Université de BLIDA 1
Faculté des Sciences
Département d'Informatique

Master Thesis

Option: *Ingénerie de Logiciel*

ABNORMAL SOUND DETECTION FOR MACHINE CONDITION MONITORING

By :

MOKHTARI Abderrahim

In front of a jury composed of:

Mr. FERFERA	President
Mrs. HADJ HENNI	Examiner
Ms. YKHLEF Hadjer	Supervisor
Ms. DIFFALLAH Zhor	Supervisor

2020/ 2021

Abstract

Anomalous sound detection (ASD) is the task to identify whether the sound emitted from a target machine is normal or anomalous. Automatically detecting mechanical failure is an essential technology in the fifth industrial revolution, including artificial intelligence -based factory automation. Prompt detection of machine anomaly by observing its sounds may be useful for machine condition monitoring. The goal of this work is to explore two different classes of ASD systems : Unsupervised and Supervised ASD using the long-mel energies features. The unsupervised ASD approach consists of a deep AutoEncoder, whereas, the Supervised ASD system implements the outlier exposed strategy based on a deep Residual Neural Network (ResNet) . We have performed multiple experiments using a huge dataset which consists of 54, 254 sound files, and supported our analysis and discussion with numerous statistical tests to analyze and compare the two ASD systems. We have dedicated experiments for investigating the impact of varying some hyperparameters of the autoencoder architecture, like the code (or bottleneck) size Our experimental findings indicate that the deep residual network (ResNet) outperforms the autoencoder model.

Keywords: Anomaly Sound Detection, Deep Learning, Statistical Test, Feature Extraction, Machine Condition Monitoring.

Résumé

La détection d'anomalies sonores est un problème d'identification si le son d'une machine cible est normal ou il sort de l'ordinaire . La détection automatique des pannes mécaniques est une technologie qui joue un rôle important dans la cinquième révolution industrielle, comprenant l'automatisation d'usine basée sur l'intelligence artificielle. La détection rapide d'une anomalie de machine en observant ses sons peut être utile pour la surveillance de l'état d'une machine. Le but de ce travail est d' explorer deux catégories différentes de système de détection d'anomalies : détection d'anomalies en mode non supervisé et détection d'anomalies en mode supervisé en utilisant la caractéristique log-mel energies. Le système de détection d'anomalies en mode non supervisé consiste de un auto-encodeur profond tandis que le système en mode supervisé adopte la stratégie d'expositions des anomalies basée sur le réseau neuronal résiduel (ResNet). Nous avons réalisé de multiples expériences sur une base de données énorme constituée de 54,254 fichiers sonores. Nous avons appuyé notre analyse par des tests statistiques afin d'interpréter et de comparer les deux systèmes de détection d'anomalies. Nous avons dédié des expériences pour étudier l'impact de la variation de certains hyperparamètres de l'architecture de l'auto-encodeur comme la taille du code .Nos résultats expérimentaux indiquent que le réseau neuronal résiduel surpasse l'auto-encodeur.

Mots Clés: La Détection d'Anomalies Sonores, Apprentissage Approfondi, Tests Statistiques, Extraction de Caractéristiques, La Surveillance de l'Etat de Machine.

ملخص

الكشف عن الخلل في الصوت هي مهمة تحديد ما إذا كان صوت الآلة المستهدفة طبيعيًا أم خارجًا عن المألوف. يعد الكشف التلقائي عن الأعطال الميكانيكية تقنية حاسمة في الثورة الصناعية الخامسة، بما في ذلك التشغيل الآلي للمصانع القائمة على الذكاء الاصطناعي. الكشف الفوري عن عطل الآلة من خلال مراقبة أصواتها يعد مفيدًا لمراقبة حالة الآلات. الهدف من هذا العمل هو استكشاف جانبين مختلفين من نظام الكشف عن الخلل في الصوت: الكشف عن الخلل في الوضع غير الخاضع للرقابة و لكشف عن الخلل في الوضع الخاضع للرقابة باستخدام ميزة طاقات log-mel. يتكون نظام الكشف عن الخلل غير الخاضع للرقابة من شبكة عصبية عميقة تعتمد على التشفير التلقائي بينما يتبنى نظام الوضع الخاضع للرقابة استراتيجية تعريض القيم المتطرفة القائمة على الشبكة العصبية (ResNet). لقد أجرينا تجارب متعددة باستخدام قاعدة بيانات ضخمة تتكون من 54,254 ملفًا صوتيًا، ودعمنا تحليلنا ومناقشتنا مع العديد من الاختبارات الإحصائية لتحليل ومقارنة نظامي الكشف عن الخلل في الصوت. لقد خصصنا تجارب للتحقيق في تأثير اختلاف بعض المعلمات الفائقة لبنية التشفير التلقائي، مثل حجم الشفرة. تشير نتائجنا التجريبية إلى أن الشبكة العصبية ResNet تتفوق على نموذج الغير الخاضع للرقابة.

الكلمات المفتاحية : الكشف عن الخلل في الصوت، التعلم العميق، الاختبارات الإحصائية، استخلاص الميزة ، مراقبة حالة الآلات.

Acknowledgements

First and foremost, praises and thanks to the God, the Almighty, for His showers of blessings throughout my research work to complete the research successfully.

I would like to express my deep and sincere gratitude to my research supervisor, Ms. YKHLEF Hadjer for giving me the opportunity to do the research and providing invaluable guidance throughout this research. Her dynamism, vision, sincerity and motivation have deeply inspired me. She has taught me the methodology to carry out the research and to present the research works as clearly as possible. It was a great privilege and honor to work and study under her guidance.

I am extending my thanks to Ms. DIFFALLAH Zhor, for her help and guidance during our research work. Special thanks goes to the examiners for the time they spent on carefully reviewing this thesis.

Finally, I am extremely grateful to my parents for their love, prayers, caring and sacrifices for educating and preparing me for my future.

Contents

List of Figures

List of Tables

Introduction

Context and problem statement

Anomalous Sound Detection (ASD) has received a lot of attention due to its diverse and practical applications [13]. ASD has been used for surveillance [82], gun-shot detection [166], product inspection [86] and product maintenance [102]. ASD is used both as an independent measure or in addition to visual/other information. Prompt response to changes observed in equipment sounds can increase reliability and safety with expensive and dangerous machinery. The aim of machine condition monitoring is to identify whether the sound emitted from a target machine is normal or anomalous after training a system only with a set of normal sounds. Moreover, automatically detecting mechanical failure is a significant technology in the fifth industrial revolution. Therefore, having an early anomaly detection for machines by observing their sounds could be very useful to detect an abnormal performance of the machines in advance.

ASD is divided into two broad categories, supervised ASD and unsupervised ASD. Supervised ASD is comprised of tasks where anomalous sounds and their acoustic structures are defined and can then be used to train the models. This includes environment detection, gun shot detection, audio tagging etc. These models are specific to the type of anomalies being studied and may perform badly or unexpectedly in case of unexpected anomalies. Unsupervised ASD tasks are more common in situations where anomalies are not defined but there is enough information of the type of normal acoustic structure expected. An anomaly is defined as anything which is significantly outside this normal or expected structure, or an outlier. Therefore, unsupervised ASD problems are popularly dealt with outlier detection techniques. The distance between a model trained on normal sounds and the given anomalous or test sound is taken. This difference is known as anomaly score, and it determines whether the test sample is an outlier or not based on a threshold [86]. Figure 1 represents the overview of an ASD system.

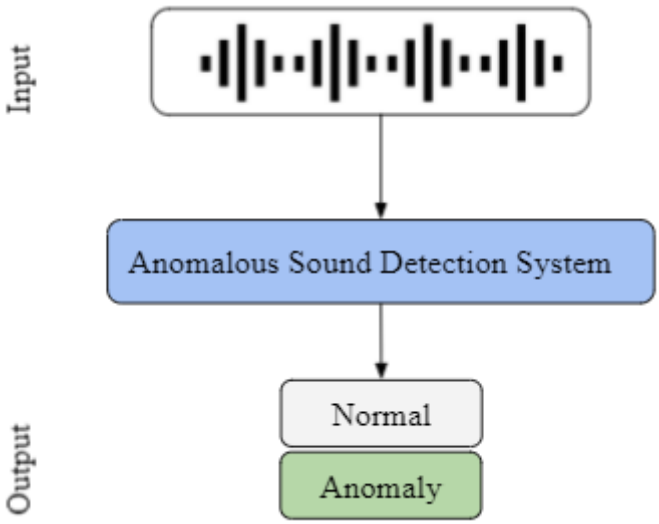

Figure 1: Overview of ASD System.

The process of Anomaly Sound Detection consists of two main stages : Feature Engineering and Machine Learning, Feature Engineering usually englobes two substeps: **Pre-processing** and **Feature Extraction**. First, the main role of sound pre-processing step is to enhance certain characteristics of the incoming audio file in order to optimize audio analysis performance in the later phases of the analysis system. Then, feature extraction is applied on the resulting preprocessed data. Here, we divide the audio signal into equal frames in order to perform feature extraction and obtain a feature vector per frame. The most common types of features used in the literature include: Mel-frequency cepstral coefficients [171], Log Mel band Energy [172] and spectral centroid. Next, the deep neural network model takes the feature vector of each frame and outputs an anomaly score of each input frame. Finally, we aggregate the score of the frames composing the sound signal to obtain the overall score.

Related Work

While various approaches on classification [93][111] and tagging of acoustic scenes [94] have been proposed in the last years, acoustic anomaly detection is still underrepresented. Due to the release of publicly available datasets [95][96] the situation is gradually improving and numerous attempts have been made by the research community [104][105]. The majority of approaches to acoustic anomaly detection relies upon deep autoencoders. For example, Marchi et al, use a bidirectional recurrent denoising AE to reconstruct auditory spectral features to detect novel events [99]. In [101] the authors compare various AE architectures, they conclude that a convolutional architecture operating on the Mel-Frequency Cepstral Coefficients is well suited for the task while a One-Class Support Vector Machine represents a strong and more

parameter efficient baseline. Kawaguchi et al, explicitly address the issue of background noise. An ensemble method of front-end modules and backend modules followed by an ensemble-based detector combines the strengths of various algorithms. Frontends consist of blind-dereverberation and anomalous sound- extraction algorithms, back-ends are AEs. The final anomaly score is computed by score-averaging [102].

Thesis structure

This thesis consists of two primary parts. The first part covers the state-of-the-art notions that are necessary for understanding the ideas developed in this thesis. **Chapter 1** is also divided into two parts, the first one gives an overview of acoustic features used to represent audio signals. Specifically, we present the different feature extraction techniques that are frequently used in literature as for the second part of this chapter we review some relevant classification concepts, providing a brief description of the supervised and unsupervised approaches, evaluation metrics and statistical tests invoked in this work. In **Chapter 2** we provide a brief description of machine learning, state-of-the-art deep learning architectures. The second half of this thesis describes the approach that we have chosen for building and evaluating our ASD systems. We provide in **Chapter 3** a detailed description of the pipeline for building our anomaly detection systems, including the dataset used, feature extraction and parameters setting.

In Chapter 4, we describe and discuss the experimental results and findings that we have obtained during our experiments through performance tables and plots. Finally, we conclude by summarizing the contributions of this thesis, the lines of limitations and future work.

Chapter 1

Generalities on Anomaly Sound Detection

1.1 Introduction

Anomaly Sound Detection (ASD) has received a lot of attention from the machine learning community in recent years [13]. Since anomalous sounds indicate symptoms of mistakes or malicious activities, their prompt detection can possibly prevent such problems. In particular, ASD has been used for various purposes including audio surveillance [81][82], animal husbandry [83][84], product inspection, and predictive maintenance [85][86]. For the last application, since anomalous sounds might indicate a fault in a piece of machinery, prompt detection of anomalies would decrease the number of defective products and/or prevent propagation of damage [13]. Digital signal processing (DSP) techniques are at the core of anomaly detection systems. Programming a computer allows for the robust control of audio signals and the creation of software for specialized tasks. Digital signal processing (DSP) is an extensive field of mathematical methods with many applications to process digital audio signals. Together, computer programming and DSP provide a framework for accomplishing many audio-related tasks [4]. In recent years, machine learning has shown tremendous capabilities in learning expressive representations of complex data, pushing the boundaries of different learning tasks. machine learning for anomaly detection aims at learning feature representations via neural networks for the sake of anomaly detection [147].

In this Chapter, we introduce a few fundamental concepts behind Anomaly Sound Detection that we require to perform our work. We begin with defining the multiple characteristics and representations of audio signals in **section 1.2-1.3**, we explain Preprocessing and feature extraction techniques in **Sections 1.4, 1.5**. Then we provide a short introduction to the relevant concepts of classification in **Section 1.6**. Then, in **Section 1.7, 1.8** we present model evaluation techniques and statistical tests. And Finally, we summarize the main concepts that we have learned.

1.2 Sound acquisition

Audible sound arises from pressure variations in the air falling on the ear drum. The human auditory system is responsive to sounds in the frequency range of 20 Hz to 20 kHz as long as the intensity lies above the frequency dependent "threshold of hearing"[1].

The sound captured by a microphone is a **time waveform** of the **air pressure** variation at the location of the microphone in the sound field [1]. Many natural phenomena produce analog signals which have a **continuous range** of values in both time and amplitude, which generally leads to an infinite number of values. Since a computer can only store and process a finite number of values, one has to convert the waveform into a discrete representation, this process is commonly referred to as **digitization**[8] i.e we convert analog signals to digital signals. Typically, digital audio signals are formed by sampling analog signals at regular intervals in time, and then quantizing the amplitudes to discrete values [3]. The conversion from analog signal to digital signal generates what is known as the representation of sound in the **time domain**, as shown in the Figure 1.1.

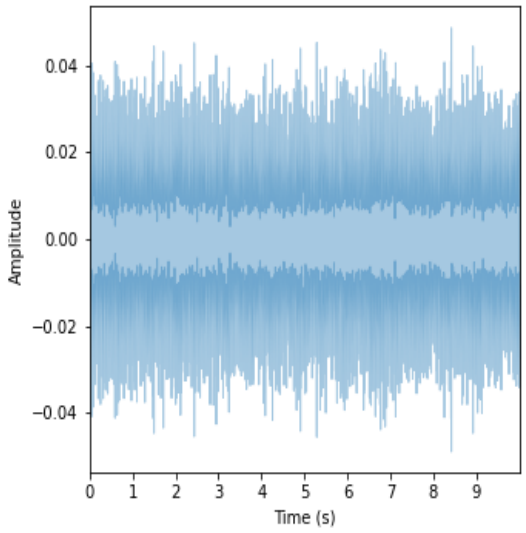

Figure 1.1: Time domain representation of a fan.

1.2.1 Sampling

Sampling a continuous-time signal implies taking snapshots of the signal at specific instances of time [2]. The samples in a digital signal occur at regular time intervals, Ts, called the **sampling period** with units of seconds per sample. The **sampling rate**, Fs = 1 /Ts, of a digital signal defines the number of samples per second and is measured in Hertz (Hz).

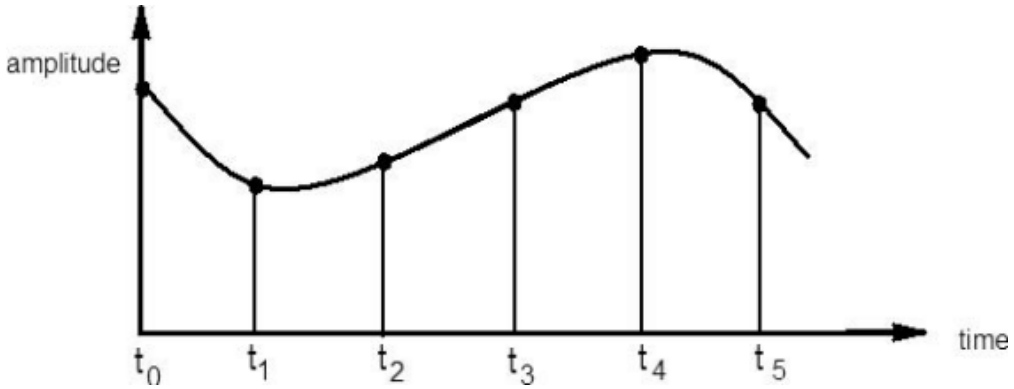

Figure 1.2: The sampling process[6].

