accuracy of the model and control over-

fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement.

Similar to RF classifier, in this study were also used the *Extra Trees* (ET) [63] classifiers — also known as *Extremely Randomized Trees*. The two ensembles have a lot in common. Both of them are composed by a large number of DTs, where the final decision is obtained taking into account the prediction of every tree. The main differences are the following: i) RF uses replicas, it subsamples the input data with replacement, whereas ET uses the whole original sample; ii) RF chooses local optimum splits while ET chooses it randomly.

Bagging Classifier (BC) [26] is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions to form a final prediction. Such a meta-estimator can typically be used as a way to reduce the misclassification error of a black-box estimator, by introducing randomization into its development procedure and then making an ensemble out of it. The total expected error of a classifier is made up of the sum of two components, the bias and the variance. More in details, the bias for a learning rate problem is the error rate for a particular learning algorithm and measures how well the learning method matches the problem. Since the used training set is finite and not fully representative of the population of instances, a second source of error is inevitably introduced. The variance is the expected value of this component of error, over all possible training sets and test sets. Combining multiple classifiers generally decreases the total expected error by reducing the variance component: the more classifiers that are included, the greater the reduction in variance.

Eventually, a *MultiLayer Perceptron (MLP)* was implemented. In particular, an MLP is a class of feedforward artificial neural network (ANN) and consists of at least three layers of nodes: i)input layer; ii) hidden layer and iii) output layer. Except for the input nodes, each node is a neuron that uses a

nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training.

As demonstrated by the recent literature [8, 124], the presented classification approaches are some of the most spread ones in the energy analytics field, especially for building energy performance assessment and load forecasting.

8.3.2 Clustering

Clustering consists in grouping together objects that are similar to each other and dissimilar to the objects belonging to other clusters [181]. In this study, the similarity between objects was based on a measure of the Euclidian distance (Eq. 8.1), as follows:

$$d(x,y) = \sqrt[2]{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{8.1}$$

where, x and y are two vectors of length n representing the samples.

K-means is a partitive clustering algorithm [181] that consists in grouping data objects into non-overlapping subsets (i.e., clusters) such that each data object can be included only in one sub-set. K-means is used for grouping data objects in a pre-determined number of K clusters which are represented by a prototype object called centroid (i.e., mean of the points in the n-dimensional space). The first step of K-means consists in the setting of the number K of desired clusters to which corresponds a prototype object (centroid) randomly located in the n-dimensional space [181]. Each object in the dataset is then assigned to the closest centroid, and each group of objects assigned to the same centroid represents a cluster. The centroid of each cluster is then recalculated as the average of all the objects assigned to the cluster. This process is repeated until the data objects do not change cluster anymore, and the centroids do not change position.

8.3.3 Local agnostic explaination analysis with LIME

One of main features of LIME is its modeling-agnostic nature, that makes the XAI tool applicable to any ML model. More in detail, LIME provides the local interpretability of a model: each instance is fed into the model providing both a prediction and a local sensitivity analysis with the aim of highlighting how sensitive the outcome is to each input feature.

In other words, the algorithm infers the behavior of the model by perturbing the input data and analyzing how the predictions change accordingly. In practise, the output of LIME is a list of explanations reflecting the contribution of each feature to the outcome of a given instance. As a consequence, LIME enables the local interpretability of a prediction, allowing to determine how the change of a feature will impact the model output.

The above discussed process, related to the explanation produced by LIME can be formalized as follows. Considering a local point x the relative explanation is obtained with the following generic formula (Eq. 8.2):

$$\xi(x) = \underset{g \in G}{argmin} \; L(f, g, \pi_x) + \Omega(g) \tag{8.2}$$

where G is a class of potential interpretable models, $\Omega(g)$ expresses a measure of the complexity of the interpretability of $g \in G$. The model being explained be denoted f, so $f(x)$ is the probability that x belongs to a certain class. Furthermore, $\pi_x(z)$ is defined as proximity measure between an instance z to x, so as to define locality around x.

Eventually, $L(f, g, \pi_x)$ is a measure of how unfaithful g is in approximating f in the locality defined by π_x. In order to ensure both interpretability and local fidelity, we must minimize $L(f, g, \pi_x)$ while having $\Omega(g)$ be low enough to be interpretable by the human user.

8.4 Case study

The analysed case study pertains to EPCs issued for about 100,000 flats located in Piedmont region (Italy). The EPCs include several features that impact on the building energy performance, as well as the parameters employed to determine its energy needs. Based on the analyses on EPCs previously carried out in [31] and [48], a proper selection of important and easy-to-collect variables has been carried out. The following four main types of input variables to the benchmarking model, were identified: (i) *Geometry*, (ii) *Envelope*, (iii) *Time* and (iv) *System*.

The variables in the category *Geometry* include different geometric features of the flat, which impact on its energy need and performance. Several variables belong to this category such as the average ceiling height, the heat transfer surface and the gross heated volume of the flat.

The variables in the category *Envelope* are representative of the main physical properties of the opaque and transparent envelope of the flat (e.g., the thermal transmittance values of the opaque and transparent building envelope).

Moreover, in the category *Time* are included time variables such as the construction year of the building (in which is located the flat considered).

Lastly, the variables related to the heating system belong to the category labelled as *System* (e.g., the average overall efficiency of the system for space heating). The variable *average overall efficiency of the heating system* is calculated according to the standard efficiency values for each subsystem (i.e., generation, distribution, control, emission) reported into the part 2 of [186].

Among all the variables that can be extracted from an EPC, the Primary Energy Demand for space heating PED_h has been selected as the target variable of the benchmarking analysis. PED_h (expressed in kWh/m^2y) is an energy-related variable defined for benchmarking purposes (it contributes to assign an energy class label to the flat) and consists in an estimation of the energy demand of a flat under standard use conditions. The PED_h value pertains to the energy demand referred to a standard period of a heating season

and it is normalized by the flat floor area. PED_h is part of the estimated overall Primary Energy Demand of flats (PED) which also includes the Primary Energy Demand for domestic hot water (PED_w). More specifically, the heating energy demand is assessed performing an energy balance of the flat. The modeling of the building geometry considers real shapes and self or over shading of other buildings/external obstructions. The calculation procedure considers a quasi steady-state approach based on the monthly balance of heat losses (due to transmission and ventilation) and heat gains (considering both solar and internal gains) that are evaluated in monthly average conditions. In particular the standard monthly outdoor climatic conditions (i.e., temperature and solar radiation) referred to a location on the national territory are reported in the national technical regulation UNI 10349-1. Specifically, the monthly outdoor climatic conditions, reported in the part 1 of [185], are evaluated according to the standard [79] which prescribes the use of at least 10 years of measured meteorological data for the calculation. The estimation of the transmission heat losses is performed considering actual stratigraphies and thermal properties of opaque and transparent envelopes and as well as the thermal bridging effect. In standard rating conditions, parametric values related to floor area or heated net volume, are used for defining the ventilation rates and internal heat gains. The dynamic effects and their influence on the net heating energy demand are modeled by introducing the dynamic parameters such as utilization factors and adjustments of the set-point temperature related to intermittent heating/cooling or set-back. These dynamic parameters are related to the building thermal inertia, the ratio between heat gains and heat losses and the occupancy/system operation schedules. From the system side, the annual PED for space heating depends on different efficiencies considering the thermal losses in the various heating sub-systems (emission, control, distribution, generation). For the heating season, the average system efficiency is calculated as the ratio between the

net building energy need and the *PED* for heating. Furthermore, the *PED* also takes into account the electrical energy demand of auxiliary systems.

In order to remove the climatic effect and make flats comparable, PED_h is recalculated according to a reference standard climatic condition. More specifically, all the EPCs issued in Piedmont region provide an estimation of the PED_h for the standard climatic conditions of the actual city (in which the building is located), and for the city of Turin (i.e., Province capital). As a consequence, the PED_h values considered in this study assume all flats as located in the same city considering then the same standard monthly outdoor climatic conditions (i.e., temperature and solar radiation). In this way, comparisons among flats are consistent. Nevertheless, if the performance rating of a flat is required to be performed for a city different from Turin, a data scaling process based on the use of standard Degree Days (DD) represents a robust approach. In particular, the scaling of the estimated PED_h can be obtained by multiplying it for the ratio between the standard DD value referred to the actual location of the flat and the ones of Turin.

8.5 Methodology

In this section the conceived methodology is presented and described. The proposed methodology aims to develop an energy benchmarking tool based on a classification model. Successively a XAI-based process is employed to interpret the obtained results in order to better understand the model behaviour and the motivations behind correct and wrong classifications. As described in the previous sections, develop a high performance and interpretable model is not a straightforward task and it needs to take into account several aspects. The methodology unfolds over four stages as shown in Fig. 8.1.

1. *Data pre-processing stage*: all the preliminary tasks necessary to provide the proper dataset to the algorithms were implemented.

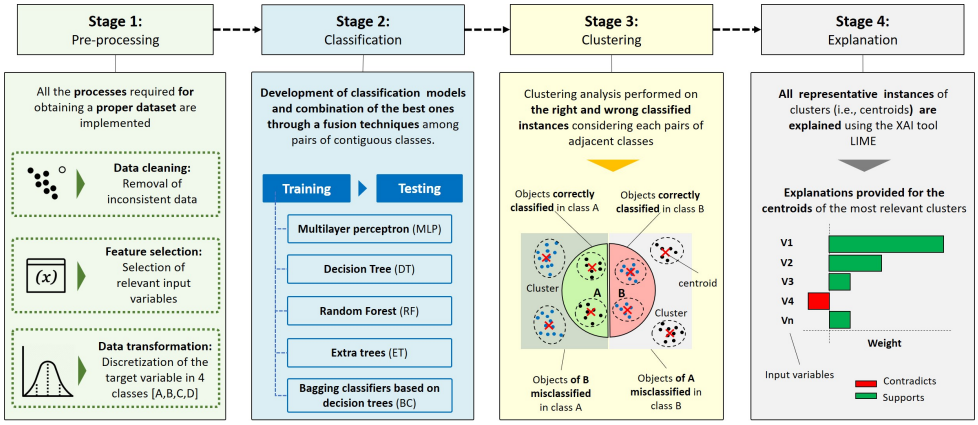


Figure 8.1: Methodological framework of the proposed study.

2. *Classification stage*: several classification algorithms were trained and tested, with the aim of evaluating how a classification model assigns the flats to different predefined energy performance classes.

3. *Clustering stage*: a clustering analysis was performed on the correct and wrong classified instances in order to identify the most relevant predictions to be explained.

4. *Explanation stage*: all the representative instances identified by means of the clustering analysis were explained and interpreted using the XAI algorithm LIME.

In the following, each stage of the methodology is described and discussed in more detail.

8.5.1 Data preprocessing

The EPC dataset includes several variables of different types (numerical, categorical, textual, etc.) related to different features affecting building energy performance as well as the variables used to quantify its energy demand. Some

of the available variables were not necessarily relevant for next data analysis, which means that their inclusion in the set of features could have increased the complexity of the benchmarking model. For this reason the dataset was inspected from energy domain experts, in order to remove the less relevant features for the analysis. The selection of the predictors has been driven from previous experiences collected on the same case study [31, 14, 48] and from the need of considering only easy-to-collect variables typically included in a EPC. All the selected features have an influence on primary energy demand from the physics point of view. It is worth to note that most of the discarded variables were poorly related with the target variable or redundant with other ones. In particular, the experiments were performed exclusively on attributes that can be categorized as geometric, thermophysical, and system-based features. The geometric and thermophysical variables are real-life variables that can be collected through surveys and inspections from energy experts before issuing an EPC. While the system-based variable (i.e., *Average global efficiency for space heating*) can be easily evaluated, with a certain degree of uncertainty, on the basis of pre-calculated values of efficiency referred to each subsystem considering the real generation system, distribution network, terminal unit, and control system installed in the building. According to this assumption, other variables that could have had a high influence on the target variable, but are difficult to be collected or involve complex calculation procedures to be determined, were excluded. The final set of variables, considering both inputs and output, is reported in Tab.8.1. For the sake of clarity, the variable aspect ratio (R) refers to the ratio of heat transfer surface area (S) to the gross heated volume (V) while the average U-values of the thermal transmittance (U_o and U_w) define the ability of the opaque and transparent envelope of the flat to transmit heat under steady-state conditions. The U-value is a measure of the quantity of heat that flows through unit area in unit time per unit difference in temperature of the environments (i.e., indoor and outdoor environment) between which the structure is located.

Category	Name	Symbol	Unit
	Input variables		
Geometry	Floor Area	A	m^2
	Heat transfer surface	S	m^2
	Average ceiling height	H	m
	Gross Heated Volume	V	m^3
	Aspect ratio	R	m^{-1}
Envelope	Average U-value of vertical opaque envelope	U_o	W/m^2K
	Average U-value of the windows	U_w	W/m^2K
System	Average global efficiency for space heating	η_h	$-$
	Target variable		
Energy	Normalized primary energy demand for space heating	PED_h	kWh/m^2y

Table 8.1: List of the input and output variables considered in the analysis.

After the feature selection, a data cleaning analysis was performed to remove statistical outliers and inconsistencies from the EPC dataset. Eventually a data transformation analysis was performed on the target variables in order to obtain a set of energy performance classes from numerical values of PED_h. Specifically, four reference classes have been considered representing respectively low energy demand flats (class A), medium energy demand flats (Class B), high energy demand flats (Class C) and very high energy demand flats (Class D). This data transformation is necessary for the construction of the classification models, which are based on a categorical response variable. The selection of threshold values between consumption classes must be accurate to obtain reliable information from the dataset. This step was performed considering PED_h distribution and selecting as threshold values, between the classes, the 25^{th}, 50^{th} and 75^{th} percentile respectively. In this way, each energy performance class roughly includes the same number of flats avoiding then class imbalance problems that could compromise the performance of the classifiers. As a result Class A includes flats with $PED_h < 91\ kWh/m^2y$,

Class B includes flats with PED_h values between 91 and 141 kWh/m^2y, while flats in Class C have 141 $kWh/m^2y \leq PED_h < 203$ kWh/m^2y, and in Class D $PED_h \geq 203$ kWh/m^2y.

8.5.2 Classification analysis

The classification analysis was aimed to develop a benchmarking model capable of predicting the membership of a new flat to one of the pre-determined energy performance classes as defined above (A, B, C, D). To this purpose, the EPC dataset was grouped by contiguous class forming three binary data sets respectively $A-B$, $B-C$ and $C-D$. Each dataset has been split 80% in the training set and the remaining 20% in the testing set. Therefore, it was carried out an exploration of ML models from the simplest to the most complex one. The following classification models were trained and tested for each dataset considered in this study: Multilayer perceptron (MLP), Decision Tree (DT), Random Forest (RF), Extra Trees (ET) and Bagging Classifier based on Decision Tree (BC).

Eventually, for each of the three data sets, the algorithm which achieved the best accuracy in testing was selected as the most valuable candidate for being used as an energy performance benchmarking model. In addition, in order to uniquely select which of the three classifiers should be used for classifying an unseen flat during the deployment phase of the benchmarking tool, a model selection technique has been implemented (Fig. 8.2).

In particular, the model selection technique exploits class contiguity by analyzing the probabilities associated with class pairs for each model. As a reference, for a new instance the first model (i.e., binary classifier $A-B$) is used. If the probability class of A is higher than B the first model is assumed to be the most suitable for performing the prediction of the new instance, otherwise the new instance is also put through the second model. In this case, if the first and second model (i.e., binary classifier $B-C$) have both the probability of class B greater than class A and class C respectively, then

Figure 8.2: Graphical representation of the model selection process.

the model with the highest probability of class B is chosen for performing the prediction on the new instance. Conversely, if the probability of class C is higher than class B for the second model, then the third model is also considered. At this stage, the above described process is the same for selecting the best model among the second and third classifier (i.e., binary classifier $C - D$).

Finally, a global SHAP analysis was performed for each of the obtained models with the aim to determine the global importance of each input variable in terms of impact that it has on the model predictions. To this purpose, according to [114] the average of absolute shapley values per feature across the data were considered.

8.5.3 Clustering analysis

The next step, after the configuration of the benchmarking models, was the clustering analysis. In particular, once the trained models were obtained, the predictions on the test sets were computed. Successively, a cluster analysis was performed in order to generate high level explanations. The result of this step, consisted in the identification of clusters that were representative of correctly predicted instances and misclassified ones, considering each pair of adjacent energy performance classes. Fig. 8.3 shows a graphical representation of the types of instances that were clustered through a K-means algorithm. As a reference for the pair of adjacent energy performance classes *A* and *B* the four clustering analysis were performed on the following types of instances in the testing set:

- objects labelled as *A* and correctly predicted as *A*;
- objects labelled as *B* and correctly predicted as *B*;
- objects labelled as *A* and wrongly predicted as *B*;
- objects labelled as *B* and wrongly predicted as *A*;

For each type of instances, all the variables labelled as input in Tab 8.1 were used by a k-means algorithm for identifying ten clusters of similar objects. Successively, only the centroids of the biggest clusters were considered in the following explanation analysis. In particular, the clusters which cover at least the 90% of the entire group of instances of the same type were selected. In this way, with a local explanation of the predictions pertaining the most relevant centroids, it is possible to extract useful knowledge about the model behaviour considering a limited number of instances in a particularly dense dataset as the considered one.

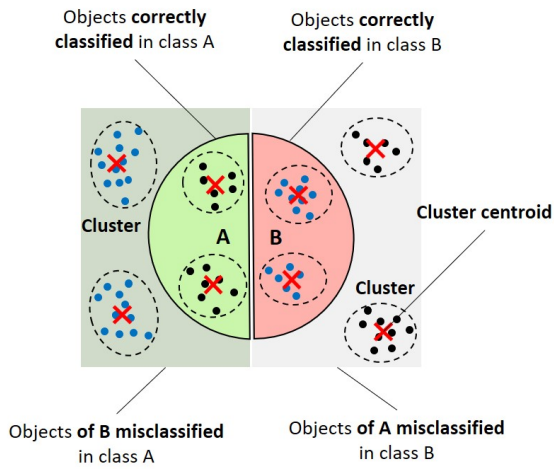


Figure 8.3: Graphical representation of the types of clustered instances.

8.5.4 Explaination analysis

Eventually, each representative instance (i.e., centroid) extracted from the previously selected clusters was explained with an XAI model-agnostic tool, that is LIME, assigning to each input variable (i.e., features of the centroid) a value of importance for a specific prediction. In particular, the main target was to explain both correct and wrong representative classifications, in order to extract, explain and interpret significant inference mechanisms learnt by the benchmarking model useful for the analyst and the end-user. In fig. 8.4 the typical output of the LIME tool, that explains the prediction of a binary classifier, is depicted. More in detail, the LIME output presents on the y-axis the input variable of the model ordered by decreasing importance, while on the x-axis the impact of each feature on the prediction for a given class. The use of color indicates the class towards which a feature has the highest impact.

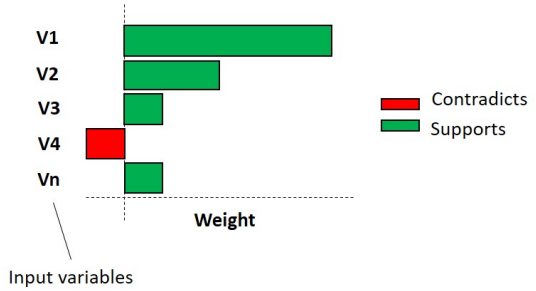


Figure 8.4: Graphical representation of the LIME output

8.6 Results

This section discusses the obtained results. The main goal is to present which are the outcomes of the proposed approach and most of all how they can be effectively used for interpreting the behaviour of the developed energy benchmarking model.

8.6.1 Classification Results

As previously explained in section 8.5, a specific pre-processing stage was considered before performing the classification task. In particular, after the data cleaning phase, the PED_h values referred to the analysed flats, were discretized and labelled with the energy performance classes A, B, C or D. The discretization was performed considering the PED_h distribution and selecting as threshold values between the classes the 25^{th}, 50^{th}, 75^{th} percentiles respectively as shown in Fig. 8.5.

Successively, the dataset was splitted 80% in the training set and 20% in the testing set and the five selected classifiers (i.e., Multilayer perceptron (MLP), Decision Tree (DT), Random Forest (RF), Extra Trees (ET) and Bagging Classifier based on Decision Tree (BC)) were developed and compared. Tab.8.2 shows the results obtained for each classifier in terms of accuracy,

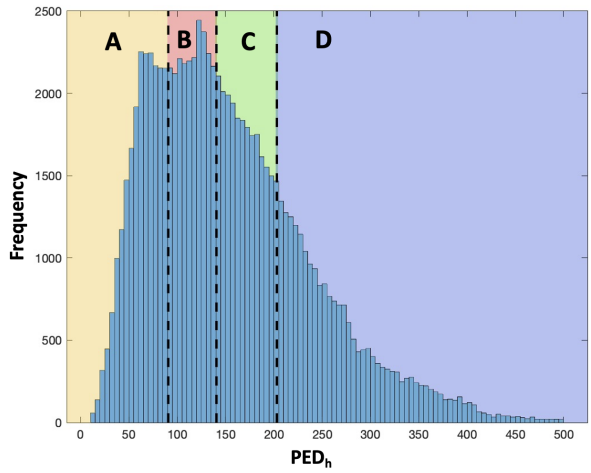


Figure 8.5: Identification of the energy performance classes on the PED_h distribution

precision and recall achieved in the testing phase. In particular according to the results obtained the RF was selected as the most suitable energy benchmarking model for energy performance classes $A-B$ while the ET for the classes $B-C$ and $C-D$. Furthermore, in Tab. 8.3 are reported the confusion matrices related to the best models selected for each class pair.

Model	**Acc.** $A-B$	**Pre.** $A-B$	**Rec.** $A-B$	**Acc.** $B-C$	**Pre.** $B-C$	**Rec.** $B-C$	**Acc.** $C-D$	**Pre.** $C-D$	**Rec.** $C-D$
DT	74.5%	74.7%	75.0%	67.3%	67.8%	66.7%	69.3%	67.9%	**74.7%**
RF	**77.8%**	**78.6%**	77.2%	69.5%	70.4%	67.8%	70.7%	70.2%	72.9%
ET	76.9%	78.5%	74.7%	**69.6%**	**70.5%**	67.7%	**71.1%**	**71.0%**	73.0%
BC	77.2%	77.9%	76.6%	68.1%	69.1%	67.2%	68.0%	69.1%	67.2%
MLP	75.7%	71.8%	**80.5%**	67.9%	67.8%	**68.9%**	70.4%	66.6%	74.2%

Table 8.2: Testing accuracy (Acc.), precision (Pre.) and recall (Rec.) achieved by each developed classifier.

Successively, a model selection technique has been employed on one hand to combine the best models, and on the other hand to identify the right model to use with a new sample according to the process described in Section 8.5.

[Random Forest]

	act. A	act. B
pred. A	3483	1035
pred. B	948	3461

[Extra Trees]

	act. B	act. C
pred. B	3081	1414
pred. C	1277	3155

[Extra Trees]

	act. C	act. D
pred. C	3299	1237
pred. D	1338	3068

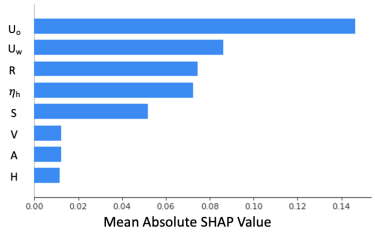
Table 8.3: Confusion Matrices related to the best models, (a) Random Forest for the class pair $A-B$, (b) Extra Trees for the class pair $B-C$ and (c) Extra Trees for the class pair $C-D$.

As a reference, the whole approach (i.e., based on the combination of three contiguous binary classifiers) was compared with a traditional multiclass classification approach. The proposed approach outperforms the multiclass approach by 5% in terms of overall accuracy.

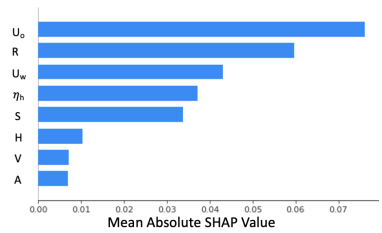
Eventually, in Fig. 8.6 the results obtained through a global SHAP analysis are reported for each of the obtained best models with the aim to highlight which are the most important features. In particular, for all of the three classification models the U_o is the most impacting variable followed by U_w or R. The efficiency of the heating system and the heat transfer surface are always ranked as the 4^{th} and 5^{th} most impacting variable respectively, while the other extensive geometric variables (i.e., V, A and H) have the lowest importance. This is consistent with the fact that the dataset includes only EPCs referred to flats that can be considered quite similar for what concerns heated gross volume, floor area and average ceiling height.

8.6.2 Clustering Results

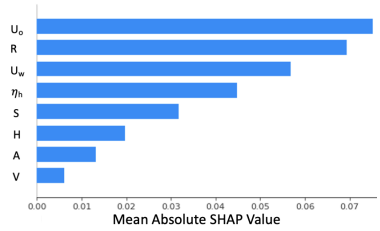
After the identification of the best classifiers to be used as energy benchmarking models for the considered set of flats, the clustering stage was performed. In particular, among each pairs of contiguous energy performance classes, the K-means algorithm was used to identify the main groups of misclassified or correctly classified flats according to the eight input variables considered in the classification stage. As previously explained, the clustering algorithm was initialised setting the number of desired clusters K = 10 for each type of



(a) Random Forest ($A-B$)



(b) Extra Trees ($B-C$)



(c) Extra Trees ($C-D$)

Figure 8.6: Global SHAP values evaluated for assessing the impact of each predictor on model output considering (a) the Random Forest for the class pair $A-B$, (b) the Extra Trees for the class pair $B-C$ and (c) the Extra Trees for the class pair $C-D$.

instances, then only the most populated clusters, that cover at least the 90% of total, were considered.

Tab. 8.4 reports the obtained results related to the contiguous energy performance classes A and B. In particular, the tags $A \rightarrow B$, $B \rightarrow A$ refer to the flats misclassified in the class B and A respectively, while tags $A \rightarrow A$ and $B \rightarrow B$ were assigned to the flats correctly classified in A and B. From the table it can be seen that for each type of instances considered, the biggest three clusters were able to include more than the 90% of the instances. It means that, according to the concept of similarity behind cluster analysis, by explaining the predictions performed for the centroids of the 12 clusters considered, the analyst can have a look at local behaviours of the classifier

Instance type	Cluster ID	n. of instances	% of total
$A \rightarrow B$	1	607	53.20%
	2	414	36.28%
	3	89	7.80%
$B \rightarrow A$	1	422	43.59%
	2	296	30.60%
	3	154	15.86%
$A \rightarrow A$	1	2494	73.85%
	2	445	13.18%
	3	431	12.76%
$B \rightarrow B$	1	2544	74.13%
	2	539	15.71%
	3	232	6.76%

1.3

Table 8.4: Cardinality of clusters identified among correct and wrong classified instances pertaining to the contiguous energy performance classes A and B

that can be used for extracting significant inference mechanisms learnt by the energy benchmarking model. In order to better characterise each cluster identified for the classes A and B, the components of each centroid were reported in Tab.8.5 with the evidence of the relative calculated average PED_h value.

Instance type	Cluster ID	A	V	H	S	R	U_o	U_w	η_h	PED_h
$A \rightarrow B$	1	80	305	3.8	213	0.7	0.5	2.5	0.80	79
	2	71	277	4.0	104	0.4	0.9	3.1	0.70	78
	3	185	688	3.8	460	0.7	0.5	2.3	0.80	73
$B \rightarrow A$	1	191	759	4.0	541	0.75	0.3	1.9	0.80	104
	2	68	259	3.8	93	0.4	0.7	2.6	0.80	108
	3	70	264	3.9	189	0.7	0.4	2.0	0.80	106
$A \rightarrow A$	1	72	277	3.9	156	0.6	0.4	1.9	0.80	59
	2	194	777	4.0	543	0.7	0.3	1.8	0.90	65
	3	67.5	265	3.9	79.4	0.3	0.8	2.8	0.80	62
$B \rightarrow B$	1	73	284	3.8	169	0.7	0.8	2.9	0.70	121
	2	62	232	4.0	164	0.3	0.85	4.9	0.70	116
	3	189	711	3.8	478	0.7	0.6	2.6	0.80	119

Table 8.5: Components of cluster centroids related to energy performance classes A and B

In addition Fig. 8.7 shows a graphical representation of the normalised centroid components referred to the most populated cluster of each instance

type. The figure is particularly useful to infer some main differences among each group of instances. For example the centroids of the cluster 1 of $A \rightarrow A$ and $B \rightarrow B$ describe similarities in terms of geometric features (A, V, H, R) and dissimilarity for thermophysical properties, such as U_o and U_w. For what concerns the centroids of the cluster 1 of $A \rightarrow B$ and $B \rightarrow A$ the combination of the input variables leads to PED_h values that are close to the border (i.e., 91 kWh/m^2y) between the two contiguous energy performance classes A and B. As explained in the section 8.5 the same clustering process was also performed on the correct and wrong predictions of the classes $B-C$ and $C-D$. The main results were included in Appendix A and Appendix B.