1.2.2 Quantization

In a digital system, a fixed number of binary digits (bits) are used to represent numbers that approximate the actual values of the samples of the analog waveform. In order to make a signal suitable for treatment by numerical circuitry, it must first be represented in a numerical format, or quantized. That is, a continuous range of values is replaced by a limited set of values separated by discrete steps [5].

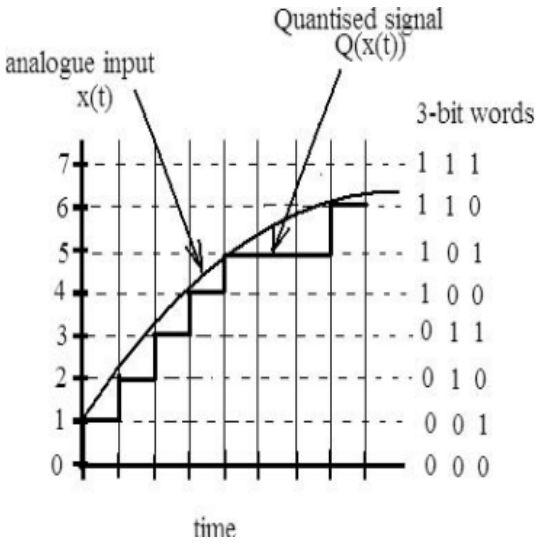

Figure 1.3: Quantization[6].

1.3 Audio signal representation

Over the years, a large amount of work has been devoted to finding appropriate representations that allow extraction of useful information from sound signals [7].The time domain representation of a sound signal, or waveform, is not easy to interpret directly [7]. Most of the time it is nearly impossible, from a waveform, to identify or even localize sound events (unless they occur

at different dynamic ranges, e.g., a loud noise in a quiet environment) and to discriminate between sound scenes. Therefore, frequency-domain representations and time-frequency domain representations have been used for years providing representations of the sound signals that are more in line with human perception [7]. The frequency-domain representation of a signal can be obtained using the **Fourier Transform**.

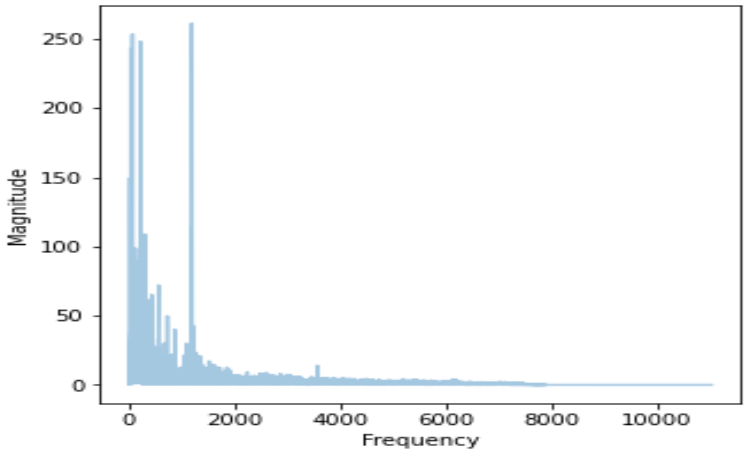

Figure 1.4: Frequency domain representation of a fan.

1.4 The Fourier Transform

The Fourier transform is a mathematical way to go between the functional representation of a signal and its Fourier representation. It is applied to turn a function of time into a function of frequency by summing up its sinusoidal or complex exponential components. It allows the passage from the **temporal representation** that shows the way the overall sound amplitude changes over time to the **frequency representation** that shows how much of the signal lies within each given frequency band over a range of frequencies. If the spectral content of the signal does not change much over time, then this works quite well, but if the signal changes over time the Fourier transform will not be able to distinguish between the different changes in the spectral content. **The short-time Fourier transform (STFT)** is an attempt to fix the lack of time resolution in the classic Fourier transform [79].

1.4.1 The Short Time Fourier Transform

The Fourier transform yields frequency information that is averaged over the entire time domain. However, the information on when these frequencies occur is hidden in the transform [8]. To recover the hidden time information, Dennis Gabor introduced back in 1946 the **short-time Fourier transform (STFT)**. Instead of considering the entire signal, the main idea of the

STFT is to consider only a small section of the signal [8]. The signal is broken into many small sequential pieces, called **frames**, and the Fourier transform is applied to each of these frames in succession. What is produced is a time-dependent representation, showing the changes in the spectrum as the signal progresses [9]. To this end, one fixes a so-called **window function**, which is a function that is nonzero for only a short period of time [8]. It gently scales the amplitude of the signal to zero at each end, reducing the discontinuity at frame boundaries[9] as shown in Figure 1.5. The original signal is then multiplied with the window function to produce a windowed signal [8]. Figure 1.5 represents a hamming window applied on a signal frame.

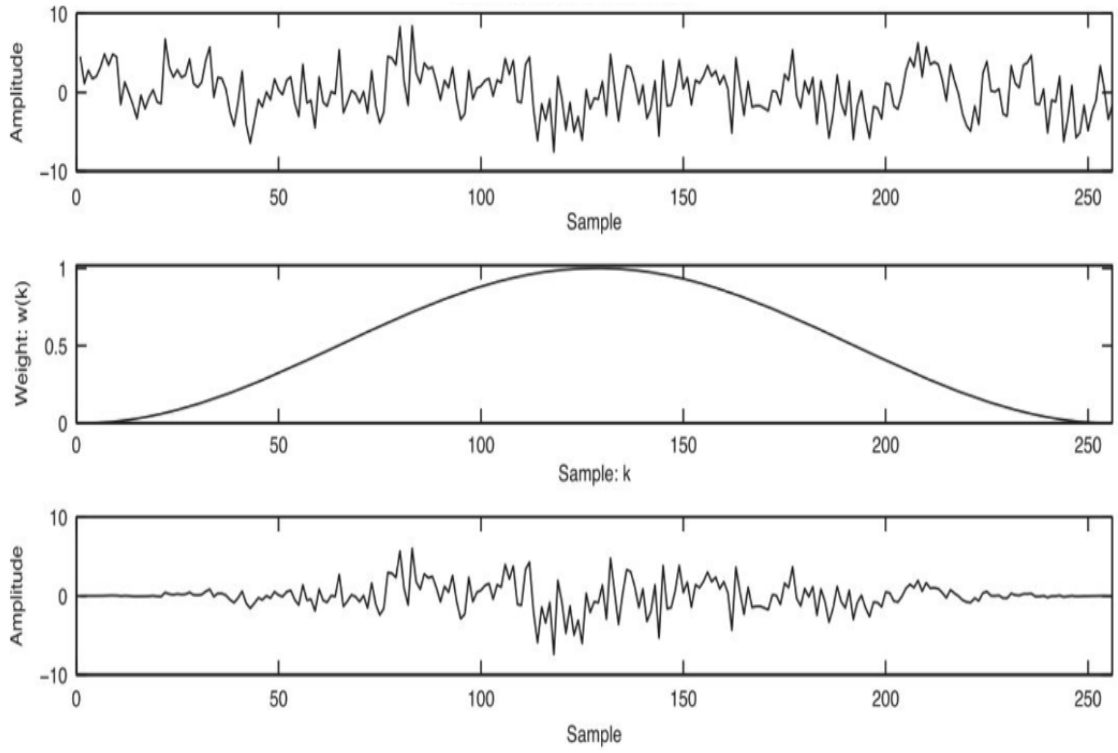

Figure 1.5: Applying the Hamming window function on a frame.

When these windowing functions are applied to a signal, it is clear that some information near the frame boundaries is lost. For this reason, a further improvement to the STFT is to overlap the frames as shown in Figure 1.6. When each part of the signal is analyzed in more than one frame, information that is lost at a frame boundary is picked up between the boundaries of the next frame [9].

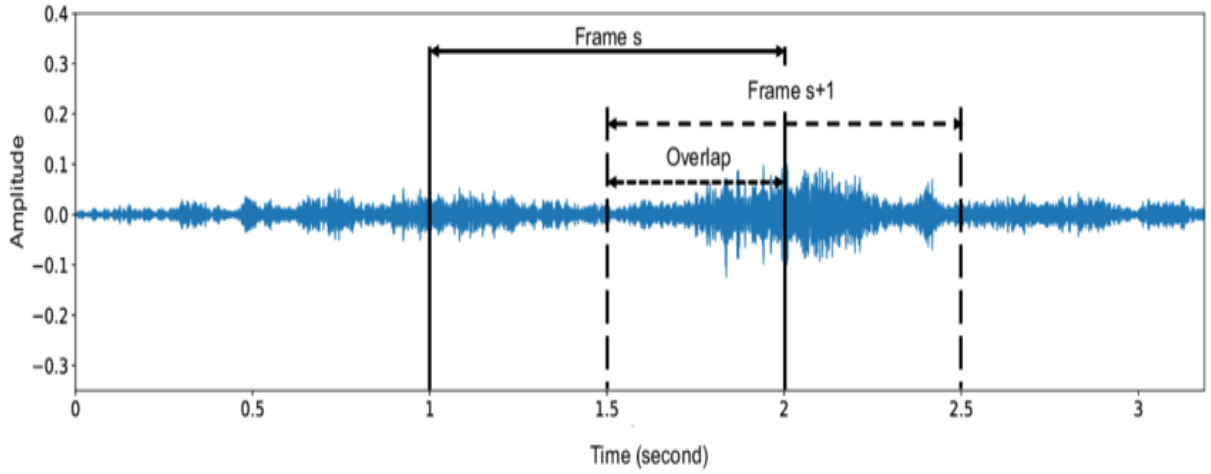

Figure 1.6: Signal Framing.

1.5 Feature Extraction

Feature extraction is one of the most significant factors in audio signal processing. It has a vital role in evaluating and characterizing audio content. Audio signals have many features, not all of which are essential for audio processing [10]. Acoustic features can be classified into 3 categories : temporal, spectral, and perceptual features [10].

1.5.1 Temporal features

These features are computed directly from raw audio signals with no preceding data. Representative instances of temporal features are zero-crossing rate, amplitude-based features, and power-based features [7]. Such features normally suggest a simple tactic to investigate audio signals. Therefore, their computational complexity is lower than that of spectral features [10]. However, it is generally necessary to combine them with spectral features for better representation [143].

1.5.2 Spectral features

Time domain representation shows the signal variation with respect to time. To analyze a signal in terms of frequency, the time-domain signal is converted into a frequency-domain signal by using Fourier transform. Frequency domain analysis is a tool of utmost importance in audio signal processing [11].

Spectrograms

A sound spectrum is a representation of a sound, usually a short sample of a sound in terms of the amount of vibration at each individual frequency, it is usually presented as a graph of either power or pressure as a function of frequency [152]. The spectrogram is the time-varying spectrum of a signal. To generate a spectrogram, the signal is broken into frames. As for the STFT, the spectrum is calculated on each frame and these spectra are displayed as a time-varying spectrum. The result is a measure of the way the frequency content of the signal changes over time [79]. It can be visualized by means of a two-dimensional image, where the horizontal axis represents time and the vertical axis represents frequency [80]. The Figure (1.7) shows the spectrogram feature of an audio recording of a pump machine.

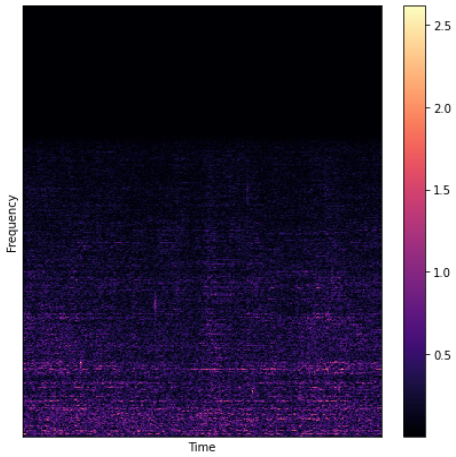

Figure 1.7: Spectrogram of a recording of a pump.

Log-Mel Spectrograms

The Mel spectrogram involves applying the Mel filter banks to a spectrogram, converting the frequency (Hz) scale into Mel scale (Figure 1.8) [7]. This is a non-linear transformation that is motivated by the human's auditory system, where frequencies are not perceived in a linear scale. It is more noticeable for a human to distinguish the sound in lower frequencies, ranging from 500 and 1000 Hz, than in higher frequencies ranging from 7500 and 8000 Hz. The range of frequencies that can be heard by humans is generally from 20 to 20,000 Hz [152], and the sensitivity is gradually lost as the frequency increases. Thus, applying the Mel filter bank is beneficial as it highlights the variations in lower frequencies and gives more discriminating and informative audio representation. The Mel scale is transformed from the frequency scale by

$$m(f) = 2595 \log_{10}(1 + \frac{f}{700}) \tag{1.1}$$

the Mel scale can be transformed back to the frequency by

$$f(m) = 700(10^{m/2595} - 1) \tag{1.2}$$

Mel filter bank is a set of triangular filters, each of the filter is centered and reaches the peak at its center frequency and decreases to zero linearly at the two neighboring filters' center frequency. Those center frequencies of triangular filters are spaced non-linearly as they are obtained by the linearly placed Mel scale and calculated by (1.2).
The linearly placed Mel scale is decided by the frequency limits and the number of Mel filters defined [87], the filter bank energy is obtained after mel filtering. Finally, the logarithmic conversion of the mel energy is calculated and then the Log Mel Spectrum is generated from the filter bank.

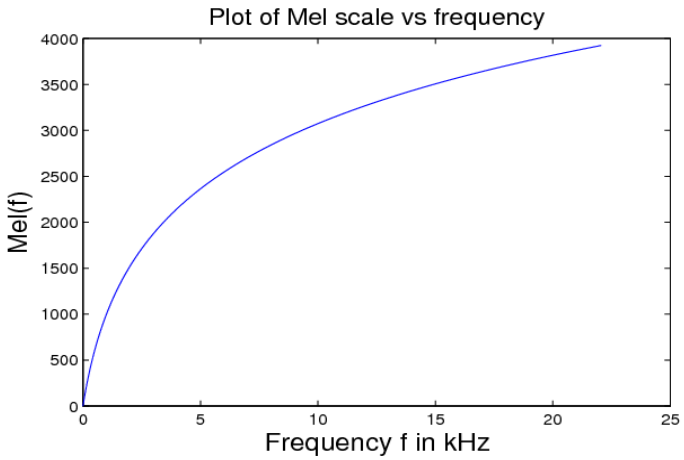

Figure 1.8: The Mel Scale.

1.5.3 Perceptual Features

Perceptual frequency features indicate information with semantic meaning in the context of human listeners. Therefore, they are organized according to semantically meaningful aspects of sounds including pitch, fundamental frequency, loudness, intensity and sharpness.

- **Loudness** (measured in sones) is the subjective impression of the intensity of a sound in such a way that a doubling in sones corresponds to a doubling of loudness [7].
- **Pitch** is a perceptual property of sounds that allows their ordering on a frequency-related scale. More commonly, pitch is the quality that makes it possible to judge sounds as «higher» and «lower» in the sense associated with sound recording.
- **Sharpness** can be interpreted as a spectral centroid based on psychoacoustic principle. It is commonly estimated as a weighted centroid of specific loudness [88].