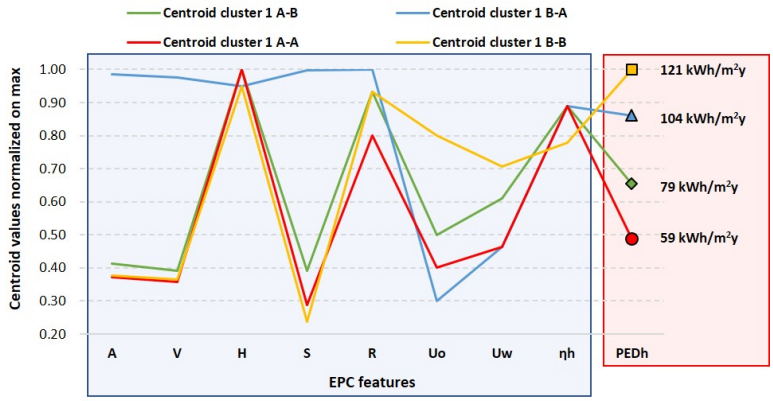
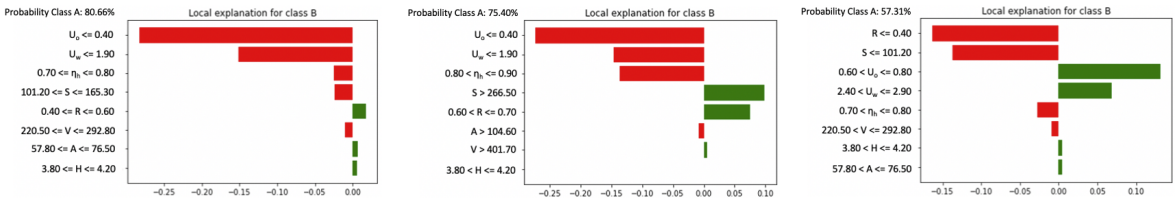


Figure 8.7: Graphical representation of the normalized centroid components related to the biggest cluster evaluated for each instance type.

8.6.3 Explanation Results

The results of the clustering analysis allowed to identify centroids that can be considered as archetypes among the analysed flats. The components of each centroid were used as input values of the classification model in order to predict for those objects the membership to a specific energy perfomance class.



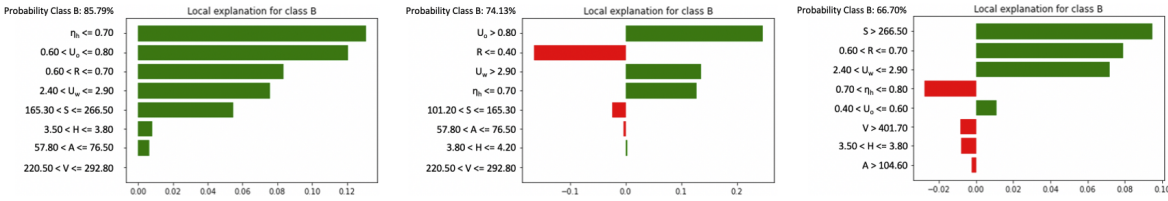
(a) Centroid of cluster 1 (b) Centroid of cluster 2 (c) Centroid of cluster 3

Figure 8.8: LIME outputs referred to the predictions of (a) centroid of Cluster 1, (b) centroid of cluster 2 and (c) centroid of cluster 3, evaluated among the instances $A \rightarrow A$

The next step employed the LIME tool for explaining why the model produced a specific prediction considering both correctly and wrongly classified flats.

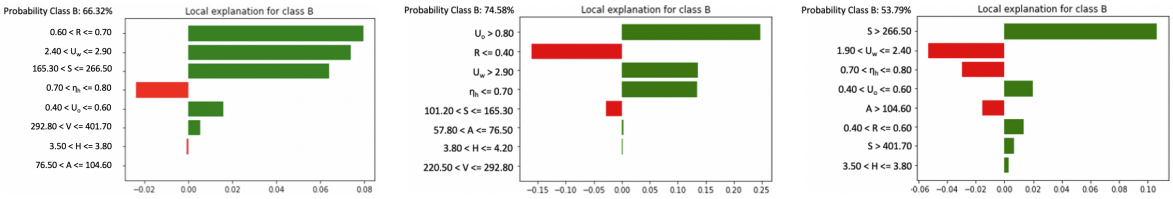
In this section, for the sake of brevity, we only reported the explanation outputs related to the clusters of the $A-B$ classes. The results obtained for classes $B-C$ and $C-D$ were included in Appendix A and B respectively. In particular, as shown in Fig. 8.8, 8.9, 8.10 and 8.11, the bars of colour red indicate the variables (and their specific numeric ranges), that support the model in predicting the class A while the green bars indicate the features that had the opposite effect dragging the prediction toward class B. The combined effect of all the input variables determined the final probability class value that was reported in the top left corner of each figure. The first three explanations, pertaining the centroids representative of the flats labelled as A and correctly classified, are shown in Fig. 8.8. The explanations of the centroid 1 and 2 were characterized by a very high class probability (over 75%). It means that when the energy benchmaking model classified the archetype flats corresponding to the two centroids, the model expressed a very high confidence in estimating their membership to the energy performance class A.

In particular the greatest impact toward class A is associated to the variables U_o and U_w that are lower than 0.4 and 1.9 W/m^2K respectively suggesting the presence of well insulated envelope. Despite this, the 3^{rd} centroid was



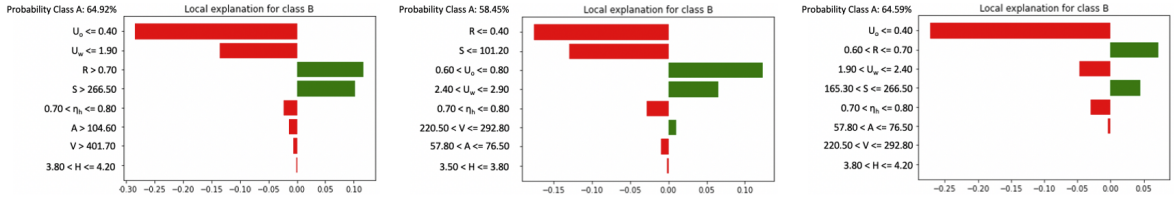
(a) Centroid of cluster 1 (b) Centroid of cluster 2 (c) Centroid of cluster 3

Figure 8.9: LIME outputs referred to the predictions of (a) centroid of Cluster 1, (b) centroid of cluster 2 and (c) centroid of cluster 3, evaluated among the instances $B \rightarrow B$



(a) Centroid of cluster 1 (b) Centroid of cluster 2 (c) Centroid of cluster 3

Figure 8.10: LIME outputs referred to the predictions of (a) centroid of Cluster 1, (b) centroid of cluster 2 and (c) centroid of cluster 3, evaluated among the instances $A \rightarrow B$



(a) Centroid of cluster 1 (b) Centroid of cluster 2 (c) Centroid of cluster 3

Figure 8.11: LIME outputs referred to the predictions of (a) centroid of Cluster 1, (b) centroid of cluster 2 and (c) centroid of cluster 3, evaluated among the instances $B \rightarrow A$

correctly classified but with a low confidence (i.e., 57%) due to higher values of U_o and U_w (that suggested a classification in class B) but lower heat transfer surface S and aspect ratio R respect to the other two centroids. It means that flats with these characteristics, can lead to potential weak predictions of the energy benchmarking model even though correctly classified. Fig. 8.9 shows the explanation results pertaining the predictions of the three centroids evaluated for the instances correctly predicted as B. The 1^{st} centroid was classified with a very high confidence of about 85%. For this case, all the variables that had an impact on the explanation, supported the classification in the right class. In particular, the most important features ($\eta_h \leq 0.70$ and $0.60 < U_o \leq 0.80$) describe a flat with both envelope and heating system less efficient than flats in the energy performance class A. The 2^{nd} centroid was classifed with a confidence of about 75%. In this case, the two most important variables disagreed in the prediction explanation. In fact, a value of U_o higher than 0.80 W/m^2K supports the class B while the aspect ratio $R \leq 0.40$ is typical of flats in class A. A low value of R often corresponds to small areas of heat transfer surfaces S then giving less importance to envelope performance variables. The last centroids of the instances $B \rightarrow B$, was explained with a probability of 67% with the most three significant variables that support the class B.

Fig. 8.10 shows the explanations referred to the misclassified instances of the type $A \rightarrow B$. In particular, the 1^{st} centroid is wrongly classified as B with a strong confidence of about 67%. For this case the geometrical and thermophisical variables have values closer to the ones of class B rather then class A. Only the η_h is consistent with the global efficiency values of buildings labelled as A. The buildings with this specific configuration of variables are particularly difficult to be classified also considering that they are characterized by an annual PED_h (of 79 kWh/m^2y) value very close to the left border between the classes A and B (91 kWh/m^2y). Similarly, the 2^{nd} centroid was also misclassified with high probability (i.e., 74%) due to a particular combination of the building features. For this case, the high

values of U_o and U_w were mitigated by a small heat transfer surface and low aspect ratio. It means that flats with these characteristics belong to class A, mostly due to geometrical aspects that positively impact on the heating energy need. Conversely, the 3^{rd} centroid was classified as B despite the values of U_w and η_h were consistent with the ones of flats in class A. However the geometry of the flat (Heat transfer surface and aspect ratio) and an high value U_o have a negative impact on the transmission heat losses. The resuls is a misclassification with a very low probability for class B. It means that the classification of such instance among class A and B is almost random for the developed classifier.

Eventually, Fig. 8.11 shows the explanations pertaining to the misclassified instances of the type $B \rightarrow A$. Also in these cases the combination of high values of aspect ratios combined with low values of thermal transmittances and vice versa, represent the main source of misclassification.

As a final remark, all the performed explanations demonstrated to be strongly consistent with the results obtained through the global SHAP analysis (Fig. 8.6) in terms of feature importance.

8.7 Discussion

The present approach focused on the analysis of EPCs evaluated for about 100,000 flats located in Piedmont (North-western region of Italy). The proposed methodology was based on the analysis of open data of EPCs and provides a robust approach for the automatic asset rating of flat energy performance. The methodology proposes a classification approach to benchmark the ideal Primary Energy Demand for space heating (PED_h) of flats according to the certification scheme used to issue their EPCs. In this section, the interpretation and the possible exploitation of the results obtained are discussed.

The classification process was based on the transformation of the numerical variable PED_h in four categorical contiguous classes identified according to the principle of equal frequency. This choice was driven by expert knowledge, in order to avoid class imbalance problems. However the ranges of PED_h that characterise each class resulted to be different between each other especially for what concern energy performance class labelled as D. The classification layer was designed to be flexible and generalizable as much as possible. In particular the classification was addressed as a three-step process. Firstly the EPC dataset was segmented in contiguous classes generating three binary datasets $A-B$, $B-C$ and $C-D$ respectively. Secondly, for each pair of classes five different classification models were trained and validated. Eventually, a model selection technique technique was implemented in order to combine the best models, which aimed to identify the right model to use with unseen data of a real-world scenario. The proposed classification process outperformed the traditional multiclass approach by 5% in terms of accuracy. From a methodological perspective, the experimental evaluation demonstrated that the approach allows to produce differentiated models, able to fit better the specific features of the related EPC segments by exploiting a limited number of input variables. In fact, thanks to the adoption of the model selection technique it was possible to use, for the pairs of contiguous energy performance classes, different algorithms to address the classification task. In addition, it is worth to note that the proposed classification approach still remains valid also considering a different discretization of the target variable that can be then set by the user (e.g., public authority) according to its specific needs. Another innovative aspect of the present work, pertains to the introduction of a generalizable approach for explaining the estimation capabilities learnt by the classification models. The explanation layer makes the results obtained from the developed energy performance benchmarking model, understandable and exploitable also for non-domain experts. Useful information can be obtained from this benchmarking tool

as it helps to discover in a straightforward way energy patterns among large dataset and at the same time understand the strengths and limitations of the estimation tool developed. To this purpose the agnostic XAI tool LIME was employed. LIME is a local explainer extremely effective in providing easy interpretations of classification results for binary problems. However, the local nature of the explanation provided should be taken into account for ensuring the feasibility of the prediction explanations, especially if large datasets are analysed. To overcome such barrier, a two-step analysis was proposed, combining LIME with an unsupervised clustering technique (i.e., K-means). The main reason behind this analysis, lied on the opportunity of extracting prototype instances in the dataset (i.e., cluster centroids) from specific groups of flats characterised by similar features (geometry, opaque and transparent envelope transmittances, system efficiencies etc.). In this way, leveraging the concept of similarity, exploited by clustering analysis, it was possible to provide local explanations of a representative combination of building features. This approach made it possible to better investigate rightly and wrongly classified instances understanding which are the main combinations of input variables that led to a specific classification results. In this perspective the explanation layer offers different opportunities from both analyst and end-user side (e.g., public authority, energy regulator, building portfolio manager). The advantages for the analyst can be summarized as follows:

- break the trade-off between model complexity and model interpretability that often constrains energy benchmarking analysis;

- refine the model and improving the feature selection. The analyst can exploit the results of a XAI process for detecting the presence of input variables with low importance or which contribution to the learnt mechanism is meaningless and removing them from the input set;

- easily understand the strengths and weakness of the developed model. During the testing and validation of the model, the analyst can use the output of the XAI layer to infer specific patterns that can be associated with high/low performances of the model;
- infer the rational behind each prediction of the model.

On the other hand the advantages for the end-user can be summarized as follows:

- understand why a certain prediction is provided and what are the supporting and conflicting model features towards it;
- according to the explanations provided for each cluster centroid, identify which are the specific combinations of building features that can compromise the trustworthiness of the model;
- easily understand which are the most important features that strongly influence the energy performance of a flat/building.

As a reference, the end-user can be aware about specific feature combinations of flats that led the model, during the validation phase, to a misclassification. There is a great added value in this kind of information given that the end-user can associate to the prediction of an unknown instance (i.e., a new flat) an explanation and a misclassification risk according to which is the closest centroid considering the features of the flat. In general, designers and authority planners can exploit such tool to understand where put their effort, among large stocks of buildings, and which could be the most convenient retrofitting strategies to be promoted considering the feature combination of the buildings of interest. In this way it is possible to support the definition of robust financial investment policies that leverage such knowledge and help to devise more targeted actions to improve energy efficiency in diversified stocks of buildings. Currently, XAI is then established as an essential requirement

towards more effective and powerful AI-based systems in many domains, and energy is included as well. From a regularity perspective, XAI is also fundamental in order to verify machine learning models and to preserve characteristics such as fairness and transparency that are more and more required to modern AI-based decision support systems.