1.6 Classification

In machine learning, classification is referred to as the problem of identifying whether an object belongs to a particular category based on a previously learned model. This model is learned statistically based on a set of training data whose categorization is predefined [25]. The concept of classification has been traditionally treated in a broad sense, very often including supervised, unsupervised and semi-supervised learning problems [26]. In **supervised learning**, available data comprises feature vectors together with target values. The data is analysed in order to tune parameters of a model, which can be used to predict the target values for novel data that was not contained in the training set [148]. Unlike supervised learning, **unsupervised learning** is concerned with learning patterns in the input data without any output values available for training [149]. given a data set D consisting of N unlabelled observations $x_n \in R^D$. These are assumed to be drawn from an unknown true distribution as

$$D = [x_n]_{n=1}^N \sim P(x) \tag{1.3}$$

The goal is to learn some useful properties of the distribution $P(x)$, where the properties of interest depend on the specific application [156]. Semi-supervised learning is more recent when compared with the supervised and unsupervised learning [22], as the name suggests; semi-supervised learning is somewhere between unsupervised and supervised learning [150]. The dataset provided to the semi-supervised learning model is partially labeled [151], and is also provided with some supervision information [18]. The main objective of semi-supervised learning is to overcome the drawbacks of both supervised and unsupervised learning [22].

Although classification is usually understood as supervised learning, semi-supervised and unsupervised scenarios can be considered as a way of obtaining better classifiers. In the semi-supervised setting, both labeled and unlabeled examples are used during the classifier's construction to complement the information obtained by considering only labeled samples [52]. Unsupervised learning is sometimes applied as a way to obtain labels for training classifiers or to derive some parameters of the classification models [53][54]. The general aim of supervised classification algorithms is to separate the classes of the problem (with a margin as wide as possible) using only training data. If the output variable has two possible values, the problem is referred to as binary classification. On the other hand, if there are more than two classes, the problem is named multiclass or multinomial classification. A classification problem can be formally defined as the task of estimating the label y of K-dimensional input vector x where $x \in X \subseteq R^K$ (note that, for most ML algorithms, input variables have to be real-valued) and

$y \in Y = \{C1, C2, ..., CQ\}$. In Supervised anomaly sound detection requires the entire dataset to be labeled "normal" or "abnormal" and this technique is basically a type of binary classification task. Semi-supervised anomaly sound detection requires only data considered "normal" to be labeled, in this technique, the model will learn what "normal" data are like. Unsupervised anomaly sound detection involves unlabeled data. In this technique, the model will learn which data is "normal" and "abnormal" [153].

1.7 Evaluation

To measure the performance of anomaly detection techniques, the "accuracy" metric was often used. Nevertheless, in the event of imbalanced datasets, the reported accuracy will not offer an accurate representation of the technique's efficiency. To measure the performance more accurately, metrics such as True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN), AUC, pAUC are used [154][155].

1.7.1 Confusion Matrix

Confusion matrix is a measure used to evaluate a classifier's performance considering a pre-known set of labeled data [145]. It contains information about actual and predicted classifications done by a classification system. Performance of such systems is commonly evaluated using the data in the matrix[146]. The row dimension contains the actual values, whereas the column dimension consists of the predicted class. Figure 1.9 provides a representation of the confusion matrix.

		Actual Value (Obtained by experiment)	
		Positives	Negatives
Predicted Value (Predicted by the test)	Positives	**True Positive** (Correct)	**False Positive** (Incorrect)
	Negatives	**False Negative** (Incorrect)	**True Negative** (Correct)

Figure 1.9: Confusion matrix [145].

- **True Positives (TP)** are the cases where the actual class of the data point is positive and the predicted also positive.
- **True Negatives (TN)** are the cases where the actual class of the data point is negative, while the predicted is negative.
- **False Positive (FP)** are the cases where the actual class of the data point is negative, while the predicted is positive.
- **False Negative (FN)** are the cases where the actual class of the data point is positive, while the predicted is negative.

The metrics that can be computed from the Confusion matrix includes precision, recall, F1-score, the mean average precision and the classification accuracy [145].

1.7.2 Area under the ROC Curve (AUC / pAUC)

A receiver operating characteristics (ROC) graph is a technique for visualizing, organizing and selecting classifiers based on their performance [73]. Estimating ROC graphs is a general approach to tuning and evaluating binary classifiers in machine learning. The ROC curve for a binary scoring-based classifier. In our case, a scoring-based anomaly detector is a two-dimensional curve in which the **Sensitivity**, also known as the **True Positives Rate (TPR)**, is plotted on the Y axis and the **false alarm rate** (1 – Specificity), also known as **False Positives Rate (FPR)**, is plotted on the X axis. Thus, a ROC curve visualises the relative tradeoff between TPR [0, 1] and FPR [0, 1] for each possible anomaly threshold [73]. Figure 1.10 illustrates an example of a Roc curve.

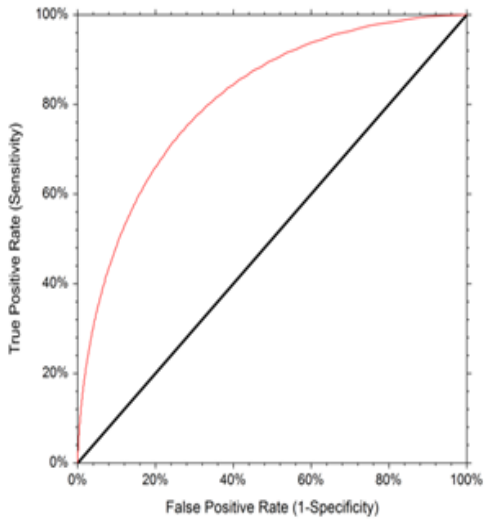

Figure 1.10: Roc Curve example.

Several ROC curve summary measures have been proposed in the literature, such as the **area under the curve (AUC)** [74] given by :

$$AUC = \int_0^1 ROC(u)du \tag{1.4}$$

AUC is a standard measure for the classification performance of binary scoring based classifiers [73]. The larger the AUC, the better the overall classification performance over all possible thresholds. AUC corresponds to a portion of the area of the unit square and, thus, its value will always be within [0, 1]. Intuitively, the AUC of an anomaly detector is equivalent to the probability that the anomaly detector will rank a random example from the abnormal class higher than a random example from the normal class [73]. Nevertheless, the **partial AUC (pAUC)** [75] has been proposed as an alternative to AUC in binary classification applications when only a subrange of the ROC curve is of practical interest, e.g., FPR [0, 0.01]. In anomaly sound detection (ASD) the reason for the additional use of the pAUC is based on practical requirements. If an ASD system frequently gives false alerts, it cannot be trusted. Therefore, it is especially important to increase the true positive rate (TPR) under low FPR conditions [144].

1.8 Statistical Tests

Recently, the machine learning community has become increasingly aware of the need for statistical validation of the published results [76]. Various researchers adopt different statistical and common-sense techniques to decide whether the differences between the algorithms are real or random. In this section we shall examine the statistical tests used in our thesis.

1.8.1 Sign Test

A popular way to compare the overall performances of classifiers is to count the number of data sets on which an algorithm is the overall winner. When multiple algorithms are compared, pairwise comparisons are sometimes organized in a matrix [65]. Some authors also use these counts in inferential statistics, with a form of binomial test that is known as the sign test [77][78]. If the two algorithms compared are, as assumed under the null-hypothesis, equivalent, each should win on approximately N/2 out of N data sets. The number of wins is distributed according to the binomial distribution; the critical number of wins can be found in [65]. For a greater number of data sets, the number of wins is under the null-hypothesis distributed according to $N(N/2, \sqrt{N}/2$ which allows for the use of z-test: if the number of wins is at least

$N/2+1.96\sqrt{N}/2$ (or, for a quick rule of a thumb, $N/2+\sqrt{N}$) the algorithm is significantly better with $p < 0.05$. Since tied matches support the null-hypothesis they should not be discounted but split evenly between the two classifiers [65].

1.8.2 Wilcoxon Signed-Ranks Test

The Wilcoxon signed-ranks test (Wilcoxon, 1945) is a non-parametric alternative to the paired t-test, which ranks the differences in performances of two classifiers for each data set, ignoring the signs, and compares the ranks for the positive and the negative differences. let d_i again be the difference between the performance scores of the two classifiers on i-th out of N data sets. The differences are ranked according to their absolute values; average ranks are assigned in case of ties. Let R^+ be the sum of ranks for the data sets on which the second algorithm outperformed the first, and R^- the sum of ranks for the opposite. Ranks of $d_i = 0$ are split evenly among the sums [65], Their formal definitions are given by:

$$R^+ = \sum_{d_i>0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i) \tag{1.5}$$

$$R^- = \sum_{d_i<0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i) \tag{1.6}$$

Finally, the statistics T is computed as $T = min(R^+, R^-)$. For small N the critical value for T can can be found in any textbook on general statistics [65], whereas for larger N the statistics:

$$z = \frac{T - \frac{1}{4}N(N+1)}{\sqrt{\frac{1}{24}N(N+1)(2N+1)}} \tag{1.7}$$

1.9 Conclusion

In this chapter, we have reviewed the concept of sound and audio signals processing, we also presented several types of features which play a huge part in our work as they are required to be an input for the learning stage. We have reviewed some important concepts of classification in general. Many classification paradigms have been applied for developing anomaly sound detection systems. Most importantly, deep learning-based systems have attracted a wide spread attention from the research community due to their effectiveness. Therefore, in the next chapter, we will give an overview of some deep learning notions, including several well-known deep architectures such as Residual neural networks and Autoencoders.

Chapter 2

Deep Learning for Machine Condition Monitoring

2.1 Introduction

In the previous chapter we discussed the concept of audio signals and the process followed to extract the features that are required for the machine learning task. and we reviewed some concepts of machine learning techniques for building an anomaly sound detection system. In addition to machine learning , deep learning algorithms have become increasingly popular at detecting anomalies , it completely surpasses traditional methods of machine learning which makes it a crucial part in anomaly sound detection systems. In this chapter we introduce the concept of Machine learning and Deep learning, then we focus on the supervised and the unsupervised techniques that we used to build our ASD system.

2.2 Machine Learning Basics

Machine learning techniques have been widely applied in a variety of areas [159]. With machine learning techniques, computers are endowed with the capability of acting without being explicitly programmed, constructing algorithms that can learn from data, and making data-driven decisions or predictions. During the past decades, machine learning has brought enormous influence on our daily life with examples including efficient web search, self-driving systems, computer vision, and optical character recognition. In addition, by adopting machine learning methods, the human-level artificial intelligence (AI) has been improved as well, see [30][31][32] for more discussions.

2.2.1 Types of problems and tasks

Anomaly sound detection (ASD) refers to the problem of finding patterns in data that do not conform to expected behavior. These non-conforming patterns are often referred to as anomalies or outliers. Machine learning techniques for anomaly sound detection are mostly classified into supervised and unsupervised. the ADS system can be trained using a supervised method that is used in various Sound environment tasks such as audio scene classification [141], sound event detection [14][15], and audio tagging [16]. On the other hand, unsupervised ASD [17][19] is the task of detecting "unknown" anomalous sounds that have not been observed. many machine learning techniques were used to detect anomalies in audio. However, two machine learning techniques stand out: the Autoencoder (AE) and the Convolutional Neural Network (CNN) [13]. Another categorization of machine learning tasks arises when one considers the desired output of a machine learned system, this categorization gathers two broad tasks in machine learning which classification and regression [23]. In regression output variable takes continuous values while in classification output variable takes class labels [24].

2.3 Deep Learning

Machine learning techniques have been widely applied in a variety of areas such as pattern recognition, natural language processing and computational learning [159]. Nevertheless, when it comes to the human information processing mechanisms (e.g. speech and vision), the performance of traditional machine learning techniques are far from satisfactory [159]. Inspired by deep hierarchical structures of human speech perception and production systems, the concept of deep learning algorithms was introduced in the late 20th century [159].

Over the last years, deep learning has emerged as a popular set of machine learning methods based on learning data representations. It has been shown that deep learning algorithms are able to beat state-of-the-art approaches in traditional machine learning problems [27][28].Deep learning is a set of learning methods attempting to model data with complex architectures combining different non-linear transformations. The elementary bricks of deep learning are the neural networks that are combined to form the deep neural networks.These techniques have enabled significant progress in the fields of sound and image processing, including facial recognition, speech recognition, computer vision, automated language processing, text classification (for example spam recognition) [160]. Deep learning is a term that combines methods based on the use of deep **artificial neural networks** inspired by the structure and function of the human brain.

2.3.1 Deep Neural Networks

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system [29]. Neural networks process information in a similar way the human brain does. The network is composed of a large number of highly interconnected processing elements (neurons) working in parallel to solve a specific problem [29].

ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well [29]. Basic ANN contains one or several layers of neurons. Layer represents a set of neurons that are not connected between each other in most cases, and stay between other layers. Some neuron layers are defined as input and output layers which depend on their position [34]. An artificial neural network with more than one hidden layer is considered to be a deep neural network, and a network with less than two hidden layers is considered a shallow neural network [161]. The advantage of deep neural networks is evident when processing large amounts of data. The quality of traditional algorithms, reaching a certain limit, no longer increases with the amount of available data. At the same time, deep neural networks can extract the features that provide the solution to the problem, so that the more data, the more subtle dependencies can be used by the neural network to improve the quality of the solution [161].

2.3.1.1 Neuron

Artificial neurons, which try to mimic the behavior of the human brain, are the fundamental component for building ANNs. The basic computational element (neuron) is called a node (or unit) which receives inputs from external sources and has some internal parameters (including weights and biases that are learned during training) which produce outputs [33]. Figure 2.1 illustrates the main components of an artificial neuron.

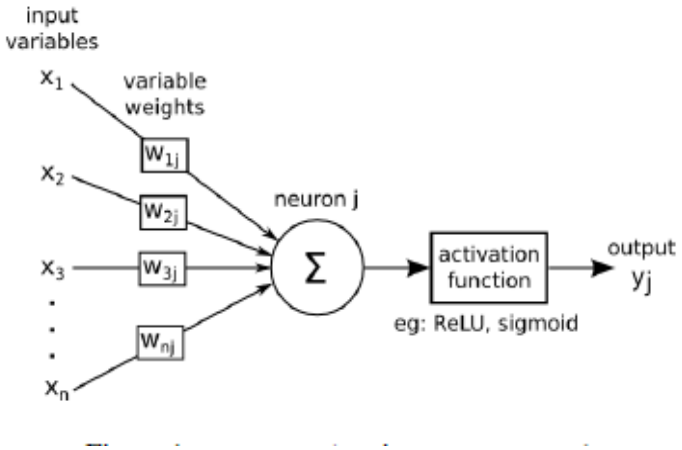

Figure 2.1: Representation of an artificial neuron [162].

2.3.1.2 Multi-layer Perceptron

The multilayer perceptron is the most known and most frequently used type of neural network. On most occasions, the signals are transmitted within the network in one direction: from input to output. There is no loop, the output of each neuron does not affect the neuron itself.

A node, also called a neuron or Perceptron, is a computational unit that has one or more weighted input connections, a transfer function that combines the inputs in some way, and an output connection.Nodes are then organized into layers to comprise a network. A single-layer artificial neural network, also called a single-layer, has a single layer of nodes, as its name suggests. Each node in the single layer connects directly to an input variable and contributes to an output variable.

Single-layer networks have just one layer of active units. Inputs connect directly to the outputs through a single layer of weights. The outputs do not interact, so a network with N outputs can be treated as N separate single-output networks [63]. A single-layer network can be extended to a multiple-layer network, referred to as a Multilayer Perceptron. A Multilayer Perceptron, or MLP for short, is an artificial neural network with more than a single layer.

The standard multilayer perceptron (MLP) is a cascade of single-layer perceptrons. There is a layer of input nodes, a layer of output nodes, and one or more intermediate layers. The interior layers are sometimes called “hidden layers” because they are not directly observable from the systems inputs and outputs [63]. the architecture of the MLP is shown in the figure below.

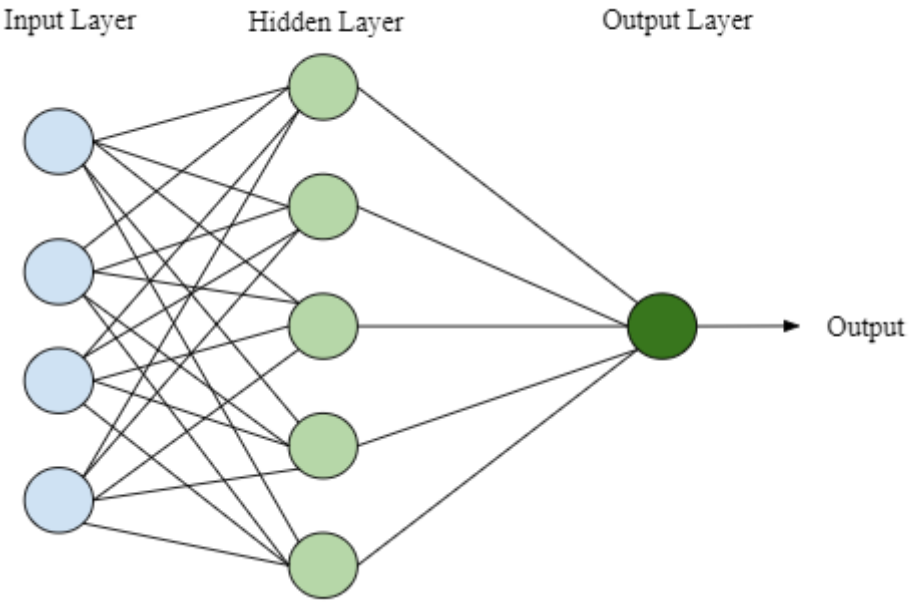

Figure 2.2: MLP network architecture [64].

2.3.1.3 Activation Functions

The primary neural networks decision-making units are activation functions. Moreover, they evaluate the output of networks neural nodes; thus, they are essential for the performance of the whole network. Hence, it is critical to choose the most appropriate activation function in neural networks calculation [163]. some commonly used functions are :

- **Sigmoid :** Sigmoid is a smooth monotonic non-linear function, It has gained popularity in Neural Networks as an activation function due to its ability to increase low signals and to not get over saturated from high signals. The function can input signals from $-\infty$ to $+\infty$ and outputs signal from 0 to 1, that makes it the best normalising function, It can be defined as follows.

$$(x) = \frac{1}{1 + e^{x}} \tag{2.1}$$

- **Rectified Linear Unit (Relu):** The Rectified Linear Unit has become very popular in the last few years. The activation is simply thresholded at zero. It was found to greatly accelerate the convergence of stochastic gradient descent compared to the sigmoid function [35]. It can be defined as follows.

$$ReLU = max(0, x) \tag{2.2}$$

- **SoftMax :** The softmax function of z, is a generalization of the sigmoid function that represents a probability distribution over a discrete variable with n possible values. Softmax functions are often used as the output units of a classifier. Formally, the softmax function is given by

$$\frac{e^{x_j}}{\sum m^{e^{x_m}}} \tag{2.3}$$

- **Tanh :** The tanh non-linearity is shown on the Figure 2.3. It squashes a real-valued number to the range [-1, 1]. Like the sigmoid neuron, its activations saturate, but unlike the sigmoid neuron its output is zero-centred [35].

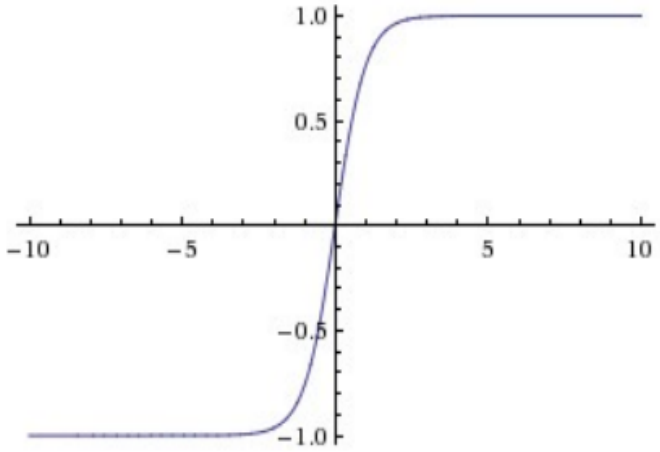

Figure 2.3: Tanh graphic.

Therefore, in practice the tanh non-linearity is always preferred to the sigmoid nonlinearity. Also note that the tanh neuron is simply a scaled sigmoid neuron, in particular the following holds:

$$F(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{2.4}$$

2.3.1.4 Training Algorithms

The output of a neural network is calculated from its input and its parameters such as weights and biases. In order to get the desired target outputs as the network output for a given input, the network is trained using a certain **optimization algorithm**. Since the input cannot be updated by the network, the network training involves the optimization of the network parameters [36].

Gradient Descent (GD)

The gradient descent approach is a first-order optimization algorithm which is used for finding the local minima of an objective function. This has been used for training ANNs in the last couple of decades successfully [42][43]. GD has three variants that differ based on the amount of data utilized to compute the gradient of the hypothesis function.

- **Stochastic Gradient Descent**,or SGD for short, is an optimization algorithm used to train machine learning algorithms, most notably artificial neural networks used in deep learning. The job of the algorithm is to find a set of internal model parameters that perform well against some performance measure such as logarithmic loss or mean squared error. The optimization algorithm is called "gradient descent", where "gradient" refers to the calculation of an error gradient or slope of error and "descent" refers to the moving down along that slope towards some minimum level of error.

- **Batch Gradient Descent** is a variation of the gradient descent algorithm that calculates the error for each example in the training dataset, but only updates the model after all training examples have been evaluated.

- **Mini-batch gradient descent** is a variation of the gradient descent algorithm that splits the training dataset into small batches that are used to calculate model error and update model coefficients.

Back propagation algorithm

Backpropagation is the essence of neural network training. It is the method of fine-tuning the weights of a neural network based on the error rate obtained in the previous epoch (i.e., iteration). Proper tuning of the weights allows to reduce error rates and make the model reliable by increasing its generalization. In other words, backpropagation compares the output calculated by the system to the given target values. calculates the difference between the two, and adjusts the weights between its units to improve its accuracy. It is a standard method of training artificial neural networks. This method helps calculate the gradient of a loss function

for a single weight by the chain rule. It efficiently computes one layer at a time, unlike a native direct computation.

2.3.1.5 Overfitting and regularizations

Deep neural networks with a large number of parameters are very powerful machine learning systems. However, overfitting is a serious problem in such networks [37]. The network is overfitted when it fits in a high extent to the training data set and therefore loses its ability to predict on new data [39]. Large networks are also slow to use, making it difficult to deal with overfitting by combining the predictions of many different large neural networks at test time [37]. There are several techniques proposed to address this issue.

- **Dropout :** It is a relatively new algorithm for training neural networks which relies on stochastically "dropping out" neurons during training [38]. With the dropout algorithm , the activations of the hidden units are dropped out with a certain probability and therefore these units do not have any effect on the output of the network [36]. This significantly reduces overfitting and gives major improvements over other regularization methods [37].

- **Early stopping :** It was invented in the early days of neural networks. It has a straightforward approach; the training of the network is being stopped before it overfits to the training data. Early stopping can be regarded as straightforward in the aspect of understanding the basics; the training should be stopped when generalization decreases. Exactly when this occurs could be clear in some cases but less so in others [39].

- **Batch Normalization :** One of the most recently proposed regularization methods is called batch normalization. It assumes normalization of each layer of neurons to have zero mean and unit variance over each training batch. It allows the use of much higher learning rates and saturating nonlinearities (such as sigmoid functions) without a risk of divergence or being stuck in the saturated regime [40].

2.3.1.6 Hyper-parameters of Neural Network

Hyper-parameters are the variables which determines the network structure(and the variables which determine how the network is trained, The hyper-parameters employed in this study are as follows:

- **The learning rate :** It is an important component for training deep neural network(DNN). The learning rate h is the step size considered during training which makes the training process faster. However, selecting the value of the learning rate is sensitive.

For example: If the value for is large, the network may start diverging instead of converging. On the other hand, if the value for h is small, it will take more time for the network to converge. In addition, it may easily get stuck in local minimal. The typical solution to this problem is to reduce the learning rate during training [41].

- **Batch size :** It is a hyper-parameter that defines the number of samples to work through before updating the internal model parameters.
- **Number of epochs :** It is a hyper-parameter that defines the number times that the learning algorithm will work through the entire training dataset. One epoch means that each sample in the training dataset has had an opportunity to update the internal model parameters. An epoch consists of one or more batches.
- **Number of hidden layers:** It determines the depth of the network. The higher the value is, the deeper the network.
- **Number of units in each layer:** hey can be different for each layer. These values determine the number of weights in total.

2.4 Unsupervised anomaly detection

Unsupervised anomaly detection methods detect anomalies without using anomalous data, these methods evolved to cater to situations where anomalous data is either extremely scarce or it has no expected structure [120]. Often clustering, dimensionality reduction, and generative techniques are considered as unsupervised learning approaches. There are several members of the deep learning family that are good at clustering and non-linear dimensionality reduction, including Auto-Encoders (AE) [33]. Any test signal that deviates from this developed model or structure is regarded as anomalous or abnormal. An anomalous "score" is calculated as the distance between the trained model and the test signal. This score is compared to a threshold and a binary decision of "normal or anomalous" is given. Figure (2.4) shows the basic flow of this process.

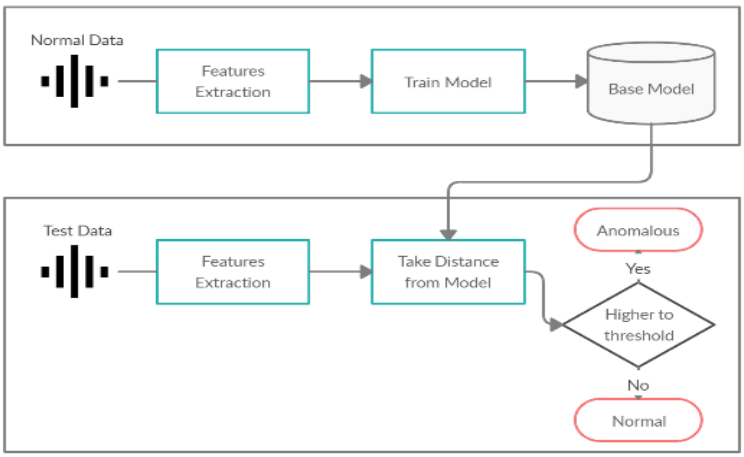

Figure 2.4: Flow of Unsupervised ASD.

2.4.1 Auto-Encoder

An autoencoder is an unsupervised neural network which is trained to reconstruct a given input from its latent representation [68]. They are simple learning circuits which aim to transform inputs into outputs with the least possible amount of distortion. These networks have a distinctive hourglass shape, with a first layer that has the same size as the last layer, but fewer neurons in their hidden layers (Figure 2.5). While conceptually simple, they play an important role in machine learning. Autoencoders were first introduced in the 1980s by Hinton and the PDP group to address the problem of "backpropagation without a teacher", by using the input data as the teacher [55].

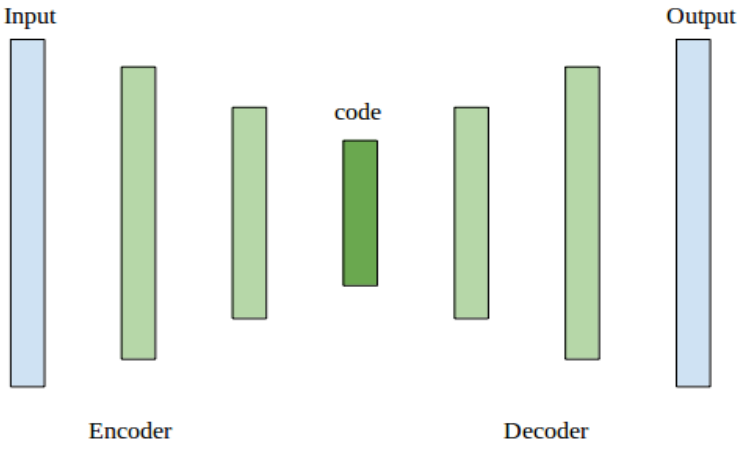

Figure 2.5: The Autoencoder architecture [69].

The aim of AutoEncoders is to learn an input function to reconstruct the input to an output of fewer dimensions. It approximates the identity function to get the outcome of a neural network similar to the input. In other words, it tries to copy the input to its output. Mathe-

matically, if x is the input (also called an encoder), x' is the network (also called a decoder). The architecture of autoencoders reduces dimensionality using non-linear optimization [57]. As represented by Eq. (2.5), the encoder network generates the feature representation by mapping the given input network traffic x to hidden layer using an activation function f parameterized by W as weights and b as the bias [70].

$$h = f(Wx + b) \tag{2.5}$$

Similarly, the decoder network reconstructs the original input network traffic from the generated feature representation using the activation function g parameterized by W and b as given below

$$Z = g(W'h + b') \tag{2.6}$$

To train an autoencoder, several parameters must be pre-set. These are parameters that determine the architecture of an autoencoder such as :

- **Code size:** It is a number of units in the middle layer of an autoencoder or the last layer of the encoder. Autoencoders who have a code size smaller than the input size are called **Undercomplete autoencoders**. These Autoencoders are designed to capture useful features in the data and reduce its dimensionality by representing the whole population with captured silent features. In this case, the network is encouraged to learn some sort of compression.

- **Number of layers :** Depending on the number of hidden layers, autoencoders can be divided into deep and shallow. A shallow autoencoder has just one hidden layer. and deep autoencoder has two or more hidden layers.

- **Optimizer :** are the algorithms that try to find optimal values of mathematical functions used in training a neural network. The gradient descent algorithm is one of the optimization algorithms used for autoencoders.It was considered in section 2.3.1.3.

The main purpose of this neural network is for the dimensionality reduction of the dataset provided [56]. The existing auto-encoder can be divided into the following types : Sparse AutoEncoder, Denoising AutoEncoder, Variational Autoencoder [58].

2.4.2 Loss function

The loss function evaluates the effectiveness of training neural networks. It returns a score that indicates how well a network performs. In the case of autoencoder, it measures how good the

reconstruction is, most commonly used loss functions are the mean squared error and the cross entropy loss functions.

2.4.2.1 Mean Squared Error loss function

Typically used for real-valued inputs, the function L represents the sum of squared Euclidean distances between the input vector x and reconstructed vector x', as shown by the equation

$$L(x, g(f(x))) = \frac{1}{n}\sum_{i=1}^{n}(x'_i - x_i)^2 \tag{2.7}$$

where $g(h)$ is the decoder output, $h = f(x)$ is the encoder output or code. When the input is real values $]-\infty;+\infty[$ it is recommended to use a linear activation function in the last reconstruction layer [164].

2.4.2.2 Cross Entropy loss function

Typically used for binary inputs, the loss function represents the sum of Bernoulli cross-entropies between two distributions [164]:

$$L(x, g(f(x))) = -1\sum_{i=1}^{n}(x_i log(x'_i) + (1 - x_i)log(1 - x'_i)) \tag{2.8}$$

2.5 Supervised anomaly detection

Unsupervised anomalous sound detection is concerned with identifying sounds that deviate from what is defined as “normal”, without explicitly specifying the types of anomalies.
A significant obstacle is the diversity and rareness of outliers, which typically prevent from collecting a representative set of anomalous sounds. As a consequence, most anomaly detection methods use unsupervised rather than supervised machine learning methods.
Nevertheless, anomalous sound detection can be effectively framed as a supervised classification problem if the set of anomalous samples is carefully substituted with outliers [66]. for instance , in case of machine monitoring Outliers are all other possible sounds in the audio domain excluding the normal data, which are all possible sounds emitted from a machine in a normal operation state; and the abnormal data, which comprises all sounds emitted from a machine in a non-normal state [67].