9 Reliability of eXplainable Artificial Intelligence in Adversarial Perturbation Scenarios

9.1 Introduction

In recent years, Artificial Intelligence (AI) has reached a significant advance providing solutions in many application areas. In particular, there has been a massive increase in the use of Machine Learning (ML) algorithms to solve tasks in different fields of science, business and social workflow. This spread is related in part to the intensification of research in Deep Learning (DL), a set of artificial neural networks characterised, among other things, by a high number of layers.

This causes deep neural networks to have a larger number of parameters, making them complex to understand and more difficult to interpret, further pushing toward the idea of considering them as obscure black-box models. Since these black-box learning models are increasingly used to make important predictions in critical contexts [41][85], the demand for transparency is growing. Indeed, the risk is to create and use decisions that are not justifiable, legitimate or simply do not allow detailed explanations of their behaviour

[69]. Thus, explanations supporting the output of a model are crucial in many areas such as precision medicine, autonomous vehicles, security and finance. For the above mentioned reasons, the number of research and use of eXplainable Artificial Intelligence (XAI) is rapidly growing. The interest for XAI has also been manifested by governments, with the European Regulation on General Data Protection (GDPR) showing the important realisation of ethics [33], trust [192], prejudice [35] of IA, as well as the impact of adversarial perturbation [97] in deceiving classification decisions, especially in the case of Convolutional Neural Networks (CNNs).

The term Adversarial Perturbation (AP) refers to the whole of techniques that inject an image with a suitable, hardly perceptible, perturbation (noise) to mislead a target machine learning model. Since their introduction [65], AP algorithms have been used against a wide variety of models in several application domains [3]. Considering APs evolution and the spread of XAI in critical contexts, it was just a matter of time before researchers started working on the use of APs against XAI algorithms. On this line, a very recent work [96] proposes a black-box attack against XAI in security-critical contexts, intending to raise the attention on the problem.

This work aims to increase awareness of the risks associated with the use of XAI in critical contexts. On this line, quantitatively analyse the impact that APs have on XAI in terms of differences in the explainability maps. Since this work wants to be just an intuitive proof-of-concept, the aforementioned experiments are run in a fashion easy to understand and to quantify, by using publicly available dataset and algorithms.

9.2 Methods

To quantify the effects of APs on XAI algorithms, we defined a set of experiments intended to measure the obtained impact while producing outcomes easy to understand. To help to match the latter point, we focused on the image

classification task for its intuitive interpretation. Nonetheless, to make the reported analysis rigorous, we:

- perform the experiments on Dog vs Cat [51] and UIUC Sports Event Dataset (Event8) [105], two datasets chosen for their characteristics. More in details, the first dataset represents a very human interpretable problem, consisting in the binary classification of dogs and cats images. The second dataset describes a more complex scenario, still easily interpretable by a human observer, involving the classification of images in eight different sport events categories (e.g. rowing, sailing, etc.);

- used four different CNNs, pre-trained on ImageNet and fine-tuned on the considered datasets. Among all the available networks we selected those that, for structure and characteristics, allow us to cover a wide part of the current literature in image classification. In particular, we used AlexNet [94], ResNet18 [73], ResNet34 [73] and EfficientNet-B0 [180]. Table 9.1 reports a brief recap of their characteristics and performance on the ImageNet dataset;

- we analysed the effects of two different adversarial perturbation algorithms, chosen for their intuitiveness (the first) and diffusion (the second). The first algorithm used is the iterative Fast Gradient Sign Method (iFGSM) [97], leveraging the sign of the prediction gradient (with respect to the input's class) to craft an additive perturbation. The second is DeepFool [128], an efficient iterative method exploiting a locally linearized version of the loss to generate a series of additive perturbations;

- considered two different versions of a XAI method, Layered Grad-CAM and Guided Grad-CAM [161]. Layered Grad-CAM calculates the target output gradients with respect to the selected layer, multiplying the average gradient for each channel by the level activations. Finally,

the results are summed over all channels. On the other hand, Guided Grad-CAM is an extension of Grad-CAM which calculates the element produced by guided backpropagation with upsampled Grad-CAM attributes. In particular, the Grad-CAM attributes are calculated with respect to the layer provided in input, and the attributes are upsampled in order to match the size of the input.

Model	Top-1 Error	Top-5 Error	Parameters	Depth
AlexNet [94]	43.45	20.91	60,965,224	8
Resnet18 [73]	30.24	10.92	11,177,538	18
Resnet34 [73]	26.70	8.58	21,285,698	34
EfficientNet-B0 [180]	22.90	6.70	5,330,571	18

Table 9.1: List of selected CNNs, evaluated on the ImageNet classification challenge [155]. For each network, the top-1 and top-5 error, together with the number of parameters and layers (depth) are shown. Reported numbers refer to the corresponding PyTorch [139] implementation (excluding EfficientNet, for which we refer to [180]).

The choice of publicly available datasets and of famous and diffused architectures is intended to make the reported examples as reproducible as possible. With the same aim, we used some open-source toolboxes for both XAI and AP algorithms. In particular, the interpretation of trained models and related outputs is based on Captum [91], a library of interpretability models for PyTorch offering a series of attribution algorithms that allow understanding the importance of input characteristics, hidden neurons and layers. Similarly, for the adversarial perturbations, we used the Adversarial Robustness Toolbox (ART) [130], a Python library designed to assess machine learning security against several adversarial threats.

9.3 Results

Within the experimental setup described in the previous section, we run the adversarial perturbation algorithms against all the considered CNNs, trained on both datasets, evaluating the output of the XAI methods before and after the adversarial attacks (Figure 9.1). To this aim, defined X_c and X_p as the explainability maps obtained on the clean and perturbed images respectively, we measure the Correlation Coefficient (CC) and the Dice Similarity Coefficient (DSC) as

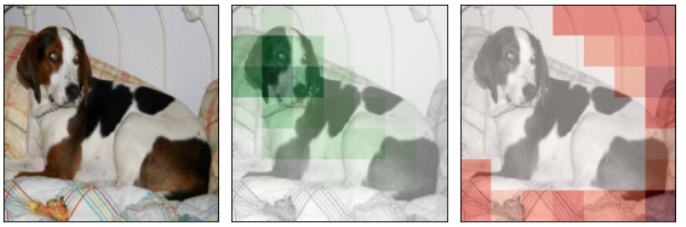
$$\mathrm{CC} = \frac{\sum \left(X_c - \overline{X_c}\right)\left(X_p - \overline{X_p}\right)}{\sqrt{\left(\sum \left(X_c - \overline{X_c}\right)^2\right)\left(\sum \left(X_p - \overline{X_p}\right)^2\right)}} \tag{9.1}$$

$$\mathrm{DSC} = 2 * \frac{n(X_c \cap X_p)}{n(X_c) + n(C_p)} \tag{9.2}$$

where $n(\cdot)$ is the number elements in each XAI map. It is worth noting that in equation 9.1 the sum is over map elements, while in equation 9.2 X_c and X_p are binary maps obtained by thresholding (using 0.9 as threshold) the corresponding probability maps.

Since the aim is to measure the impact of APs on XAI in a realistic scenario, realistic parameters for the AP algorithms have been set. In particular, for iFGSM method we set $\varepsilon = 0.1$, while for DeepFool method we set the maximum number of iterations $i_{max} = 100$ and $\varepsilon = 10^{-06}$. As a result, the adversarial attacks may not be successful for all the considered images. Therefore, the results are reported by dividing the cases in which the adversarial perturbation was successful and the cases in which it failed.

We start by reporting the boxplots for the CC and for the DSC evaluated on the explainability maps (before and after the perturbation) for all the considered CNNs, by varying the dataset, the XAI and the AP algorithms. All the plots have been drawn in couples, with green bar referring to positive XAI



Clean image (Dog: 100% - Cat: 0%)

Perturbed image (Dog: 23.41% - Cat: 76.59%)

Figure 9.1: Example of explanation maps for a clean (top row) and for a successfully perturbed (bottom row) image of a dog. In green, the explanation map for the portions of the image that have contributed to the selected output (dog in the case of the clean image, cat in the case of the perturbed image). In red, the explanation map for the portions of the image that have contributed to the non-selected output (cat in the case of the clean image, dog in the case of the perturbed image). It is worth noting that the portion of the image contributing to the predicted class is in both cases close to the dog's face.

maps (i.e. the portion of the image that contributed to the predicted output) and red bar referring to negative XAI maps (i.e. the portion of the image that contributed to the non-predicted output).

Figures 9.2 and 9.3 report the results obtained on the Dogs vs Cats dataset, under the DeepFool attack, using the Layered Grad-CAM algorithm for successfully perturbed images (the former) and failed-attacks ones (the latter). Similarly, figures 9.4 and 9.5 reports the DSC measured under the same experimental settings. Results show that, in all the cases and for all the CNNs, the XAI masks strongly vary after the perturbation, for both the positive and negative masks, as well as for successful and failed attacks. Interestingly,

from a visual perception perspective (figures 9.6 and 9.7), the portion of the image contributing to the predicted class is in both cases over the subject face (thus even when the network wrongly classifies the images into the opposite class).

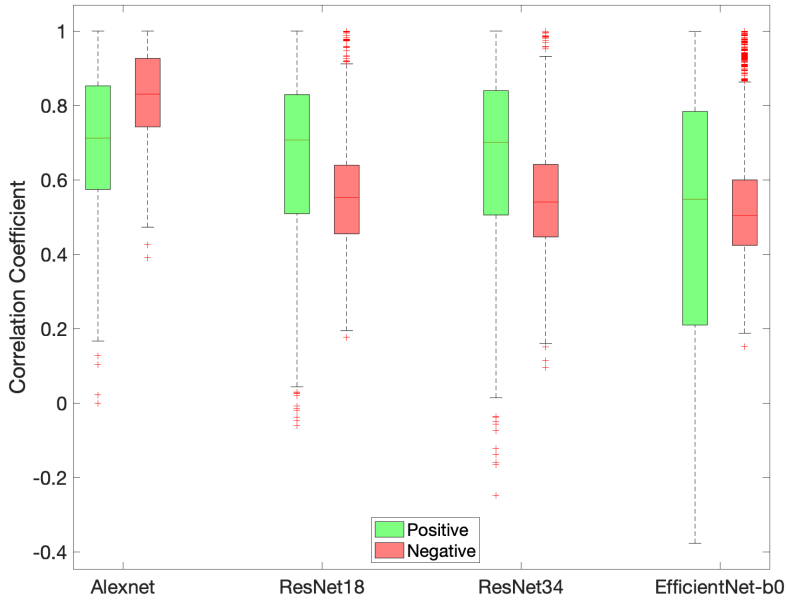


Figure 9.2: Correlation Coefficient (CC) between the Layered Grad-CAM outputs before and after the execution of DeepFool, for all the images in the Dogs vs Cats dataset for which the attack succeeded.

Figures 9.8 and 9.9 report the same set of results, obtained for the same dataset and with the same AP algorithm, but by using Guided Grad-CAM (GGC) as XAI procedure. At a first sight, the plots seem to suggest that the perturbation did not affect this scenario. However, as clearly shown in figures 9.10 and 9.11, those results are because of the huge number of pixels that remained unchanged after the perturbation (as GGC works as pixel level). Indeed, if we measure the CC only over the portions of the explanation maps that actually changed, the values drop (on average) of ~ 0.4.

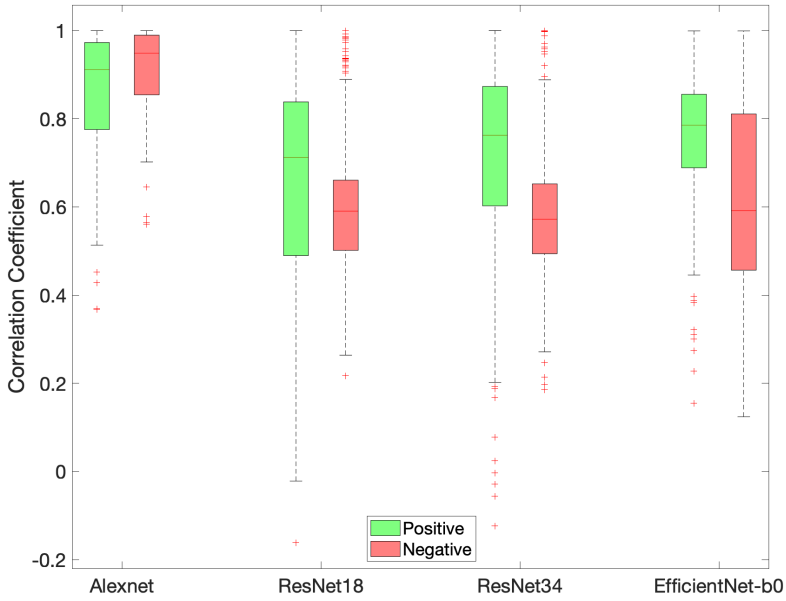


Figure 9.3: Correlation Coefficient (CC) between the Layered Grad-CAM outputs before and after the execution of DeepFool, for all the images in the Dogs vs Cats dataset for which the attack failed.

Similar results have been obtained for the FGSM perturbation algorithms and for the Event8 dataset (reported only in the supplementary material document due to paper length limits), suggesting that the number of classes and the perturbation algorithms do not change the effects that adversarial perturbations have on explainability maps.

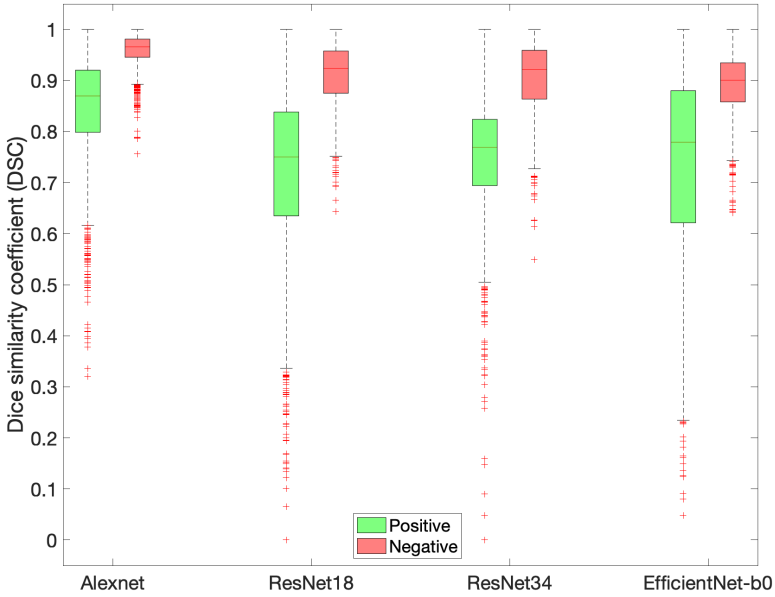


Figure 9.4: Dice Similarity Coefficient (DSC) between the Layered Grad-CAM outputs before and after the execution of DeepFool, for all the images in the Dogs vs Cats dataset for which the attack succeeded.