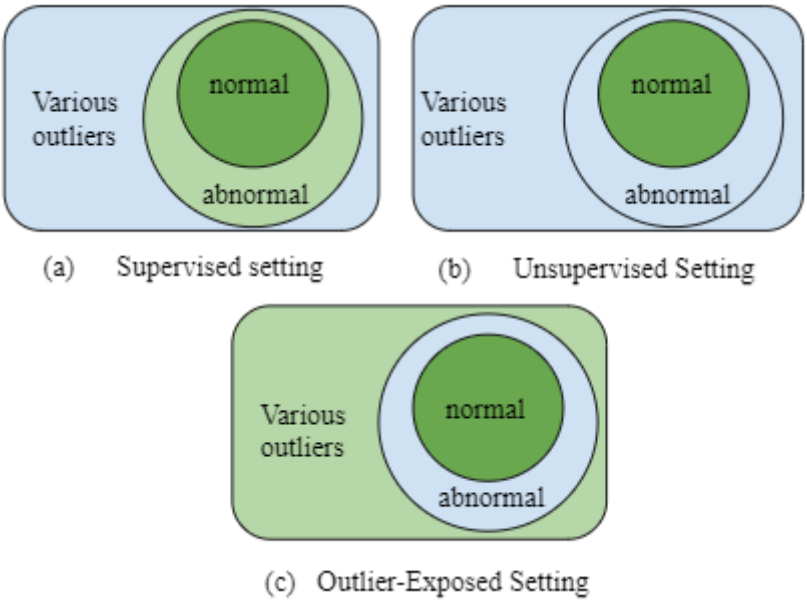

Figure 2.6: Comparison between supervised, unsupervised, and outlier-exposed ASD [67].

In Figure 2.6 dark green area represents the negative, light green the positive class. Samples in the light blue area are not available/ used.

If both normal and abnormal samples are available for training, anomaly detection can be considered a classification task, where normal samples belong to the negative class, and abnormal samples belong to the positive class (Figure 2.6.a). However, the lack of anomalous samples generally prevents from modeling anomaly detection in this supervised framework. To be able to still treat anomaly detection as a classification task, the definition of the positive class extends to include various other samples, i.e., samples that are neither normal nor abnormal but still in the same domain (outliers for short). If the anomaly detection problem is framed this way, the classifier has to distinguish normal samples from any other possible sample in the same domain [67].

Under the Outlier-exposed anomaly detection settings, many supervised learning techniques can be used. For instance, Qiuqiang Kong et al, used convolutional neural networks (CNN) in [116] due to the great performance achieved by CNNs in image processing. In this project, we have chosen a deep residual neural network used by koutini et al [141] which is a variant of the convolutional network, which has been successfully adopted for various audio-related classification tasks [165].

2.5.1 Convolutional neural networks

This network structure was first proposed by Fukushima in 1988 [44].However, It was not widely used due to limits of computation hardware for training the network. In the 1990s, LeCun et al [45] applied a gradient-based learning algorithm to CNNs and obtained successful results for the handwritten digit classification problem. After that, researchers further improved CNNs and reported state-of-the-art results in many tasks. Convolutional Neural Networks, like their fully connected counterparts, consist of a series of layered transformations of input data mediated by computational nodes. However, they have more diverse architecture and are more efficient at encoding relationships between input and output (in terms of number of parameters)[46], The **max pooling** layer of CNNs is effective in absorbing shape variations. Moreover, composed of sparse connections with tied weights, CNNs have significantly fewer parameters than a fully connected network of similar size. Most of all, CNNs are trained with the gradient-based learning algorithm and suffer less from the vanishing gradient problem. this problem occurs when long term components go exponentially fast to norm 0, making it impossible for the model to learn correlation between temporally distant events during the training [90]. Given that the gradient-based algorithm trains the whole network to minimize an error criterion directly, CNNs can produce highly optimized weights [33]. A CNN is made up primarily of 3 kinds of layers: **Convolutional layers**, **Pooling layers**, and **Fully Connected layers** [47]. the sample architecture of the convolutional neural networks is shown in the figure below.

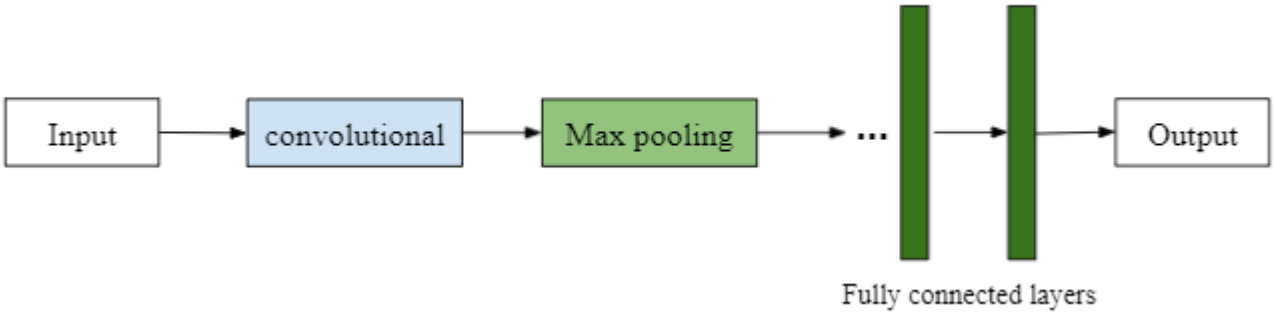

Figure 2.7: Convolutional neural network architecture [47].

The Convolutional Layer : Convolutional layer is the most important component of any CNN architecture. It contains a set of convolutional kernels (also called filters) [49]. Every filter is small spatially (along width and height), but extends through the full depth of the input volume [50]. The output of the kernels goes through a linear or non-linear activation function, such as sigmoid, hyperbolic tangent, Softmax, rectified linear, and identity functions) to form

the output feature maps. Each of the output feature maps can be combined with more than one input feature map [33].

The Pooling Layer : Pooling is a key-step in convolutional based systems that reduces the dimensionality of the feature maps [51] also referred as subsampling or downsampling[50]. It combines a set of values into a smaller number of values, i.e., the reduction in the dimensionality of the feature map. It transforms the joint feature representation into valuable information by keeping useful information and eliminating irrelevant information. Pooling operators provide a form of spatial transformation invariance as well as reducing the computational complexity for upper layers by eliminating some connections between convolutional layers [51].
Two common types of pooling layers are max pooling and average pooling: the max pooling selects the maximum valued cell out of the receptive field of that neuron, whereas average pooling passes the average of the cell values in the local receptive field of the neuron. Either before or after the pooling layer an additive bias and sigmoidal nonlinearity is applied to each feature map [47].

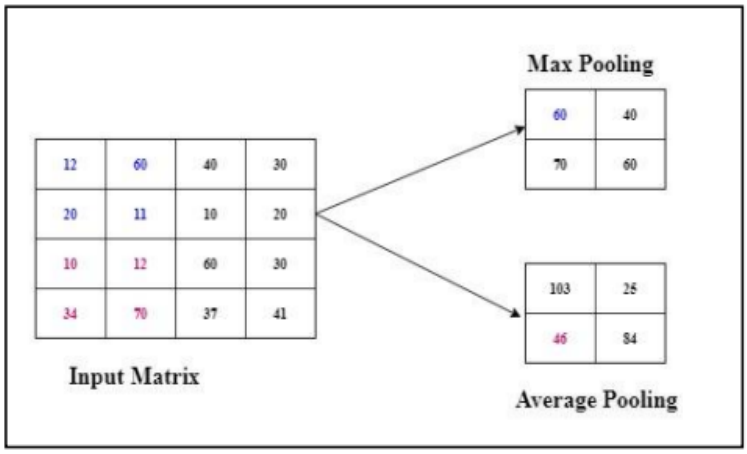

Figure 2.8: Max and Average Pooling [57].

The Fully Connected Layer : The output feature maps of the final convolution or pooling layer is typically flattened, i.e., transformed into a one-dimensional (1D) array of numbers (or vector), and connected to one or more fully connected layers, also known as dense layers, in which every input is connected to every output by a learnable weight. Once the features extracted by the convolution layers and downsampled by the pooling layers are created, they are mapped by a subset of fully connected layers to the final outputs of the network, such as the probabilities for each class in classification tasks. The final fully connected layer typically has the same number of output nodes as the number of classes. Each fully connected layer is followed

by a nonlinear function [52]. In addition, Regularization techniques can be applied in the fully connected layers to prevent overfitting [50]. table 2.1 represents some of hyperparameters used in convolutional neural networks.

Table 2.1: A list of parameters and hyperparameters in a convolutional neural network (CNN)[52]

	Parameters	**Hyperparameters**
Convolution layer	Kernels	Kernel size, number of kernels, stride, padding, activation function
Pooling layer	None	Pooling method, filter size, stride, padding
Fully connected layer	Weights	Number of weights, activation function
Others		Model architecture, optimizer, learning rate, loss function, mini-batch size, epochs, regularization, weight initialization, dataset splitting

2.5.2 Deep residual neural networks

Residual Network (ResNet) is is a Convolutional Neural Network, developed by the MSRA (Microsoft Research Asia) in 2015 [91]. They acknowledged the fact that stacking of more and more convolutional layers does not always lead to better performance [91]. ResNet was developed with the intent of designing ultra-deep networks that did not suffer from the vanishing gradient problem that predecessors had [33]. ResNet is a traditional feedforward network with a residual connection [33]. It comprises many stacked residual blocks with a key component called the **skip connection** [88] that skips one or more layers. Instead of hoping each few stacked layers directly fit a desired underlying mapping, these layers fit a residual mapping. Formally, denoting the desired underlying mapping as $H(x)$ of the input x, the stacked nonlinear layers fit another mapping of $F(x) = H(x) - x$ [91]. The basic block diagram of the ResNet architecture is shown in Figure 2.9.

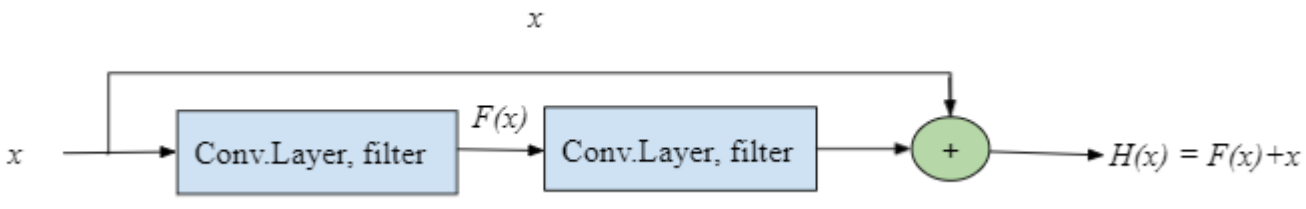

Figure 2.9: Basic diagram of the Residual block [33].

The skip connections in the residual blocks facilitate preserving the norm of the gradient, avoiding by this manner the vanishing gradient problem and leading to stable backpropagation [92].

2.6 Summary of empirical and theoretical findings

While various approaches on classification [93][111] and tagging of acoustic scenes [94] have been proposed in the last years, acoustic anomaly detection is still underrepresented. Due to the release of publicly available datasets [95][96][97][98] the situation is gradually improving and numerous attempts have been made by the research community [104][105][106] The majority of approaches to acoustic anomaly detection relies upon deep autoencoders. For example, Marchi et al, use a bidirectional recurrent denoising AE to reconstruct auditory spectral features to detect novel events [99]. In [101] the authors compare various AE architectures, they conclude that a convolutional architecture operating on the Mel-Frequency Cepstral Coefficients is well suited for the task while a One-Class Support Vector Machine represents a strong and more parameter efficient baseline. Kawaguchi et al [102], explicitly address the issue of background noise. An ensemble method of front-end modules and backend modules followed by an ensemble-based detector combines the strengths of various algorithms. Frontends consist of blind-dereverberation and anomalous sound- extraction algorithms, back-ends are AEs. The final anomaly score is computed by score-averaging [102].
In [86] anomalous sound detection is interpreted as statistical hypothesis testing where they propose a loss function based on the Neyman-Pearson lemma. In contrast to these architecture-driven approaches, Koizumi et al, introduced Batch- Uniformization, a modification to the AE's training procedure where the reciprocal of the probabilistic density of each sample is used to up-weigh rare sounds [104]. Another line of work investigates upon methods that operate directly on the raw waveform [105][112]. These methods use generative, WaveNet-like (Oord et al., 2016)[106] architectures to predict the next sample and take the prediction error as a measure of abnormality. Their results indicate a slight advantage over AE based approaches at the cost of higher computational demands. Fully-supervised approaches for anomaly detection usually ignore unlabeled data during the training-phase: for example, (Almgren and Jonsson, 2004) [107] employ a max-margin classifier that separates the innocuous data from the attacks. Stokes and Platt, present a technique which combines approaches for effective discrimination [108]. Mao et al. take a multi-view and co-training approach based on [110] to learn from labeled and unlabeled data. table 2.2 shows a synthesis of the previous studies analyzed including: dataset, audio features, ML model and evaluation method for each one.

Table 2.2: Summary of related works

Study	Year	Dataset	Audio Features	ML Model	Evaluation Method
[113]	2020	Mivia Dataset [123]	STFT, MFCC, Mel-Scale	DenseNet-121, MobileNetV2, ResNet-50	RR[1], MDR[2], ER[3], FPR,
[114]	2020	ToyADMOS [137],MIMII [136]	Mel-Filterbank	SPIDERnet, AE, Naive MSE, PROTOnet	AUC, ROC, TPR, FPR, F-measure
[115]	2019	UrbanSound8K [126], TUT Dataset [127]	LPC, MFCC, and GFCC	Agglomerative Clustering, BIRCH	Precision, Recall, F1-score, TP, FP, FN
[116]	2019	DCASE2018 Task 1 [128], DCASE2018 Task 2 [129]	FFT, and Log mel spectrogram	CNN	F1-score, AUC, mAP[4], AP[5], ER
[117]	2019	TUT Dataset [127], NAB Data Corpus[130]	Raw data	One-Class SVM, and LSTM-AE	Accuracy
[118]	2019	DCASE 2016 Dataset[131]	MFCC	AE, VAE, and VAEGAN	AUC, TPR, and pAUC
[119]	2019	Toy Car Running Dataset [132]	Time-series of acoustic	AE	AUC
[105]	2018	General Sound Effects Library[133]	Log mel filter bank	WaveNet, AE, BLSTM-AE, AR-LSTM	F1-score
[121]	2018	A3FALL[134]	Log mel energies, DWT	Siamese NN, SVM, One-Class SVM	F1-score, Recall, Precision
[122]	2018	TUT Dataset[127]	MFCC	Elliptic Envelope, Isolation Forest	F1-score, and ER

[1]Recognition Rate
[2]Miss Detection Rate
[3]Error Rate
[4]mean average precision
[5]average precision

in this work we used the MIMII and ToyAdmos dasets used in [114] and we extracted the log mel energies features from every audio file as they are used in many Anomaly Sound Detection tasks [105][121], our models' architecture is inspired by the auto-encoders used in [105][114][118] and a deep residual neural network used in [113][141], we evaluated our developed systems using area under the receiver operating characteristic curve (AUC) and pAUC as they are the most used evaluation metrics in numerous ASD studies [114][116][118] due to the problem of imbalanced datasets in anomaly detection tasks.

2.7 Challenges

2.7.1 Lack of Data

In the case of real world factories, from the view of the development cost, it is impracticable to deliberately damage the expensive target machine. In addition, actual anomalous sounds occur rarely and have high variability [86]. Therefore Obtaining an exhaustive number of recordings from anomalous operation for training is not suitable as it would require either deliberately damaging machines or waiting a potentially long time until enough machines suffered from damages, thus there is an imbalance in the data between number anomalous sounds and non-anomalous sounds.

2.7.2 Noisy Data

In factories, where there are many other types of equipment operating alongside the equipment being tested for anomalies. All the sounds emitted by the additional equipment will constitute background noise that interferes with the collection of sounds from the target equipment. In such environments, the background noise can drown out the sounds made by the operation of the target equipment. Therefore, it is essential to reduce this background noise in order to use sound as a means of detecting equipment operating anomalously [103].