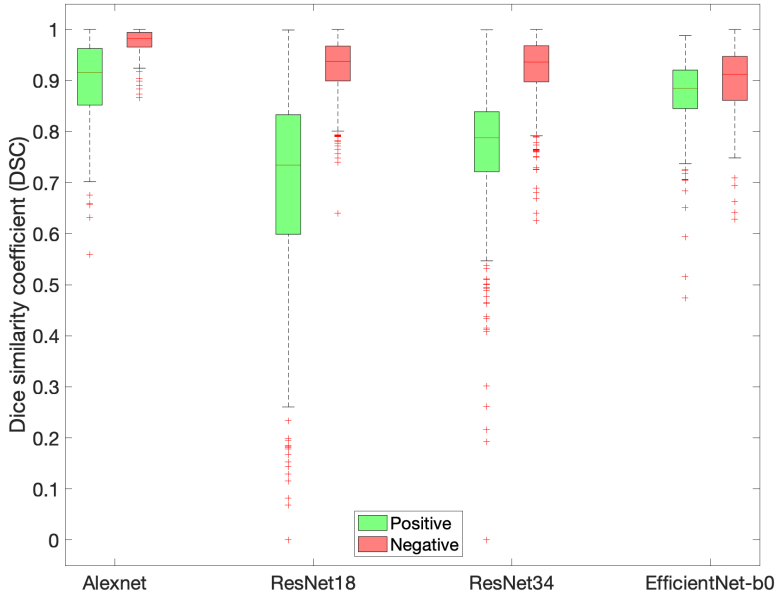
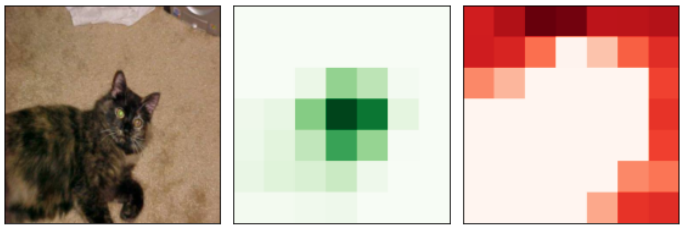
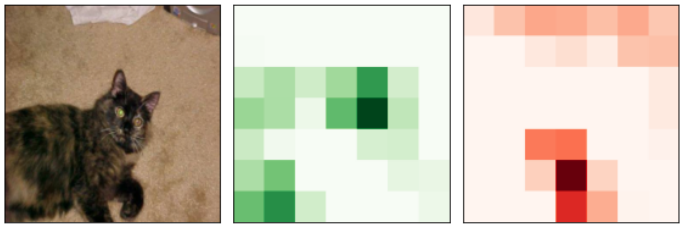


Figure 9.5: Dice Similarity Coefficient (DSC) between the Layered Grad-CAM outputs before and after the execution of DeepFool, for all the images in the Dogs vs Cats dataset for which the attack failed.

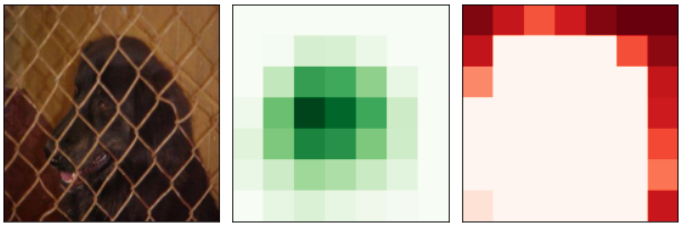


Clean image (Dog: 0% - Cat: 100%)

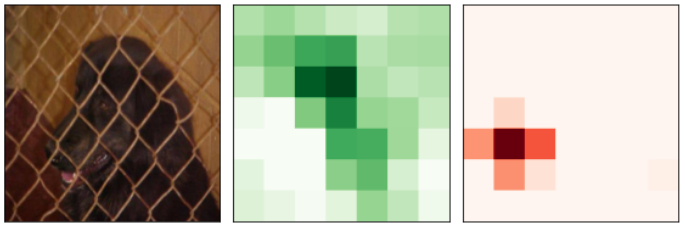


Perturbed image (Dog: 51.93% - Cat: 48.07%)

Figure 9.6: Example of explanation maps for a clean (top row) and for a successfully perturbed (bottom row) image of a cat. In green, the explanation map for the portions of the image that have contributed to the selected output (cat in the case of the clean image, dog in the case of the perturbed image). In red, the explanation map for the portions of the image that have contributed to the non-selected output (dog in the case of the clean image, cat in the case of the perturbed image). It is worth noting that the portion of the image contributing to the predicted class is in both cases over the cat's face.



Clean image (Dog: 100% - Cat: 0%)



Perturbed image (Dog: 56.09% - Cat: 43.91%)

Figure 9.7: Example of explanation maps for a clean (top row) and for an unsuccessfully perturbed (bottom row) image of a dog. In green, the explanation map for the portions of the image that have contributed to the selected output (dog in both cases). In red, the explanation map for the portions of the image that have contributed to the non-selected output (cat in both cases). It is worth noting that the portion of the image contributing to the predicted class is in both cases over the dog's face.

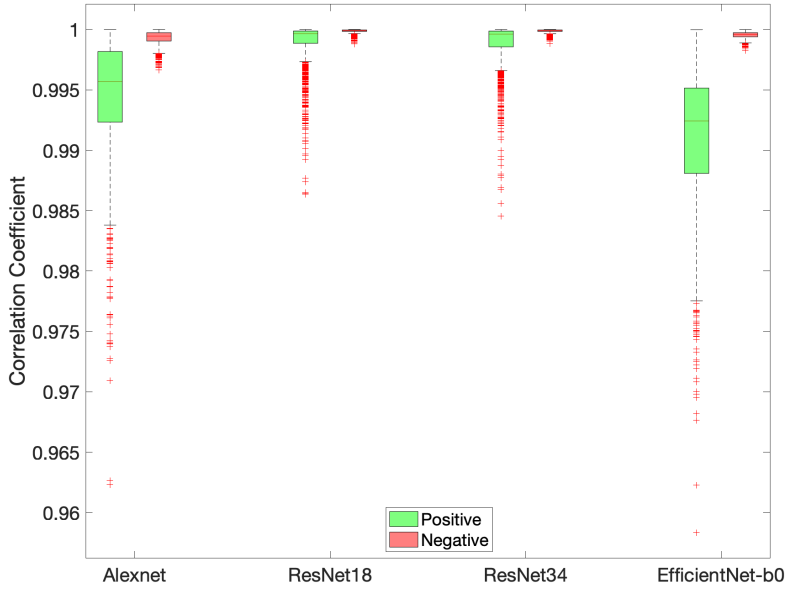


Figure 9.8: Correlation Coefficient (CC) between the Guided Grad-CAM outputs before and after the execution of DeepFool, for all the images in the Dogs vs Cats dataset for which the attack succeeded.

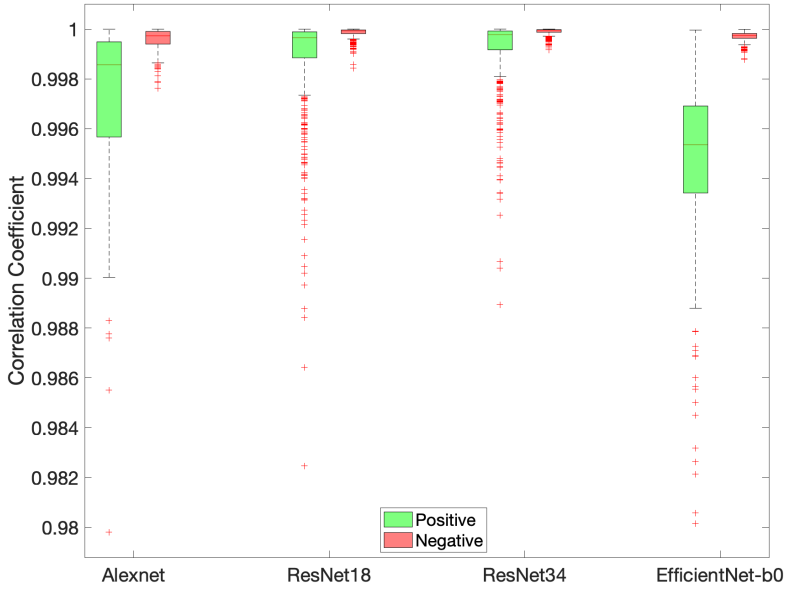
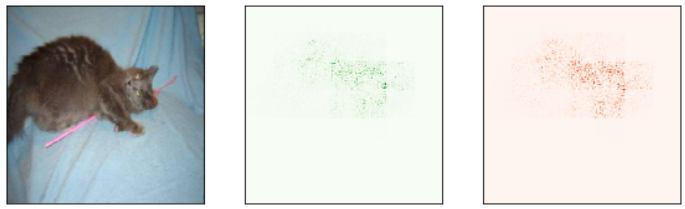
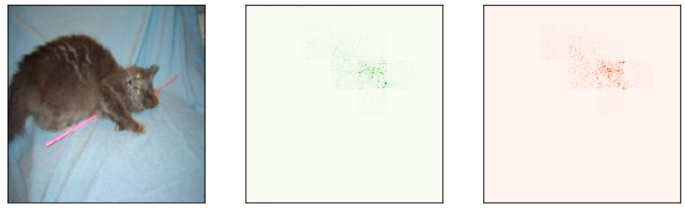


Figure 9.9: Correlation Coefficient (CC) between the Guided Grad-CAM outputs before and after the execution of DeepFool, for all the images in the Dogs vs Cats dataset for which the attack failed.

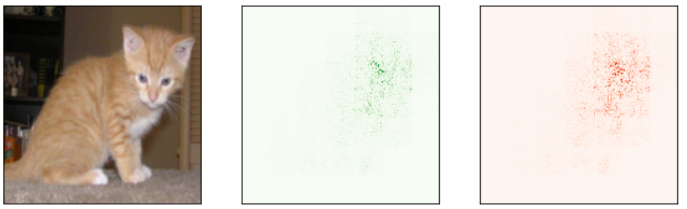


Clean image (Dog: 0% - Cat: 100%)

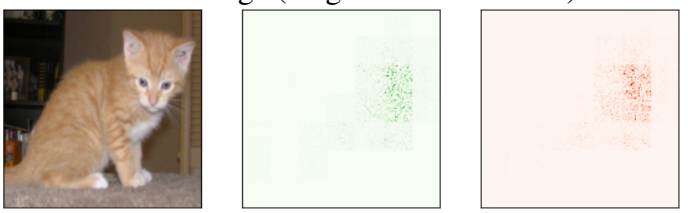


Perturbed image (Dog: 90.32% - Cat: 9.68%)

Figure 9.10: Example of explanation maps for a clean (top row) and for a successfully perturbed (bottom row) image of a cat. In green, the explanation map for the portions of the image that have contributed to the selected output (cat in the case of the clean image, dog in the case of the perturbed image). In red, the explanation map for the portions of the image that have contributed to the non-selected output (dog in the case of the clean image, cat in the case of the perturbed image). It is worth noting that the portion of the image contributing to the predicted class is in both cases close to the cat's face.



Clean image (Dog: 0% - Cat: 100%)



Perturbed image (Dog: 49.90% - Cat: 50.10%)

Figure 9.11: Example of explanation maps for a clean (top row) and for an unsuccessfully perturbed (bottom row) image of a cat. In green, the explanation map for the portions of the image that have contributed to the selected output (cat in both cases). In red, the explanation map for the portions of the image that have contributed to the non-selected output (dog in both cases). It is worth noting that the portion of the image contributing to the predicted class is in both cases close to the cat's face.

Discussions and Open Issues

This thesis proposes a methodology for predicting hard drives' health level which combines machine learning prediction techniques based on LSTMs, with an automatic approach for hard drive health status definition. In the past decades, being able to predict the health level of HDD timely and accurately has started playing a fundamental role in the administration of large data centers, as optimizing maintenance strategies has obvious impacts on overhead costs. LSTMs are interesting in the context of HDD failure prediction, as they are able to take advantage of the highly sequential nature of the information available to the model. To explore the effectiveness of this idea, we investigated how extracting sequences over time windows (TW) of different sizes affects the performance of the LSTM model. In line with the ability of LSTMs to learn long-distance dependencies, the best results were obtained with 48-hour time windows size and 14-day time windows size for Baidu and Backblaze datasets respectively. Furthermore, we showed that LSTM based models outperform sequence independent models in classifying sequences belonging to hard drives that are going to fail. As can be seen from the summary Tables 5.4, 7.4, 5.4 and 7.2a the performance gap between models on healthy samples or sequences is small. We interpret this as evidence of

the fact that it is not difficult for a classifier to identify good sequences, but it is hard to identify disks at risk of failure. In such cases, the LSTM models strongly outperforms the sequence independent models. Propose approach achieves state-of-the-art results on a hard drive health status assessment task (Tables 5.4 and 7.4a) over the SMART data-set, and competitive results in terms of FDR and FAR scores (Tables 5.4 and 7.4). Moreover, as shown in Tables 5.4 and 7.4, our approach outperforms the others in term of FDR. Although our model does not result in the lowest FAR value (Table 5.4), it keeps it to a reasonably low value. Importantly, the advantage of our solution compared to that introduced in [198] lies specifically in the implementation of a step for the automatic definition of health levels. Notably, LSTMs have been already used in the past in predictive maintenance tasks. The fundamental contribution of this thesis though, is in the way LSTMs are coupled with a technique to address the unbalanced nature of the training data, and the data-driven definition of health degree levels. Traditionally, HDD have been classified based on fixed health levels determined by domain experts. In this sense, optimal maintenance strategies would benefit from as detailed a prediction as possible, while maintaining high accuracy. However, the complex nature of HDD health status makes this technique less effective over real world data, thus requiring trade-offs between detailed and timely predictions, and prediction accuracy. By providing an automatic methodology to detect ranges on the base of each disk's behavior, our approach shows more flexibility to the varied nature of the underlying data, thus outperforming a variety of alternative models. Comparisons with alternative methods also based on LSTMs support the effectiveness of the particular approach taken in this thesis, and shows the advantages (in terms of accuracy of the prediction task) of enriching LSTMs with data-driven, flexible identification of HDD health levels and health degree settings (Table 7.4a and Table 7.4).

Additionally, our model automatically extracted degree levels were assessed over a range of prediction windows (15 to 45 days) compatible with

what already present in the literature, thus supporting the reliability of the approach in real-word scenarios. Based on such results, we argue that the health level prediction task should be preferred to other forms of evaluations when considering these kind of predictive models, as its results provide the most insightful information towards hard drive maintenance. Future work should be focused on exploring the best way data center's technicians and manager can leverage the finer-grained predictions provided by our approach to optimize long-term maintenance. Furthermore, we should conduct an extensive, detailed investigation of different health degree settings, evaluating the trade-offs of incorporating constraints from real-world applications into the automated processes discussed in this thesis so to further refine health degree definition strategies. As mentioned before, anomaly detection can be seen as a data-driven approach for predictive maintenance. Since it gives the possibility to avoid downtime, to reduce costs related to unnecessary operations and to optimize maintenance procedures, in last years a lot of anomaly detection system have been developed and studied by researches. In this thesis, we designed an anomaly sound detection framework for predictive maintenance, demonstrating that the involvement of machines identifiers (useful for conditioning) into the autoencoder learning process enables models training on sounds recorded in proximity of different versions of the same machine type (like different pumps), improving their detection capabilities. Moreover, the experimental part of the work shows that the conditioning can be done with a classical neural network, which is also compatible with the various types of encoder-decoder layers. Experiments have been conducted on *DCASE 2020 Task 2 Challenge* dataset, which is already arranged to face the task in unsupervised way, replicating a real scenario in which normal sound clips represent most of the available data for training phase. Two instances of the proposed framework have been realized, the first based on *convolutional autoencoder*, while the second uses a recurrent approach based on LSTM layers in both encoder and decoder. Experimental results have

been evaluated using AUC and pAUC, as indicated by the challenge, and they highlight improvements up to 17.61%, with respect to the corresponding non-conditioned autoencoder versions and the baseline model results, especially for pAUC. Future works will be devoted to analyze different types of conditioning networks (i.e. *Variational AutoEncoders* (VAE) or *Generative Adversarial Networks* (GANs)) and operations together with the application of several pre-processing strategies to get better training performances, like noise reduction, with the aim of reducing the audio clips background noise surely present in factory environments, or audio data augmentation techniques, as well as pitching, time-shifting and so on.

As discussed in Chapter X , there is not only a need for a methodology to achieve high accuracy but in several domains there is a need to interpret by which criteria a given prediction was obtained.

For this reason, we have argued how XAI tools can improve HDD preventive maintenance systems, by providing a transparent interpretation of the model's prediction. In the future, it will be interesting to further assess the contribution to XAI technologies to the design of accurate HDD health supervision strategies. In order to validate the use of xai in multiple application domains this thesis proposed a multi-step methodology to automatically benchmark energy performance of flats by means of classification algorithms. The analysis was complemented including in the analytical process an explanation layer based on the LIME tool in order to make the results of the analysis interpretable as much as possible. The methodology was tested on a large collection of EPCs pertaining about 100,000 flats located in Piedmont (Italy). Thanks to the performed analysis, also some limitations of this work have emerged, especially for what concerns the deployment phase of the energy performance benchmarking tool. In particular the main limitations of our work are related to the quality of EPCs data. In fact, EPCs can be subjected to different kinds of errors (i.e., clerical and calculation errors made by technicians while issuing the EPC) with potential negative impact on the

accuracy of the developed models. In the perspective of increasing the size of the EPC source dataset, a robust pre-processing layer need to be deployed in order to avoid a decrease of model performance over time. In addition, still considering the deployment phase, the re-training of three binary classification models instead of one multiclass classifier, represents a task with higher computational cost. Respect to the future work, further research will be focused on the testing of alternative configurations of algorithms (i.e., classifiers, clustering algorithms, XAI tools) with respect to the one considered in this study by performing a benchmark analysis between them especially from the explainability point of view. Eventually, particularly promising is the integration of a XAI layer also in other building Energy Management and Information System (EMIS) tools that typically leverage AI techniques such as advanced controllers and automated anomaly detection and diagnosis systems. Finally, as discussed in Chapter XXX, the raise of APs and the spread of XAI in critical contexts are pushing researchers in analysing the risks that the former can have on the latter. If, on one hand, some works are analysing this problematic relation [46][78][203], on the other some authors are already designing attacks in critical scenarios [96]. The aim of this thesis was to increase awareness on the risks associated with the use of XAI in critical contexts by providing a quantitative analysis of the impact that adversarial perturbations have on XAI methods in terms of differences in the explainability maps. To this aim, we defined a set of experiments intended to measure the obtained impact while producing outcomes easy to understand. To help to match the latter point, we focused on the image classification task for its intuitive interpretation. Nonetheless, to make the reported analysis rigorous, we considered four different Convolutional Neural Networks (CNNs), two AP algorithms, two XAI methods and two public datasets. Reported results show that APs can strongly affect the XAI outcomes, even in the case of a failed attack. In particular, the Correlation Coefficient (CC) and Dice Similarity Coefficient (DSC) distributions suggest not only that the injected

perturbations are able to mislead the target CNN, but also that the effects on the XAI maps are not deterministic. Indeed, while the maps themselves are different before and after the perturbation, this difference is not easily perceptible by a human operator unaware of the attack, since the portion of the images that support the decision taken by the networks are almost the same. Interestingly, this happens for all the considered networks and for all the considered XAI and AP algorithms. Even the number of classes in the used dataset does not seem to affect the reported behaviour. Future works will further analyse the reported aspects, also by considering the effects on bigger datasets (with hundred of classes) and by taking into account black-box AP algorithms.

Bibliography

[1] Adadi, A. and Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160.

[2] Akhlaghi, Y. G., Aslansefat, K., Zhao, X., Sadati, S., Badiei, A., Xiao, X., Shittu, S., Fan, Y., and Ma, X. (2020). Hourly performance forecast of a dew point cooler using explainable artificial intelligence and evolutionary optimisations by 2050. *Applied Energy*, 281:116062.

[3] Akhtar, N. and Mian, A. (2018). Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430.

[4] Al-Homoud, M. S. (2001). Computer-aided building energy analysis techniques. *Building and Environment*, 36(4):421–433.

[5] Alfeo, A. L., Cimino, M. G., Manco, G., Ritacco, E., and Vaglini, G. (2020). Using an autoencoder in the design of an anomaly detector for smart manufacturing. *Pattern Recognition Letters*, 136:272–278.

[6] Ali, S., Rauf, A., Islam, N., Farman, H., and Khan, S. (2017). User profiling: A privacy issue in online public network. *Sindh University Research Journal-SURJ (Science Series)*, 49(1).

[7] Allen, B. (2004). Monitoring hard disks with smart. *Linux Journal*, (117):74–77.

[8] Amasyali, K. and El-Gohary, N. M. (2018). A review of data-driven building energy consumption prediction studies. *Renewable and Sustainable Energy Reviews*, 81:1192–1205.

[9] Amoroso, D. and Tamburrini, G. (2017). The ethical and legal case against autonomy in weapons systems. *Global Jurist*, 18(1).

[10] Anantharaman, P., Qiao, M., and Jadav, D. (2018). Large scale predictive analytics for hard disk remaining useful life estimation. In *2018 IEEE International Congress on Big Data (BigData Congress)*, pages 251–254. IEEE.

[11] Arjunan, P., Poolla, K., and Miller, C. (2020). Energystar++: Towards more accurate and explanatory building energy benchmarking. *Applied Energy*, 276:115413.

[12] Arjunan, P., Poolla, K., and Miller, C. (2022). Beem: Data-driven building energy benchmarking for singapore. *Energy and Buildings*, 260:111869.

[13] Assael, Y. M., Shillingford, B., Whiteson, S., and de Freitas, N. (2016). Lipnet: End-to-end sentence-level lipreading.

[14] Attanasio, A., Piscitelli, M. S., Chiusano, S., Capozzoli, A., and Cerquitelli, T. (2019). Towards an automated, fast and interpretable estimation model of heating energy demand: A data-driven approach exploiting building energy certificates. *Energies*, 12(7):1273.

[15] Aussel, N., Jaulin, S., Gandon, G., Petetin, Y., Fazli, E., and Chabridon, S. (2017). Predictive models of hard drive failures based on operational data. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 619–625. IEEE.

[16] Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and MÃžller, K.-R. (2010). How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun):1803–1831.

[17] Bala, A. and Chana, I. (2015). Intelligent failure prediction models for scientific workflows. *Expert Systems with Applications*, 42(3):980–989.

[18] Ballings, M., Van den Poel, D., Hespeels, N., and Gryp, R. (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications*, 42(20):7046–7056.

[19] Barreto, L., Amaral, A., and Pereira, T. (2017). Industry 4.0 implications in logistics: an overview. *Procedia Manufacturing*, 13:1245–1252.

[20] Basak, S., Sengupta, S., and Dubey, A. (2019). Mechanisms for integrated feature normalization and remaining useful life estimation using lstms applied to hard-disks. In *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 208–216.

[Bauer] Bauer, R. HDD vs SSD: What does the future for storage hold? https://www.backblaze.com/blog/ssd-vs-hdd-future-of-storage/. Accessed: 2019-10-04.

[22] Bayram, B., Duman, T. B., and Ince, G. (2021). Real time detection of acoustic anomalies in industrial processes using sequential autoencoders. *Expert Systems*, 38(1).

[23] Biran, O. and Cotton, C. (2017). Explanation and justification in machine learning: A survey. In *IJCAI-17 workshop on explainable AI (XAI)*, volume 8, pages 8–13.

[24] Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., et al. (2016). End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*.

[25] Botezatu, M. M., Giurgiu, I., Bogojeska, J., and Wiesmann, D. (2016). Predicting disk replacement towards reliable data centers. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 39–48.

[26] Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

[27] Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

[28] Calvo-Bascones, P., Sanz-Bobi, M. A., and Welte, T. M. (2021). Anomaly detection method based on the deep knowledge behind behavior patterns in industrial components. application to a hydropower plant. *Computers in Industry*, 125:103376.

[29] Capozzoli, A., Grassi, D., and Causone, F. (2015). Estimation models of heating energy consumption in schools for local authorities planning. *Energy and Buildings*, 105:302–313.

[30] Capozzoli, A., Piscitelli, M. S., Neri, F., Grassi, D., and Serale, G. (2016a). A novel methodology for energy performance benchmarking of buildings by means of linear mixed effect model: The case of space and dhw heating of out-patient healthcare centres. *Applied Energy*, 171:592–607.

[31] Capozzoli, A., Serale, G., Piscitelli, M. S., and Grassi, D. (2016b). Data mining for energy analysis of a large data set of flats. In *Proceedings of the Institution of Civil Engineers-Engineering Sustainability*, volume 170, pages 3–18. Thomas Telford Ltd.

[32] Carlini, N. and Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE.

[33] Cath, C., Wachter, S., Mittelstadt, B., Taddeo, M., and Floridi, L. (2018). Artificial intelligence and the 'good society': the us, eu, and uk approach. *Science and engineering ethics*, 24(2):505–528.

[34] Chalapathy, R. and Chawla, S. (2019). Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*.

[35] Challen, R., Denny, J., Pitt, M., Gompels, L., Edwards, T., and Tsaneva-Atanasova, K. (2019). Artificial intelligence, bias and clinical safety. *BMJ Quality & Safety*, 28(3):231–237.

[36] Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3).

[37] Chandrakala, S. and Jayalakshmi, S. L. (2019). Environmental audio scene and sound event recognition for autonomous surveillance: A survey and comparative studies. *ACM Comput. Surv.*, 52(3).

[38] Chen, H.-Y. and Lee, C.-H. (2020). Vibration signals analysis by explainable artificial intelligence (xai) approach: Application on bearing faults diagnosis. *IEEE Access*, 8:134246–134256.

[39] Chen, J.-C., Patel, V. M., and Chellappa, R. (2016). Unconstrained face verification using deep cnn features. In *2016 IEEE winter conference on applications of computer vision (WACV)*, pages 1–9. IEEE.

[40] Chen, T., Liu, X., Xia, B., Wang, W., and Lai, Y. (2020). Unsupervised anomaly detection of industrial robots using sliding-window convolutional variational autoencoder. *IEEE Access*, 8:47072–47081.

[41] Chugh, T., Cao, K., and Jain, A. K. (2018). Fingerprint spoof buster: Use of minutiae-centered patches. *IEEE Transactions on Information Forensics and Security*, 13(9):2190–2202.

[42] Chung, W. (2011). Review of building energy-use performance benchmarking methodologies. *Applied Energy*, 88(5):1470–1479.

[43] Cinque, M., Della Corte, R., Moscato, V., and Sperlí, G. (2021). A graph-based approach to detect unexplained sequences in a log. *Expert Systems with Applications*, 171:114556.

[Coughlin] Coughlin, T. Mamr hard disk drives enable future data centers. https://www.forbes.com/sites/tomcoughlin/2017/10/11/mamr-hard-disk-drives-enable-future-data-centers/#77f5702e2054. Accessed: 2019-06-24.

[45] Dalzochio, J., Kunst, R., Pignaton, E., Binotto, A., Sanyal, S., Favilla, J., and Barbosa, J. (2020). Machine learning and reasoning for predictive maintenance in industry 4.0: Current status and challenges. *Computers in Industry*, 123:103298.

[46] Das, A. and Rad, P. (2020). Opportunities and challenges in explainable artificial intelligence (xai): A survey. *arXiv preprint arXiv:2006.11371*.

[47] Datta, A., Tschantz, M. C., and Datta, A. (2015). Automated experiments on ad privacy settings. *Proceedings on Privacy Enhancing Technologies*, 2015(1):92–112.

[48] Di Corso, E., Cerquitelli, T., Piscitelli, M. S., and Capozzoli, A. (2017). Exploring energy certificates of buildings through unsupervised data mining techniques. In *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 991–998. IEEE.

[49] Dong, Y., Liao, F., Pang, T., Su, H., Hu, X., Li, J., and Zhu, J. (2017). Boosting adversarial attacks with momentum. arxiv preprint. *arXiv preprint arXiv:1710.06081*.