2.8 Conclusion

Throughout this chapter, we have reviewed some important concepts of deep learning methods. First, we have presented the deep neural networks architectures used in our work, both supervised and unsupervised techniques. Furthermore, we have summarized some empirical and theoretical findings on the differences of the architectures presented in this chapter. Finally, we have discussed the challenges related to anomaly sound detection systems.

Chapter 3

Design of Anomaly Detection Systems for Machine Condition Monitoring

3.1 Introduction

This chapter presents the pipeline for building our anomaly detection systems. First, in **Section 3.2** we present our dataset. Then, in **Section 3.3**, we describe our implemented Anomaly Sound Detection systems, and present the features used to train the system as well as the model topology.

3.2 Dataset

The data used for this task comprises parts of ToyADMOS [137] and the MIMII Dataset [136] consisting of the normal/anomalous operating sounds of six types of toy/real machines. Anomalous sounds in these datasets are collected by deliberately damaging the target machines. The following six types of toy/real machines are used in this task: Toy-car and Toy-conveyor from ToyADMOS, and Valve, Pump, Fan, and Slide rail from the MIMII Dataset. To simplify the task, we use only the first channel of multichannel recordings; all recordings are regarded as single-channel recordings of a fixed microphone. Each recording is an approximately 10-sec-long audio that includes both the target machine's operating sound and environmental noise. The sampling rate of all signals has been downsampled to 16 kHz. We mixed a target machine sound with environmental noise, and only noisy recordings are provided as training/test data. The environmental noise samples were recorded in several real factory environments. the details of the recording procedure are in chapter 4 section 4.2.

In this task, we define two important terms: Machine Type and Machine ID. Machine Type means the kind of machine and Machine ID is the identifier of each individual of the same type of machine, which numbers three or four in the training dataset and three in the test dataset. For each Machine Type and Machine ID, the development dataset includes around 1000 sam-

ples of normal sounds for training and 100–200 samples each of normal and anomalous sounds for the test. The evaluation dataset consists of around 400 test samples each for Machine Type and Machine ID, none of which have a condition label (i.e., normal or anomaly). Note that the Machine IDs of the evaluation dataset are different from those of the development dataset. Thus, we also provide an additional training dataset that includes around 1,000 normal samples each for Machine Type and Machine ID used in the evaluation dataset.

3.3 Design and analysis of Anomaly Sound Detection systems

We aim at building two different Anomaly Sound Detection systems for machine condition monitoring that are able to identify whether the sound emitted from a target machine is normal or abnormal under a condition of using only normal sound samples as training data. Our two systems consist of an unsupervised system based on an autoencoder and a supervised system based on a residual neural network. For both our systems we used log-mel energies as the acoustic features with the same parameters as input to our models to calculate the anomaly score of each test sample instead of the decision result. Here, the anomaly score takes a large value when the input signal seems to be anomalous, and vice versa. After calculating anomaly scores per frame, we aggregate them into an anomaly score of the sound signal. Figure 3.1 represents the process of our anomaly sound detection systems.

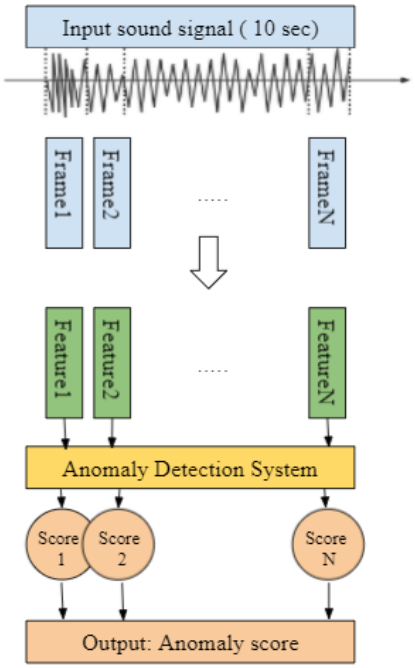

Figure 3.1: Overview of our ASD systems.

3.3.1 Feature Extraction

Feature extraction is one of the most crucial stages in the anomaly sound detection task framework. This stage contributes greatly to the effectiveness of an ASD system. One of the most popular features applied in the ASD tasks are representations of Mel-frequency scales such as Log-Mel energies [140]. The main reason for their success is that they provide a reasonably good representation of the spectral properties of the audio signal. In addition, they produce a reasonably high inter-class variability allowing for class discrimination.

We have extracted the frequency domain features: Log-Mel band energies from the audio samples to train our models. The sampling rate of all signals has been down sampled to 16 kHz. As a first feature extraction step, we have applied a **Short-Time Fourier Transform** with a **Hamming windowing function** size of 1024 and hop length of 512 samples. Next, **64 bin Mel-filter bands** are applied to the calculated power spectrums. Finally, the resulting Mel-energy values are **logarithmically scaled** to obtain the Log-Mel features. We have extracted these features using the Librosa version 0.6.3 python package.Table 3.1 represents a summary of the values and variants used in extracting Log-Mel band energies.

Table 3.1: Log-Mel energies parameters.

Parameter	**Configuration**
Sample rate	16000 Hz
Hamming Window	1024
Mel-filter bands	64
Hop length	512

3.3.2 Neural Network Architectures

We have built two network architectures to build our ASD systems. The first one employs an unsupervised approach which is a simple autoencoder, whereas the second is based on a supervised approach which is a deep residual network.

3.3.2.1 Unsupervised approach: an autoencoder for Sound Anomaly Detection

Autoencoder is a kind of artificial neural network used in semi-supervised learning and unsupervised learning. It can learn the efficient representation of input data, so it is widely applied in dimensionality reduction and anomaly detection.

The encoder and decoder networks consist of four fully-connected layers with 128 hidden units, followed by Batch Normalization and ReLU as the activation function. The bottleneck layer is set as one fully-connected layer with 8 hidden units, resulting in a 8-dimensional latent space. And 5 concatenated frames to form a 320-dimensional input vector. Figure 3.2 illustrates the AE architecture.

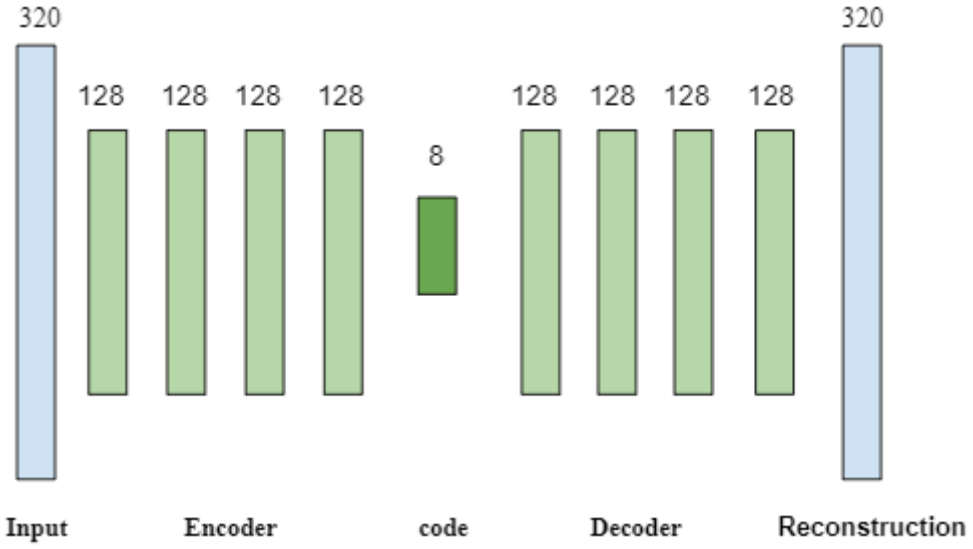

Figure 3.2: AE Network architectures.

The difference between the original input vector and the AE output response is called the reconstruction error , in this task the reconstruction error element is used to detect sound anomalies. Firstly, an AE is trained with only normal sound samples, aiming to minimize the reconstruction error. The obtained model is assumed to be capable of compressing the input features, learning their most relevant relationships. Secondly, the trained AE can be tested with unseen data. If the unseen data is similar to the trained patterns (related to the normal sounds), when the AE should reproduce the new input with good accuracy. However, if the unseen data is anomalous, the AE should not be able to reconstruct the input and the error will be greater. Thus, the magnitude of the reconstruction error can be used to detect anomalies. The test samples' reconstruction error, averaged over the whole sample, is used as anomaly score. we first calculated the log-mel energies of input $X \in R^{F \times T}$ where $F = 64$ and $T = 5$ are the numbers of mel filters and time frames, respectively. Then, the acoustic feature at t is obtained by concatenating before/after P frames of log-mel-filterbank outputs as $\phi_t = (X_{t-P}, ...X_{t+P})$. The anomaly score is calculated as

$$A_\theta(x) = \frac{1}{T} \sum_{t=1}^{T} \parallel \phi_t - AE_\theta(\phi_t) \parallel_2^2 \tag{3.1}$$

where $||.||$ is the $\ell 2$ norm , and AE is an autoencoder with parameter θ. The $\ell 2$ norm calculates the distance of the vector coordinate from the origin of the vector space. As such, it is also known as the Euclidean norm as it is calculated as the Euclidean distance from the origin. The result is a positive distance value. The $\ell 2$ norm is calculated as the square root of the sum of the squared vector values.

The encoder and decoder were trained to minimize the Mean Squared Error (MSE) between input and its reconstruction, using a learning rate of 0.001 and the Adam optimizer for 100 epochs with batch size of 512.

3.3.2.2 Supervised approach: a Residual Network for Sound Anomaly Detection

We choose the model architecture introduced by Koutini et al [141], a receptive-field-regularized, fully convolutional, residual network (ResNet) [91], which has been successfully adopted for various audio-related classification tasks [165]. Figure 3.3 represents our ResNet model architecture that we used in this task , the residual block is represented inside the red frame.

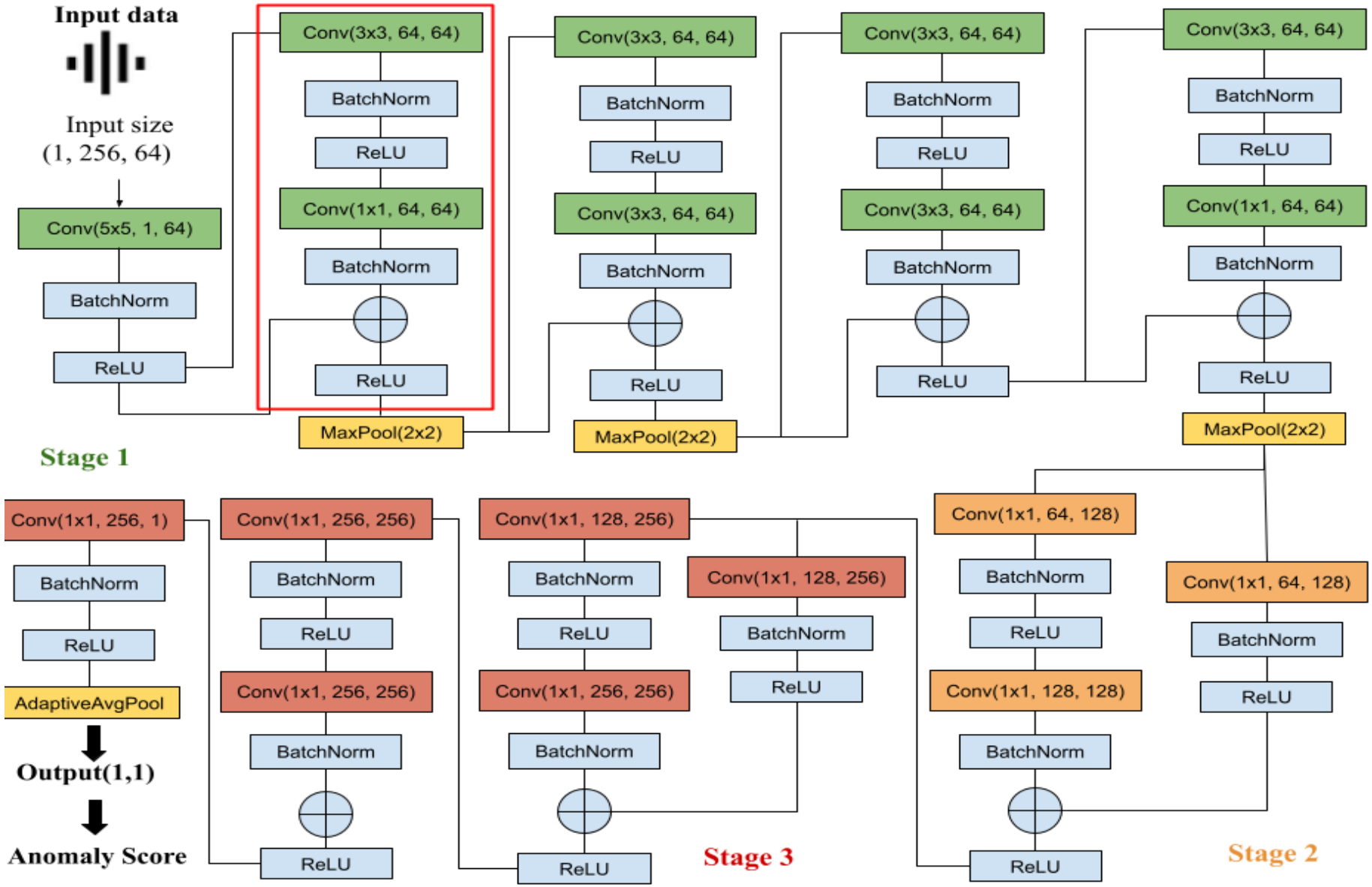

Figure 3.3: The ResNet architecture.

The model consists of three stages with four residual blocks for the first stage, one and two residual blocks for the second and third stages respectively. The first stage is preceded by a convolutional layer with 64 filters of kernel size 5×5. We apply global average pooling after the last layer, which allows us to use inputs of varying sizes. In the first stage, a max-pooling layer with kernel size 2×2 follows after the first, second, and fourth residual block. Each residual block consists of two convolutional layers, where we use 64, 128, and 256 filters in the first, second, and third stage, respectively. In the first stage, we use filters of size 3×3, except for the second convolution of the first and fourth block where we use 1×1 filters. In the other two stages, we use filters of size 1×1. A batch normalization layer follows each convolutional layer and we use ReLU activations.

To overcome the scarcity problem of anomalous sounds, we propose to substitute real abnormals with Outliers, i.e., carefully selected recordings that are neither normal nor abnormal sounds. Note that, compared to anomalous sounds, outliers are cheap and easy to collect if not already available in abundance. To determine what kind of outliers can be utilized for acoustic Machine Condition Monitoring, we take advantage of machine sounds contained in the combined version of the MIMII [136] and ToyAdmos [137] datasets. The training process is for a specific machine instance (ID) and not per machine type which means we use the training set of this particular machine instance as normal data and we set the remaining machines instances' training sets as outliers, Figure 3.4 represents the training method we used in this task.

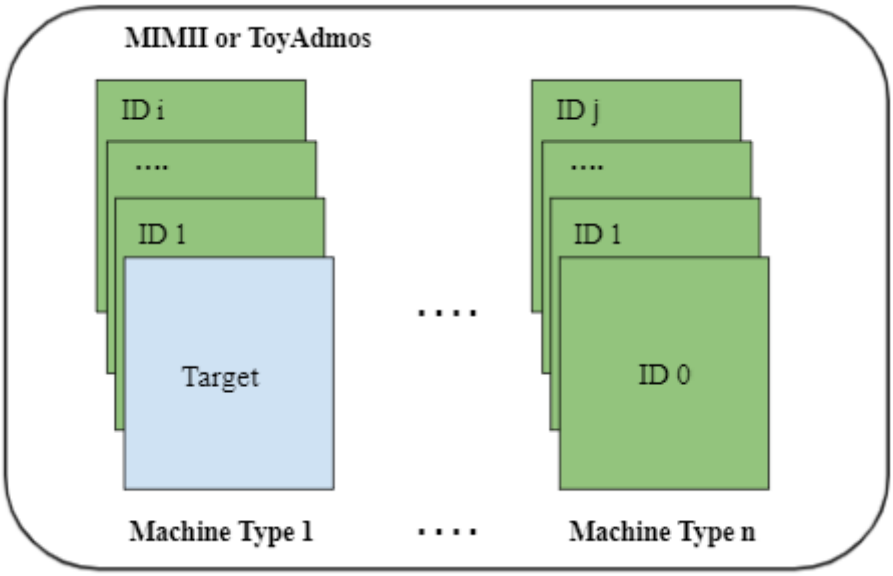

Figure 3.4: Outlier Exposed Strategy.

In Figure 3.4 The target set contains the normal sounds (light blue). outliers sets are selected from the remaining machines' training sets (green).

Our model is trained on random snippets of 256 frames length to minimize the Binary Cross

Entropy (BCE) which is defined as a measure of the difference between two probability distributions for a given random variable or set of events [142]. We have 100 epochs and ADAM update rule and batch size of 64. Batches are stratified to contain 32 positive and 32 negative samples.
The anomaly score for each test example is obtained by collecting all 256-frame windows of the input, computing the logit score for each of them, and then mean aggregating all the logit scores to a single value.

3.4 conclusion

In this chapter, we have described the setup used to conduct our task. We have presented the dataset used to train our systems as well as the features and the models used in our ASD systems. In the following chapter, we will present the results of our experiments and analyze them in order to derive guidelines for building Anomaly Sound Detection systems based on numerous statistical comparisons.

Chapter 4

Experimental Results and Discussion

4.1 Introduction

This chapter describes and discusses the experimental results and findings that we have obtained during our experiments. The main goal of our case study is to compare the performances of several Anomaly detection systems.

4.2 Data acquisition procedure

We have carried out our experiments on the **MIMII** dataset [136] and the **ToyADMOS** dataset [137], first, the MIMII dataset contains the sound of four different types of machines: valves, pumps, fans, and slide rails. The valves are solenoid valves that are repeatedly opened and closed. The pumps are water pumps that drain water from a pool and discharge water to the pool continuously. The fans represent industrial fans, which are used to provide a continuous flow of gas or air in factories. The slide rails represent linear slide systems, which consist of a moving platform and a stage base [136].
Each type of machine includes seven individual machines IDs, a total of 26,092 normal sound segments for all machines were recorded. In addition to this, different real-life anomalous scenarios have been considered for each kind of machine: contamination, leakage, rotating unbalance, rail damage, etc. The various running conditions are listed in Table 4.1.

Table 4.1: list of operations and anomalous conditions [136]

Machine type	**operations**	**Examples of anomalous conditions**
Valve	Open / close repeat with different timing	More than two kinds of contamination.
Pump	Suction from / discharge to a water pool	Leakage, contamination, clogging, etc.
Fan	Normal operation	Unbalanced, voltage change, clogging, etc.
Slide rail	Slide repeat at different speeds	Rail damage, loose belt, no grease, etc.

Secondly, The ToyADMOS dataset consists of normal and abnormal sounds of two sub-datasets for two types of tasks. A different toy is used for each task. The list of sound files for each machine type of ToyAdmos dataset is provided in Table 3.1. The name and overview of each sub-dataset are as follows:

- **Toy car:** Designed for product-inspection task. A toy car runs on an inspection device. Sound data are collected with four microphones arranged close to the inspection device. Anomalous sounds were generated by deliberately damaging the shaft, gears, tires, and voltage of the ToyCar [137], as shown in Table 4.2.

- **Toy conveyor:** Designed for fault diagnosis of a fixed machine. A toy conveyor is fixed on a desk, and sound data are collected with four microphones. One is fixed on the body of the conveyor, and the other three are placed on the desk. Anomalous sounds were generated by deliberately damaging the tension pulley, trail pulley, and belt and excessively lowering/raising the voltage of the ToyConveyor [137], as shown in Table 4.2.

Table 4.2: List of deliberately damaged parts and their conditions [137]

ToyCar		**ToyConveyor**	
Parts	Condition	Parts	Condition
Shaft	- Bent	Tension pulley	- Excessive tension
Gears	- Deformed - Melted	Tail pulley	- Excessive tension - Removed
Tires	- Coiled (plastic ribbon) - Coiled (steel ribbon)	Belt	- Attached metallic object 1 - Attached metallic object 2 - Attached metallic object 3
Voltage	- Over voltage - Under voltage	Voltage	- Over voltage - Under voltage

The number of train and test samples we used in our task for both the development and evaluation datasets of MIMII and ToyAdmos are in table 4.3 below.

Table 4.3: Summary of provided datasets.

Machine Type	**mode**	**ID**	**Audio Files**		**Machine Type**	**mode**	**ID**	**Audio Files**	
			Train	Test				Train	Test
Valve	Dev	00	891		Slide rail	Dev	00	968	
		02	608				02	968	
		04	900				04	434	
		06	892				06	434	
	Eval	01		899		Eval	01		1246
		03		1083			03		1246
		05		1399			05		712
Pump	Dev	00	906		ToyCar	Dev	01	1000	
		02	905				02	1000	
		04	602				03	1000	
		06	936				04	1000	
	Eval	01		1119		Eval	05		1515
		03		819			06		1515
		05		1256			07		1515
Fan	Dev	00	911		ToyConveyor	Dev	01	1000	
		02	916				02	1000	
		04	933				03	1000	
		06	915						
	Eval	01		1360		Eval	04		1555
		03		1340			05		1555
		05		1458			06		1555

4.2.1 Recording environment and setup :

The MIMII dataset was collected using a TAMAGO-03 microphone manufactured by System In Frontier Inc [135]. It is a circular microphone array that consists of eight distinct microphones. The microphone array was kept at a distance of 50 cm from the machines (10 cm in the case of valves), and 10-second sound segments were recorded. Note that each machine sound was recorded in a separate session. Under the running condition, the sound of the machine was recorded as 16-bit audio signals sampled at 16 kHz in a reverberant environment. Apart from the target machine sound, background noise in multiple real factories was continuously recorded

and later mixed with the target machine sound for simulating real environments. For recording the background noise, the same microphone array was used as for the target machine sound [136].
Four omnidirectional microphones (SHURE SM11-CN) were used in ToyAdmos dataset for collecting these sounds, The main advantage of the ToyADMOS dataset over other datasets [138][139] is that it was built under controlled conditions [137].

4.3 Development environment and utility libraries

We have conducted our experiments using Python which is an object oriented open source programming language [167] we begun with extracting features from the dataset and displaying it using librosa which is a python package for audio analysis, Furthermore, the machine learning process was managed using multiple Python packages, including :

Librosa: Librosa is a Python package for audio and music signal analysis and processing. It provides implementations of a variety of common functions that fall into four categories hat are audio and time-series operations, spectrogram calculation, time and frequency conversion, and pitch operations [168] These functions are heavily used throughout our experiments.

Tensorflow: Tensorflow is an open-source library that implements automatic learning methods based on the principle of deep learning neural networks. We have used Tensorflow in our work as it supports a variety of applications, with a focus on training and inference on deep neural networks [169].

Keras: Keras is a high-level API written in python that runs on a Tensorflow backend. It is an approachable, highly-productive interface for solving machine learning problems, with a focus on modern deep learning. Its simplicity helps users develop a deep learning model quickly and provides a ton of flexibility while still being a high-level API [170].

Pytorch: PyTorch is an open-source library developed by Facebook that performs instantaneous dynamic tensor computations with automatic differentiation and GPU acceleration, while maintaining performance comparable to the fastest modern libraries for deep learning [34].
In addition to the above mentioned deep learning libraries we have made use of the Numpy library to perform manipulation operations on our data, the Matplotlib library for plotting and

graphical representations, tqdm library that allow the output of smart progress bars. Table 4.4 provides additional information about the libraries we have used in our work.

Table 4.4: Utility python libraries

Utility Library	**Version**
Python	3.7
Librosa	0.8.1
Keras	2.6.0
Tensorflow	2.6.0
PyTorch	1.0.0
Matplotlib	3.2.2
Numpy	1.19.5
Tqdm	4.62.0

We have trained our classifiers using Google Colaboratory which is a google cloud environment,for machine learning, data analysis and educationit consist of an executable Python notebooks (Jupyter notebooks) that execute code on Google's cloud servers, and are highly integrated with Google Drive which we used as a storage of our dataset, making them easy to set up, access, and share.

4.4 Evaluation

Both of our systems are evaluated using the area under the receiver operating characteristic (ROC) curve (AUC) and the partial-AUC ($pAUC$). The $pAUC$ is an AUC calculated from a portion of the ROC curve over a prespecified range of interest. In our metric, the $pAUC$ is calculated as the AUC over a low false-positive-rate (FPR) range $[0,\ p]$ with $p = 0.1$. The reason for the additional use of the $pAUC$ is based on practical requirements. If an ASD system gives false alerts frequently, it could not be trusted Therefore, it is especially important to increase the true-positive-rate under low FPR conditions. AUC and $pAUC$ are calculated as

$$AUC = \frac{1}{N_- N_+} \sum_{i=1}^{N_-} \sum_{j=1}^{N_+} H(A_\theta(x_j^+) - (x_i^-)) \tag{4.1}$$

$$pAUC = \frac{1}{\lfloor pN_- \rfloor N_+} \sum_{i=1}^{N_-} \sum_{j=1}^{N_+} H(A_\theta(x_j^+) - (x_i^-)) \tag{4.2}$$

where $\lfloor . \rfloor$ is the flooring function and $H(x)$ returns 1 when $x > 0$ and 0 otherwise. Here, x_i^- and x_j^+ 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.

4.5 Experiments

The main goal of this work is to design Anomaly detection systems for monitoring machine condition. To this end, we have developed two systems: an AutoEncoder and an outlier-exposed ResNet. Note that the description of both systems is provided in Chapter 3. In this section, we describe and discuss three conducted experiments :

- Experiment 1: Supervised vs Unsupervised.
- Experiment 2: Impact of the code size.
- Experiment 3: Deep vs Shallow Neural Networks.

4.5.1 Experiment 1 : Supervised vs Unsupervised

We dedicate this experiment to compare the AutoEncoder system with the outlier-exposed ResNet. Table 4.5 reports the obtained AUC and pAUC scores of each machine.

Table 4.5: AUC and pAUC scores of AE and ResNet.

		Outlier Exposure		**AutoEncoder**	
ToyCar	**Machine ID**	**AUC**	**pAUC**	**AUC**	**pAUC**
	1	81.89 %	74.84 %	62.75 %	52.66 %
	2	87.69 %	76.74 %	64.41 %	53.08 %
	3	96.07 %	90.6 %	55.94 %	52.16 %
	4	99.91 %	99.53 %	61.94 %	51.07 %
	Average	**91.39%**	**85.43 %**	**61.26 %**	**52.24 %**
Toy Conveyor	**Machine ID**	**AUC**	**pAUC**	**AUC**	**pAUC**
	1	88.45 %	80 %	73.97 %	59.99 %
	2	77.05 %	63.12 %	62.58 %	54.83 %
	3	72.82 %	62.10 %	68.6 %	57.36 %
	Average	**79.44 %**	**68.41%**	**68.39 %**	**57.4 %**
Fan	**Machine ID**	**AUC**	**pAUC**	**AUC**	**pAUC**
	0	62.39 %	60.48 %	58.31 %	50.23 %
	2	98.48 %	95.10 %	46.68 %	50.10 %
	4	76.23 %	67.15 %	60.28 %	52.17 %
	6	100 %	100 %	48.22 %	51.07 %
	Average	**84.27 %**	**80.63 %**	**53.37 %**	**50.89 %**
Pump	**Machine ID**	**AUC**	**pAUC**	**AUC**	**pAUC**
	0	82.43 %	78.46 %	65.06 %	55.64 %
	2	76 %	61.87 %	57.6 %	57.51 %
	4	99.83 %	99.10 %	81.56 %	58.10 %
	6	97.5	93.6 %	72.15 %	56.96 %
	Average	**88.94 %**	**83.26 %**	**69.09 %**	**57.05 %**
Slider	**Machine ID**	**AUC**	**pAUC**	**AUC**	**pAUC**
	0	99.58%	97.81 %	91.78 %	62.55 %
	2	91.43 %	75.61 %	76.39 %	56.75 %
	4	99.76 %	98.75 %	94.52%	72.85 %
	6	83.13 %	48.07 %	62.44 %	48.07 %
	Average	**93.47 %**	**80.06 %**	**81.28 %**	**60.05 %**
Valve	**Machine ID**	**AUC**	**pAUC**	**AUC**	**pAUC**
	0	100 %	100 %	68.42 %	53.69 %
	2	32.07 %	48.99 %	63.12 %	51.27 %
	4	98.89 %	96.14 %	74.65 %	52.45 %
	6	99.81 %	99.56 %	56.36 %	49.38 %
	Average	**82.69 %**	**86.17 %**	**65.64 %**	**51.7 %**

The performance of each system in terms of AUC and pAUC averaged per machine type is illustrated in Figures 4.1 and 4.2. The results shown in Table 4.5 indicate that the outlier-exposed ResNet outperforms the AutoEncoder system over most machines and by a large margin. In order to assess whether this margin is significant or not, we have opted for a sign test. The sign test is a non-parametric statistical test alternative to the paired t-test. In addition, according to numerous authors [65], this test is considered the best strategy to compare two algorithms over multiple domains when the data is drawn from unknown distribution. Under the null hypothesis, we assume that the observed differences are due to chance. The results as shown in Table 4.5 indicate that the supervised learner achieves 6 significant wins over its counterpart learner. According to statistical tables [65], this difference is significant with p-$value < 0.05$; specifically ($p - value = 0.03$), which allows to. This finding confirms our initial observation regarding the difference in performances.

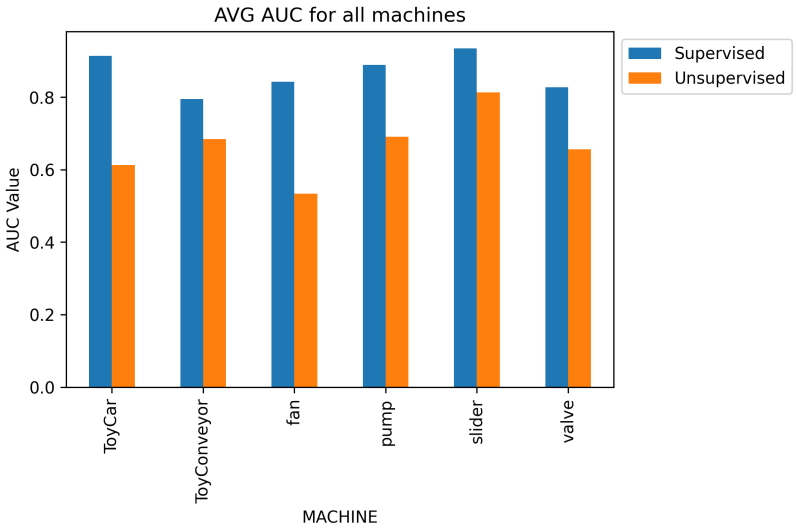

Figure 4.1: Average AUC per Machine Type

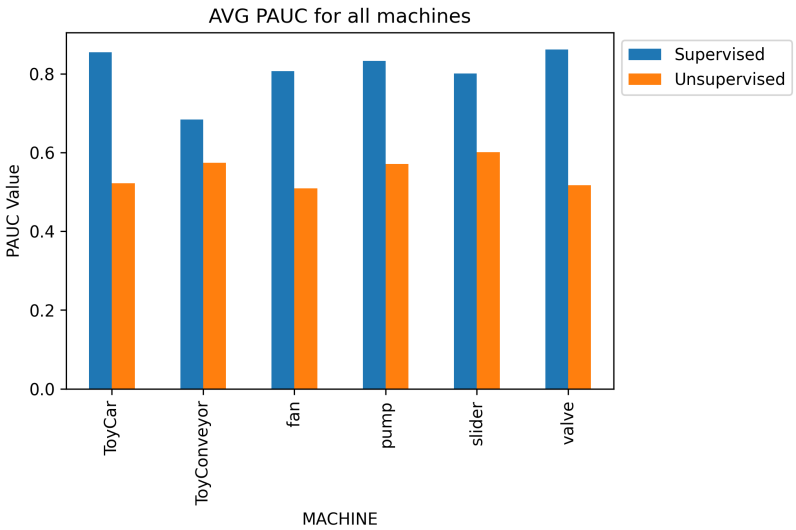

Figure 4.2: Average pAUC per Machine Type

4.5.2 Experiment 2 : Impact of the code size

In this experiment, we investigate the impact of varying the size of the bottleneck layer, i.e the code size, on the performance of the AutoEncoder-based system with regards to each machine type. To this end, we have carried out the following experiment: we have tested 10 values of code sizes 4, 5, 8, 10, 12, 20, 40, 80, 160, 320 and plotted the AUC score for each machine type. We report in Figure 4.3 the obtained results.