[50] Došilović, F. K., Brčić, M., and Hlupić, N. (2018). Explainable artificial intelligence: A survey. In *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*, pages 0210–0215. IEEE.

[51] Elson, J., Douceur, J. J., Howell, J., and Saul, J. (2007). Asirra: a captcha that exploits interest-aligned manual image categorization.

[52] Eskandarnia, E., Al-Ammal, H. M., and Ksantini, R. (2022). An embedded deep-clustering-based load profiling framework. *Sustainable Cities and Society*, 78:103618.

[53] Fabrizio, E., Corrado, V., and Filippi, M. (2010). A model to design and optimize multi-energy systems in buildings at the design concept stage. *Renewable Energy*, 35(3):644–655.

[54] Fan, C., Sun, Y., Zhao, Y., Song, M., and Wang, J. (2019a). Deep learning-based feature engineering methods for improved building energy prediction. *Applied energy*, 240:35–45.

[55] Fan, C., Xiao, F., Yan, C., Liu, C., Li, Z., and Wang, J. (2019b). A novel methodology to explain and evaluate data-driven building energy performance models based on interpretable machine learning. *Applied Energy*, 235:1551–1560.

[56] Fernando, T., Gammulle, H., Denman, S., Sridharan, S., and Fookes, C. (2021). Deep learning for medical anomaly detection – a survey. *ACM Comput. Surv.*, 54(7).

[57] Fidel, G., Bitton, R., and Shabtai, A. (2020). When explainability meets adversarial learning: Detecting adversarial examples using shap signatures. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

[58] Filogamo, L., Peri, G., Rizzo, G., and Giaccone, A. (2014). On the classification of large residential buildings stocks by sample typologies for energy planning purposes. *Applied Energy*, 135:825–835.

[59] Fink, O., Wang, Q., Svensen, M., Dersin, P., Lee, W.-J., and Ducoffe, M. (2020). Potential, challenges and future directions for deep learning in prognostics and health management applications. *Engineering Applications of Artificial Intelligence*, 92:103678.

[60] Frick, N. M., Schiller, S., Stuart, E., Schwartz, L., Kramer, C., and Faesy, R. (2017). Evaluation of u.s. building energy benchmarking and transparency programs: Attributes, impacts, and best practices. Technical report.

[61] Gao, X. and Malkawi, A. (2014). A new methodology for building energy performance benchmarking: An approach based on intelligent clustering algorithm. *Energy and Buildings*, 84:607–616.

[62] Geraldi, M. S. and Ghisi, E. (2022). Data-driven framework towards realistic bottom-up energy benchmarking using an artificial neural network. *Applied Energy*, 306:117960.

[63] Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine learning*, 63(1):3–42.

[64] Gibson, G. A. and Patterson, D. A. (1993). Designing disk arrays for high data reliability. *Journal of Parallel and Distributed Computing*, 17(1-2):4–27.

[65] Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

[66] Gravina, M., Marrone, S., Piantadosi, G., Sansone, M., and Sansone, C. (2019). 3tp-cnn: Radiomics and deep learning for lesions classification in dce-mri. In *International Conference on Image Analysis and Processing*, pages 661–671. Springer.

[67] Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., and Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42.

[68] Guiochet, J., Machin, M., and Waeselynck, H. (2017). Safety-critical advanced robots: A survey. *Robotics and Autonomous Systems*, 94:43–52.

[69] Gunning, D. (2017). Explainable artificial intelligence (xai). *DARPA*, 2.

[70] Guo, S., Yang, T., Gao, W., and Zhang, C. (2018). A novel fault diagnosis method for rotating machinery based on a convolutional neural network. *Sensors*, 18(5):1429.

[71] Hamerly, G., Elkan, C., et al. (2001). Bayesian approaches to failure prediction for disk drives. In *ICML*, volume 1, pages 202–209.

[72] Hayashi, T., Yoshimura, T., and Adachi, Y. (2020). Conformer-based id-aware autoencoder for unsupervised anomalous sound detection. Technical report, Tech. Rep., DCASE2020 Challenge.

[73] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[74] Himeur, Y., Ghanem, K., Alsalemi, A., Bensaali, F., and Amira, A. (2021). Artificial intelligence based anomaly detection of energy consumption in buildings: A review, current trends and new perspectives. *Applied Energy*, 287:116601.

[75] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

[76] Hong, T., Piette, M. A., Chen, Y., Lee, S. H., Taylor-Lange, S. C., Zhang, R., Sun, K., and Price, P. (2015). Commercial building energy saver: an energy retrofit analysis toolkit. *Applied Energy*, 159:298–309.

[77] Hsieh, R.-J., Chou, J., and Ho, C.-H. (2019). Unsupervised online anomaly detection on multivariate sensing time series data for smart manufacturing. In *2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA)*, pages 90–97.

[78] Ignatiev, A., Narodytska, N., and Marques-Silva, J. (2019). On relating explanations and adversarial examples. In *Advances in Neural Information Processing Systems*, pages 15883–15893.

[79] ISO-15927 (2003). Iso 15927-1, hygrothermal performance of buildings-calculation and presentation of climatic data - part 1: Monthly means of single meteorological elements. *International Organization for Standardization*.

[80] Jalali, A., Schindler, A., and Haslhofer, B. (2020). Dcase challenge 2020: Unsupervised anomalous sound detection of machinery with deep autoencoders.

[81] Jalayer, M., Orsenigo, C., and Vercellis, C. (2021a). Fault detection and diagnosis for rotating machinery: A model based on convolutional lstm, fast fourier and continuous wavelet transforms. *Computers in Industry*, 125:103378.

[82] Jalayer, M., Orsenigo, C., and Vercellis, C. (2021b). Fault detection and diagnosis for rotating machinery: A model based on convolutional lstm, fast fourier and continuous wavelet transforms. *Computers in Industry*, 125:103378.

[83] Kalmet, P. H., Sanduleanu, S., Primakov, S., Wu, G., Jochems, A., Refaee, T., Ibrahim, A., Hulst, L. v., Lambin, P., and Poeze, M. (2020). Deep learning in fracture detection: a narrative review. *Acta Orthopaedica*, pages 1–6.

[84] Kapka, S. (2020). Id-conditioned auto-encoder for unsupervised anomaly detection. *arXiv preprint arXiv:2007.05314*.

[85] Kelly, L., Sachan, S., Ni, L., Almaghrabi, F., Allmendinger, R., and Chen, Y.-W. (2020). Explainable artificial intelligence for digital forensics: Opportunities, challenges and a drug testing case study. In *Digital Forensic Science*. IntechOpen.

[86] Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[87] Klein, A. Backblaze Hard Drive stats Q1 2019. https://www.backblaze.com/blog/backblaze-hard-drive-stats-q1-2019/. Accessed: 2019-10-04.

[88] Klein, A. The shocking truth: Managing for Hard Drive failure and data corruption. https://www.backblaze.com/blog/managing-for-hard-drive-failures-data-corruption/. Accessed: 2019-10-04.

[89] Koizumi, Y., Kawaguchi, Y., Imoto, K., Nakamura, T., Nikaido, Y., Tanabe, R., Purohit, H., Suefusa, K., Endo, T., Yasuda, M., et al. (2020). Description and discussion on dcase2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring. *arXiv preprint arXiv:2006.05822*.

[90] Koizumi, Y., Murata, S., Harada, N., Saito, S., and Uematsu, H. (2019). Sniper: Few-shot learning for anomaly detection to minimize false-negative rate with ensured true-positive rate. pages 915–919.

[91] Kokhlikyan, N., Miglani, V., Martin, M., Wang, E., Alsallakh, B., Reynolds, J., Melnikov, A., Kliushkina, N., Araya, C., Yan, S., et al. (2020). Captum: A unified and generic model interpretability library for pytorch. *arXiv preprint arXiv:2009.07896*.

[92] Kolosnjaji, B., Zarras, A., Webster, G., and Eckert, C. (2016). Deep learning for classification of malware system call sequences. In *Australasian Joint Conference on Artificial Intelligence*, pages 137–149. Springer.

[93] Kong, Z., Cui, Y., Xia, Z., and Lv, H. (2019). Convolution and long short-term memory hybrid deep neural networks for remaining useful life prognostics. *Applied Sciences*, 9(19):4156.

[94] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

[95] Kumarl Ibrahim Ben Daya, D., Vats, K., Feng, J., Taylor, G., and Wong, A. (2019). Beyond explainability: Leveraging interpretability for improved adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 16–19.

[96] Kuppa, A. and Le-Khac, N.-A. (2020). Black box attacks on explainable artificial intelligence (xai) methods in cyber security. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

[97] Kurakin, A., Goodfellow, I., and Bengio, S. (2016). Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.

[98] Kuzlu, M., Cali, U., Sharma, V., and Güler, Ö. (2020). Gaining insight into solar photovoltaic power generation forecasting utilizing explainable artificial intelligence tools. *IEEE Access*, 8:187814–187823.

[99] Lee, J., Davari, H., Singh, J., and Pandhare, V. (2018). Industrial artificial intelligence for industry 4.0-based manufacturing systems. *Manufacturing letters*, 18:20–23.

[100] Lee, S. H., Hong, T., Piette, M. A., and Taylor-Lange, S. C. (2015). Energy retrofit analysis toolkits for commercial buildings: A review. *Energy*, 89:1087–1100.

[101] Lee, W., Yik, F., and Burnett, J. (2003). A method to assess the energy performance of existing commercial complexes. *Indoor and Built Environment*, 12(5):311–327.

[102] Lee, W.-S. and Lee, K.-P. (2009). Benchmarking the performance of building energy management using data envelopment analysis. *Applied Thermal Engineering*, 29(16):3269–3273.

[103] Li, J., Ji, X., Jia, Y., Zhu, B., Wang, G., Li, Z., and Liu, X. (2014a). Hard drive failure prediction using classification and regression trees. In *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 383–394. IEEE.

[104] Li, K., Sun, Y., Robinson, D., Ma, J., and Ma, Z. (2020). A new strategy to benchmark and evaluate building electricity usage using multiple data mining technologies. *Sustainable Energy Technologies and Assessments*, 40:100770.

[105] Li, L.-J. and Fei-Fei, L. (2007). What, where and who? classifying events by scene and object recognition. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

[106] Li, Y., Li, X., Zhang, Y., Liu, M., and Wang, W. (2018). Anomalous sound detection using deep audio representation and a blstm network for audio surveillance of roads. *IEEE Access*, 6:58043–58055.

[107] Li, Z., Han, Y., and Xu, P. (2014b). Methods for benchmarking building energy consumption against its past or intended performance: An overview. *Applied Energy*, 124:325–334.

[108] Lindemann, B., Maschler, B., Sahlab, N., and Weyrich, M. (2021). A survey on anomaly detection for technical systems using lstm networks. *Computers in Industry*, 131:103498.

[109] Linzen, T., Dupoux, E., and Goldberg, Y. (2016). Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

[110] Liu, X., Ding, Y., Tang, H., and Xiao, F. (2021). A data mining-based framework for the identification of daily electricity usage patterns and anomaly detection in building electricity consumption data. *Energy and Buildings*, 231:110601.

[111] Lom, M., Pribyl, O., and Svitek, M. (2016). Industry 4.0 as a part of smart cities. In *2016 Smart Cities Symposium Prague (SCSP)*, pages 1–6. IEEE.

[112] Louppe, G. (2014). Understanding random forests: From theory to practice. *arXiv preprint arXiv:1407.7502*.

[113] Lundberg, S. M. and Lee, S.-I. (2017a). A unified approach to interpreting model predictions. In *NIPS 30*, pages 4765–4774.

[114] Lundberg, S. M. and Lee, S.-I. (2017b). A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777.

[115] Madhikermi, M., Malhi, A. K., and Främling, K. (2019). Explainable artificial intelligence based heat recycler fault detection in air handling unit. In *International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems*, pages 110–125. Springer.

[116] Mahdisoltani, F., Stefanovici, I., and Schroeder, B. (2017). Proactive error prediction to improve storage system reliability. In *2017 ATC 17)*, pages 391–402.

[117] Manogaran, G., Thota, C., Lopez, D., and Sundarasekar, R. (2017). Big data security intelligence for healthcare industry 4.0. In *Cybersecurity for Industry 4.0*, pages 103–126. Springer.

[118] Manyika, J. and Bughin, J. (2018). The promise and challenge of the age of artificial intelligence. *McKinsey Global Institute Executive Briefing*.

[119] Marrone, S. and Sansone, C. (2019a). An adversarial perturbation approach against cnn-based soft biometrics detection. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

[120] Marrone, S. and Sansone, C. (2019b). Adversarial perturbations against fingerprint based authentication systems. In *2019 International Conference on Biometrics (ICB)*, pages 1–6. IEEE.

[121] Mauro, G. M., Hamdy, M., Vanoli, G. P., Bianco, N., and Hensen, J. L. (2015). A new methodology for investigating the cost-optimality of energy retrofitting a building category. *Energy and Buildings*, 107:456–478.

[122] Meire, M. and Karsmakers, P. (2019). Comparison of deep autoencoder architectures for real-time acoustic based anomaly detection in assets. In *2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, volume 2, pages 786–790.

[123] Millar, D., Tonolo, G., and Ziebinska, U. (2016). Energy efficiency indicators: Highlights. *International Energy Agency (IEA), Paris, France*.

[124] Miller, C. (2019a). More buildings make more generalizable models—benchmarking prediction methods on open electrical meter data. *Machine Learning and Knowledge Extraction*, 1(3):974–993.

[125] Miller, C. (2019b). What's in the box?! towards explainable machine learning applied to non-residential building smart meter classification. *Energy and Buildings*, 199:523–536.

[126] Minsky, M. L. (1967). *Computation*. Prentice-Hall Englewood Cliffs.

[127] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

[128] Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. (2016). Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2574–2582.

[129] Murray, J. F., Hughes, G. F., and Kreutz-Delgado, K. (2005). Machine learning methods for predicting failures in hard drives: A multiple-instance application. *Journal of Machine Learning Research*, 6(May):783–816.

[130] Nicolae, M.-I., Sinn, M., Tran, M. N., Buesser, B., Rawat, A., Wistuba, M., Zantedeschi, V., Baracaldo, N., Chen, B., Ludwig, H., et al. (2018). Adversarial robustness toolbox v1. 0.0. *arXiv preprint arXiv:1807.01069*.

[131] Nunes, E. C. (2021). Anomalous sound detection with machine learning: A systematic review. *arXiv preprint arXiv:2102.07820*.

[132] Pang, G., Cao, L., and Aggarwal, C. (2021a). Deep learning for anomaly detection: Challenges, methods, and opportunities. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, page 1127–1130, New York, NY, USA. Association for Computing Machinery.

[133] Pang, G., Shen, C., Cao, L., and Hengel, A. V. D. (2021b). Deep learning for anomaly detection: A review. *ACM Comput. Surv.*, 54(2).

[134] Papadopoulos, S. and Kontokosta, C. E. (2019). Grading buildings on energy performance using city benchmarking data. *Applied Energy*, 233:244–253.

[135] Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. (2016). The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 372–387. IEEE.

[136] Park, Y. and Yun, I. D. (2018a). Fast adaptive rnn encoder–decoder for anomaly detection in smd assembly machine. *Sensors*, 18(10).

[137] Park, Y. and Yun, I. D. (2018b). Fast adaptive rnn encoder–decoder for anomaly detection in smd assembly machine. *Sensors*, 18(10).

[138] Pasichnyi, O., Wallin, J., Levihn, F., Shahrokni, H., and Kordas, O. (2019). Energy performance certificates—new opportunities for data-enabled urban energy policy instruments? *Energy Policy*, 127:486–499.

[139] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037.

[140] Pereira, S., Meier, R., Alves, V., Reyes, M., and Silva, C. A. (2018). Automatic brain tumor grading from mri data using convolutional neural networks and quality assessment. In *Understanding and interpreting machine learning in medical image computing applications*, pages 106–114. Springer.

[141] Petcharat, S., Chungpaibulpatana, S., and Rakkwamsuk, P. (2012). Assessment of potential energy saving using cluster analysis: A case study of lighting systems in buildings. *Energy and Buildings*, 52:145–152.

[142] Piscitelli, M. S., Mazzarelli, D. M., and Capozzoli, A. (2020). Enhancing operational performance of ahus through an advanced fault detection and diagnosis process based on temporal association and decision rules. *Energy and Buildings*, 226:110369.

[143] Polletta, F. and Callahan, J. (2019). Deep stories, nostalgia narratives, and fake news: Storytelling in the trump era. In *Politics of meaning/meaning of politics*, pages 55–73. Springer.

[144] Ponemon, L. (2016). Cost of data center outages. *Data Center Performance Benchmark Serie*.

[145] Purohit, H., Tanabe, R., Ichige, K., Endo, T., Nikaido, Y., Suefusa, K., and Kawaguchi, Y. (2019). Mimii dataset: Sound dataset for malfunctioning industrial machine investigation and inspection. *arXiv preprint arXiv:1909.09347*.

[146] Qiao, Z., Hochstetler, J., Liang, S., Fu, S., Chen, H.-b., and Settlemyer, B. (2018). Developing cost-effective data rescue schemes to tackle disk failures in data centers. In *International Conference on Big Data*, pages 194–208. Springer.

[147] Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1(1):81–106.

[148] Ribeiro, A., Matos, L. M., Pereira, P. J., Nunes, E. C., Ferreira, A. L., Cortez, P., and Pilastri, A. (2020). Deep dense and convolutional autoencoders for unsupervised anomaly detection in machine condition sounds. *arXiv preprint arXiv:2006.10417*.

[149] Ribeiro, M., Lazzaretti, A. E., and Lopes, H. S. (2018). A study of deep convolutional auto-encoders for anomaly detection in videos. *Pattern Recognition Letters*, 105:13–22.

[150] Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *22nd ACM SIGKDD*, pages 1135–1144.

[151] Robnik-Šikonja, M. and Kononenko, I. (2003). Theoretical and empirical analysis of relieff and rrelieff. *Machine learning*, 53(1-2):23–69.

[152] Roth, J., Lim, B., Jain, R. K., and Grueneich, D. (2020). Examining the feasibility of using open data to benchmark building energy usage in cities: A data science and policy perspective. *Energy Policy*, 139:111327.

[153] Rudin, C., Waltz, D., Anderson, R. N., Boulanger, A., Salleb-Aouissi, A., Chow, M., Dutta, H., Gross, P. N., Huang, B., Ierome, S., et al. (2011). Machine learning for the new york city power grid. *IEEE transactions on pattern analysis and machine intelligence*, 34(2):328–345.

[154] Runge, J. and Zmeureanu, R. (2019). Forecasting energy use in buildings using artificial neural networks: A review. *Energies*, 12(17):3254.

[155] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

[156] Sabour, S., Cao, Y., Faghri, F., and Fleet, D. J. (2015). Adversarial manipulation of deep representations. *arXiv preprint arXiv:1511.05122*.

[157] Sankar, S., Shaw, M., Vaid, K., and Gurumurthi, S. (2013). Datacenter scale evaluation of the impact of temperature on hard disk drive failures. *ACM TOS*, 9(2):1–24.

[158] Sardianos, C., Varlamis, I., Chronis, C., Dimitrakopoulos, G., Alsalemi, A., Himeur, Y., Bensaali, F., and Amira, A. (2021). The emergence of explainability of intelligent systems: Delivering explainable and personalized recommendations for energy efficiency. *International Journal of Intelligent Systems*, 36(2):656–680.

[159] Schlingensiepen, J., Nemtanu, F., Mehmood, R., and McCluskey, L. (2016). Autonomic transport management systems—enabler for smart cities, personalized medicine, participation and industry grid/industry 4.0. In *Intelligent transportation systems–problems and perspectives*, pages 3–35. Springer.

[160] Schwendemann, S., Amjad, Z., and Sikora, A. (2021). A survey of machine-learning techniques for condition monitoring and predictive maintenance of bearings in grinding machines. *Computers in Industry*, 125:103380.

[161] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626.

[162] Sengupta, S., Basak, S., Saikia, P., Paul, S., Tsalavoutis, V., Atiah, F., Ravi, V., and Peters, A. (2020). A review of deep learning with special emphasis on architectures, applications and recent trends. *Knowledge-Based Systems*, page 105596.

[163] Shao, H., Jiang, H., Lin, Y., and Li, X. (2018). A novel method for intelligent fault diagnosis of rolling bearings using ensemble deep auto-encoders. *Mechanical Systems and Signal Processing*, 102:278–297.

[164] Shao, H. and Soong, B.-H. (2016). Traffic flow prediction with long short-term memory networks (lstms). In *2016 IEEE Region 10 Conference (TENCON)*, pages 2986–2989. IEEE.

[165] Shen, J., Wan, J., Lim, S.-J., and Yu, L. (2018). Random-forest-based failure prediction for hard disk drives. *International Journal of Distributed Sensor Networks*, 14(11).

[166] Simmhan, Y., Aman, S., Kumbhare, A., Liu, R., Stevens, S., Zhou, Q., and Prasanna, V. (2013). Cloud-based software platform for big data analytics in smart grids. *Computing in Science & Engineering*, 15(4):38.

[167] Sinwar, D., Sharma, M. K., and Verma, H. (2020). Remote sensing classification under deep learning: A review. In *Smart Systems and IoT: Innovations in Computing*, pages 813–823. Springer.

[168] Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.

[169] Stock, T. and Seliger, G. (2016). Opportunities of sustainable manufacturing in industry 4.0. *Procedia Cirp*, 40:536–541.

[170] Su, C.-J. and Huang, S.-F. (2018). Real-time big data analytics for hard disk drive predictive maintenance. *Computers & Electrical Engineering*, 71:93–101.

[171] Su, C.-J. and Li, Y. (2019). Recurrent neural network based real-time failure detection of storage devices. *Microsystem Technologies*.

[172] Su, J., Vargas, D. V., and Kouichi, S. (2017). One pixel attack for fooling deep neural networks. *arXiv preprint arXiv:1710.08864*.

[173] Su, J., Vargas, D. V., and Sakurai, K. (2019). One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841.

[174] Sun, X., Chakrabarty, K., Huang, R., Chen, Y., Zhao, B., Cao, H., Han, Y., Liang, X., and Jiang, L. (2019). System-level hardware failure prediction using deep learning. In *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE.

[175] Sun, Y., Haghighat, F., and Fung, B. C. (2020). A review of the-state-of-the-art in data-driven approaches for building energy prediction. *Energy and Buildings*, 221:110022.

[176] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

[177] Tahsildoost, M. and Zomorodian, Z. S. (2015). Energy retrofit techniques: An experimental study of two typical school buildings in tehran. *Energy and Buildings*, 104:65–72.

[178] Taigman, Y., Yang, M., Ranzato, M., and Wolf, L. (2014). Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708.

[179] Takeichi, N., Kaida, R., Shimomura, A., and Yamauchi, T. (2017). Prediction of delay due to air traffic control by machine learning. In *AIAA Modeling and Simulation Technologies Conference*, page 1323.

[180] Tan, M. and Le, Q. V. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*.

[181] Tan, P.-N., Steinbach, M., and Kumar, V. (2013). Data mining cluster analysis: basic concepts and algorithms. *Introduction to data mining*, pages 487–533.

[182] Thudumu, S., Branch, P., Jin, J., and Singh, J. J. (2020). A comprehensive survey of anomaly detection techniques for high dimensional big data. *Journal of Big Data*, 7(1):1–30.

[183] Thuemmler, C. and Bai, C. (2017). Health 4.0: Application of industry 4.0 design principles in future asthma management. In *Health 4.0: How virtualization and big data are revolutionizing healthcare*, pages 23–37. Springer.

[184] Turing, A. M. (2009). Computing machinery and intelligence. In *Parsing the Turing Test*, pages 23–65. Springer.

[185] UNI-10349 (2016). Uni 10349-1:2016, heating and cooling of buildings - climatic data - part 1: Monthly means for evaluation of energy need for space heating and cooling and methods for splitting global solar irradiance into the direct and diffuse parts and for calculate the solar irradiance on tilted planes. *Italian Organization for Standardization*.

[186] UNI/TS-11300 (2008). Uni/ts 11300-2, energy performance of buildings-part 2. *Italian Organization for Standardization*.

[187] Vafeiadis, A., Votis, K., Giakoumis, D., Tzovaras, D., Chen, L., and Hamzaoui, R. (2020). Audio content analysis for unobtrusive event detection in smart homes. *Engineering Applications of Artificial Intelligence*, 89:103226.

[188] von Birgelen, A., Buratti, D., Mager, J., and Niggemann, O. (2018). Self-organizing maps for anomaly localization and predictive maintenance in cyber-physical production systems. *Procedia CIRP*, 72:480–485. 51st CIRP Conference on Manufacturing Systems.

[189] Wang, S., Yan, C., and Xiao, F. (2012). Quantitative energy performance assessment methods for existing buildings. *Energy and buildings*, 55:873–888.

[190] Wang, Y., Ma, E. W., Chow, T. W., and Tsui, K.-L. (2013). A two-step parametric method for failure prediction in hard disk drives. *IEEE Transactions on industrial informatics*, 10(1):419–430.

[191] Watkins, D., Gallardo, G., and Chau, S. (2018). Pilot support system: A machine learning approach. In *2018 IEEE 12th International Conference on Semantic Computing (ICSC)*, pages 325–328. IEEE.

[192] Weld, D. S. and Bansal, G. (2019). The challenge of crafting intelligible intelligence. *Communications of the ACM*, 62(6):70–79.

[193] Wu, J., Zhao, Z., Sun, C., Yan, R., and Chen, X. (2020). Fault-attention generative probabilistic adversarial autoencoder for machine anomaly detection. *IEEE Transactions on Industrial Informatics*, 16(12):7479–7488.

[194] Wüstrich, L., Schröder, L., and Pahl, M.-O. (2021). Cyber-physical anomaly detection for ics. In *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 950–955. IEEE.

[195] Xiao, J., Xiong, Z., Wu, S., Yi, Y., Jin, H., and Hu, K. (2018). Disk failure prediction in data centers via online learning. In *Proceedings of the 47th International Conference on Parallel Processing*, page 35. ACM.

[196] Xie, Q., Hovy, E., Luong, M.-T., and Le, Q. V. (2019a). Self-training with noisy student improves imagenet classification. *arXiv preprint arXiv:1911.04252*.

[197] Xie, Y., Feng, D., Wang, F., Tang, X., Han, J., and Zhang, X. (2019b). Dfpe: Explaining predictive models for disk failure prediction. In *35th MSST*, pages 193–204. IEEE.

[198] Xu, C., Wang, G., Liu, X., Guo, D., and Liu, T.-Y. (2016). Health status assessment and failure prediction for hard drives with recurrent neural networks. *IEEE Transactions on Computers*, 65(11):3502–3508.

[199] Xu, Y., Sui, K., Yao, R., Zhang, H., Lin, Q., Dang, Y., Li, P., Jiang, K., Zhang, W., Lou, J.-G., et al. (2018). Improving service availability of cloud systems by predicting disk error. In *2018 {USENIX} Annual Technical Conference ({USENIX}{ATC} 18)*, pages 481–494.

[200] Yang, Z., Roth, J., and Jain, R. K. (2018). Due-b: Data-driven urban energy benchmarking of buildings using recursive partitioning and stochastic frontier analysis. *Energy and Buildings*, 163:58–69.

[201] Yao, G., Lei, T., and Zhong, J. (2019). A review of convolutional-neural-network-based action recognition. *Pattern Recognition Letters*, 118:14–22.

[202] Yin, C., Zhang, S., Wang, J., and Xiong, N. N. (2020). Anomaly detection based on convolutional recurrent autoencoder for iot time series. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

[203] Yoon, J., Kim, K., and Jang, J. (2019). Propagated perturbation of adversarial attack for well-known cnns: Empirical study and its explanation. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 4226–4234. IEEE.

[204] Zaytar, M. A. and El Amrani, C. (2016). Sequence to sequence weather forecasting with iong short-term memory recurrent neural networks. *International Journal of Computer Applications*, 143(11):7–11.

[205] Zhang, J., Wang, J., He, L., Li, Z., and Philip, S. Y. (2018). Layerwise perturbation-based adversarial training for hard drive health degree prediction. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 1428–1433. IEEE.

[206] Zhang, Y., O'Neill, Z., Dong, B., and Augenbroe, G. (2015). Comparisons of inverse modeling approaches for predicting building energy performance. *Building and Environment*, 86:177–190.

[207] Zhao, Y., Zhang, C., Zhang, Y., Wang, Z., and Li, J. (2020). A review of data mining technologies in building energy systems: Load prediction, pattern identification, fault detection and diagnosis. *Energy and Built Environment*, 1(2):149–164.

[208] Zhu, B., Wang, G., Liu, X., Hu, D., Lin, S., and Ma, J. (2013). Proactive drive failure prediction for large scale storage systems. In *2013 IEEE 29th Symposium on Mass Storage Systems and Technologies (MSST)*, pages 1–5. IEEE.

List of figures

List of tables