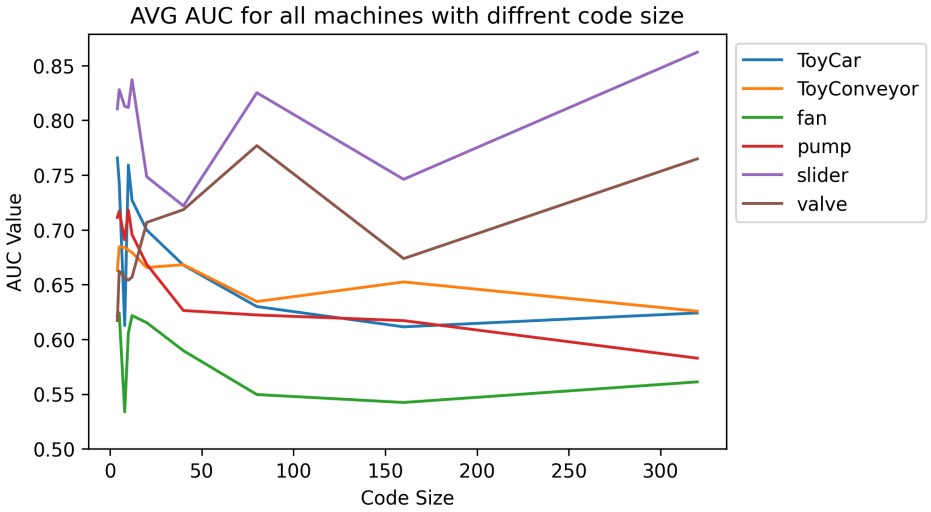

Figure 4.3: Average AUC per Machine Type with different code size

We can summarize the analysis of the illustrated results by two main observations:

1. For machine type: slider and valve, increasing the code size has an **overall positive** impact on the AUC scores. Specifically, the curves exhibit fluctuation for all code sizes. This behavior is expected since we have not tested sufficient values of code sizes, which make the curves appear inconsistent and noisy. Most importantly, we notice a remarkable increase in AUC scores when the code size is set to 320. Additional investigations should be conducted in order to fully understand the impact of the code size on these two machine types.

2. For the remaining machine types, increasing the code size has an **overall negative** impact on AUC scores. Specifically, the curves fluctuate for code sizes lower than 50; then begin decreasing slowly as the code size gets larger.

4.5.3 Experiment 3 : Deep vs Shallow Neural Networks

The goal of this experiment is to examine the influence of the number of hidden layers of the autoencoder. To this end, we have built several detection systems, while varying the number of hidden layers. We have set this parameter to the following values : 1, 2, 3, 4 and 5. We report in Table 4.6 the AUC scores of these 5 systems.

Table 4.6: AUC and pAUC scores (%) for different number of layers

	1 Layer		**2 Layers**		**3 Layers**		**4 Layers**		**5 Layers**	
	AUC	**pAUC**	**AUC**	**pAUC**	**AUC**	**pAUC**	**AUC**	**pAUC**	**AUC**	**pAUC**
ToyCar	75.49	64.82	77.53	66.37	75.33	64.63	61.26	52.24	75.34	65.22
ToyConveyor	65.18	55.62	67.89	57.49	67.86	58.01	68.39	57.4	68.9	58.41
Fan	60.68	52.47	61.82	53.03	62.11	52.28	53.37	50.89	62.55	52.14
Pump	68.94	58.29	70.74	60.31	71.63	61.88	69.09	57.05	68.73	54.17
Slider	79.04	61.48	82.13	64.35	81.94	63.31	81.28	60.05	84.11	68.04
Valve	51.29	49.87	59.16	50.44	53.14	48.81	65.64	51.7	66.01	51.1

In order to compare these scores, we have followed Demsar methodology [65]. We have chosen the Wilcoxon signed-ranks test. This test is used for comparing two techniques over multiple domains; in this case, machine types. The main difference between the sign and Wilcoxon test is that, the Wilcoxon test ranks the differences between the scores and compares the ranks for the positive and the negative differences, whereas the sign test counts the number of machine types on which a system is the overall winner without considering the differences between the scores.

Table 4.7: Summary of the Wilcoxon signed-ranks statistics.

		Layer 2	**Layer 3**	**Layer 4**	**Layer 5**
Layer 1	W/T/L	**6/0/0**	**5/0/1**	4/0/2	4/0/2
	p-value	**0.0277**	**0.0464**	0.7532	0.1159
Layer 2	W/T/L		3/0/3	2/0/4	4/0/2
	p-value		0.6002	0.2489	0.7532
Layer 3	W/T/L			2/0/4	5/0/1
	p-value			0.1730	0.3454
Layer 4	W/T/L				**5/0/1**
	p-value				**0.0464**

We compare in Table 4.7 the AUC scores in a pairwise manner based on the Wilcoxon test. We report in each entry of this table the number of Win/Tie/Loss of the system in the column over the system in the row, and the second row shows the p-values for the Wilcoxon test.
The results shown in Table 4.7 indicate that:
the models trained with two and three layers exhibit significantly better performance compared to the one layer model with p-value 0.02 and p-value 0.04 respectively.
the 5 layer model outperforms the 4 layer model with p-value 0.04 which leaves the 4 layer model to have the poorest performance among the other models.
Based on these observations, we can conclude that the number of the hidden layers influences on the AE system performance. Specifically, our analysis indicates that deep architectures do not always guarantee better predictive performance because they tend often to overfit [157], although the shallow model (one layer) on the other hand has the poorest performance within the other models, thus we observed that the 2 layers and 3 layers models surpasses the shallow and deeper models (4/5 layers) at predicting anomalies.

Conclusion

The primary goal of this thesis is to conduct empirical analysis and comparisons among Anomaly Sound Detection systems that are able to distinguish between normal and abnormal sounds emitted from a target machine. To this end, we have carried out two sets of ASD systems and conducted multiple experiments to analyze the behavior of the unsupervised system.

summary of experimental findings

We have carried out our experiments on the **MIMII** dataset [136] and the **ToyADMOS** dataset [137] consisting of the normal/anomalous operating sounds of six types of toy/real machines which consists of 54, 254 sound files in wav format. In this project we made use of two deep learning-based models: AutoEncoder (AE) and Residual Neural Network (ResNet). We trained these models on Log-Mel energies features using the same parameters for both models.To evaluate each developed system, we used AUC and pAUC due to its efficacy with unbalanced datasets. Finally, we supported our analysis and discussion with various powerful statistical tests namely: Sign test and Wilcoxon signed rank test. From this experimental study, we can derive the following conclusions:

1. Framing the anomaly sound detection as a supervised classification problem using a deep residual neural network showed robust and consistent results of the ASD system compared to the unsupervised system.

2. Increasing the code size of the autoencoder has an overall positive impact on detecting anomalies for some machines and a negative impact for the other remaining machines.

3. Varying the number of the hidden layers of the autoencoder model showed that shallow AE has a poor performance compared to deeper AEs.

4. Statistically testing the obtained scores is a powerful mechanism for comparing and unraveling existing differences among anomaly sound detection systems.

Contributions

Although a great number of Anomaly Sound Detection systems have been developed using different audio processing methods and machine learning paradigms, most of them have focused on extracting relevant features and finding suitable classifiers to improve the overall performance. Furthermore, several seminar papers have been published recently, most of them have invoked recent deep learning architecture, for instance: **AutoEncoders**, **Deep Residual Neural Network** [13]. Motivated by these needs, we have designed and analyzed the behavior of two Anomaly Sound Detection systems. Additionally, we have conducted extensive experimental comparisons among the developed systems. In what follows, we summarize our main contributions:

1. We conduct thorough and extensive experiments on anomaly sound detection systems trained using a large-scale audio dataset of parts of ToyADMOS [137] and the MIMII Dataset [136] consisting of the normal/anomalous operating sounds of six types of toy/real machines.

2. We have designed our ASD systems using well-known deep neural network architectures, which have been successfully used in audio-related tasks [141][114]. Specifically, we have studied Deep Residual Neural Network (ResNet) and AutoEncoder (AE). In addition, we have supported our analysis and discussion with numerous statistical tests.

3. We analyze the importance of well-tuning of some parameters of the unsupervised model architecture and its impact on the performance of the ASD system.

4. We expand our research work by statistically comparing the two developed ASD systems using numerous tests.

Limits and Future work

In the previous section, we have summarized our main contributions. However, it is of paramount importance to specify the limitations of our work and highlight potential future work directions. In our experiments, we have only considered the use of the log-mel energy feature, an appealing work direction would be to thoroughly investigate other features such as MFCC and LFC.

Similarly, the hyperparameters used for training the deep learning models considerably affect the detection ability. A natural extension of this work would be to investigate tuning several hyperparameters for the ResNet model by varying the learning rate instead of fixing it which showed a better impact in numerous studies. Another appealing work direction would be to use different advanced deep learning architectures for both our systems such as Convolution Recurrent Neural Network (CRNN) for the supervised system and Denoising AE, variational AE for the unsupervised system. During this project, we have encountered many difficulties. The training of the learning models took a very long time due to the lack of dedicated computational platforms. In addition, when performing model selection, storing the trained classifiers caused a considerable increase in the usage of memory space.

In this project we learned the main steps for building a machine learning experiment, and how to statistically compare between systems. We also learned how to process sound and extract useful features from it for further machine learning tasks.

Bibliography

[1] P. Rao, Audio Signal Processing , Chapter in Speech, Audio, Image and Biomedical Signal Processing using Neural Networks, (Eds.) Bhanu Prasad and S. R. Mahadeva Prasanna, Springer-Verlag, 2007.

[2] S. H. Mneney, An Introduction to Digital Signal Processing: A Focus on Implementation.

[3] Perry R. Cook , Real Sound Synthesis for Interactive Applications.

[4] Hack Audio, An Introduction to Computer Programming and Digital Signal Processing in MATLAB.

[5] J. Belleman CERN, Geneva, Switzerland , From analog to digital.

[6] Stanley H. Mneney University of KwaZulu-Natal Durban South Africa An Introduction to Digital Signal Processing: A Focus on Implementation.

[7] Tuomas Virtanen , Mark D. Plumbley , Dan Ellis Editors, Computational Analysis of Sound Scenes and Events.

[8] Müller, M. (2015). Fourier Analysis of Signals. Fundamentals of Music Processing, 39–114. doi:10.1007/978-3-319-21945-5-2.

[9] Gerhard, David Bruce. "Computationally measurable differences between speech and song." PhD diss., Theses (School of Computing Science)/Simon Fraser University, 2003.

[10] Elham Babaee, Nor Badrul Anuar, Ainuddin Wahid Abdul Wahab, Shahaboddin Shamshirband Anthony T. Chronopoulos (2018): An Overview of Audio Event Detection Methods from Feature Extraction to Classification, Applied Artificial Intelligence, DOI: 10.1080/08839514.2018.1430469.

[11] Trends in audio signal feature extraction methods Garima Sharma , Kartikeyan Umapathy, Sridhar Krishnan The Department of Electrical and Computer Engineering, Ryerson University, ON M5B 2K3, Canada.

[12] Logan B (2000) Mel frequency cepstral coefficients for music modeling. Proc of the Intl Symp on Music Information Retrieval (ISMIR).

[13] Anomalous Sound Detection with Machine Learning: A Systematic Review Eduardo Carvalho NUNES, ALGORITMI Centre, Department of Information Systems, University of Minho, Guimaraes, Portugal.

[14] Tom Bäckström, "Introduction to Speech Processing," Alto University Wiki, 2019.

[15] Haytham Fayek, Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between.

[16] http://www.britannica.com/EBchecked/topic/1116194/ machine-learning This is a tertiary source that clearly includes information from other sources but does not name them.

[17] C. Sammut and G. I. Webb, eds., Encyclopedia of Machine Learning. Springer, 2010.

[18] C. M. Bishop (2006). Pattern Recognition and Machine Learning. Springer. ISBN 0-387-31073-8.

[19] Russell, Stuart; Norvig, Peter (2003) [1995]. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall. ISBN 978-0137903955.

[20] Olivier Chapelle, Bernhard Scholkopf ,Alexander Zien , Semi-Supervised Learning, The MIT Press ,Cambridge, Massachusetts London, England.

[21] Xiangli Yang, Zixing Song, Irwin King, Fellow, IEEE, Zenglin Xu, Senior Member, IEEE, A Survey on Deep Semi-supervised Learning.

[22] Y. C A Padmanabha Reddy, P. Viswanath, and B. Eswara Reddy, "Semi-supervised learning: a brief review," Int. J. Eng. Technol., vol. 7, no. 1.8, p. 81, 2018, doi: 10.14419/ijet.v7i1.8.9977.

[23] P. Dönmez, "Introduction to Machine Learning, 2nd ed., by Ethem Alpaydın. Cambridge, MA: The MIT Press2010.

[24] Ng A. "CS229 Lecture notes."

[25] Vishal Sharma , Survey of Classification Algorithms and Various Model Selection Methods .

[26] M. Pérez-Ortiz, S. Jiménez-Fernández, P. A. Gutiérrez, E. Alexandre, C. Hervás-Martínez, and S. Salcedo-Sanz, "A review of classification problems and algorithms in renewable energy applications," Energies, vol. 9, no. 8, pp. 1–27, 2016, doi: 10.3390/en9080607.

[27] J. Bergstra, O. Breuleux, F. F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math compiler in Python," in Proceedings of the Python for Scientific Computing Conference (SciPy), no. Scipy, 2010, pp. 1–7.

[28] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, Q. V. Le, and A. Y. Ng, "Large Scale Distributed Deep Networks," in Advances in Neural Information Processing Systems 25 (NIPS 25). Curran Associates Inc., 2012, pp. 1223–1231.

[29] Sonali. B. Maind, Priyanka Wankar, Research Paper on Basic of Artificial Neural Network.

[30] Z. H. Ling, S. Y. Kang, H. Zen, A. Senior, M. Schuster, X. J. Qian, H. M. Meng, and L. Deng, "Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends," IEEE Signal Processing Magazine, vol. 32, no. 3, pp. 35–52, 2015.

[31] J. Schmidhuber, "Deep learning in neural networks: An overview," Neural Networks, vol. 61, pp. 85–117, 2015.

[32] D. Yu, and L. Deng, "Deep learning and its applications to signal and information processing [exploratory dsp]," IEEE Signal Processing Magazine, vol. 28, no. 1, pp. 145–154, 2011.

[33] Z. Alom et al., "A State-of-the-Art Survey on Deep Learning Theory and Architectures," Electronics, vol. 8, no. 292, pp. 1–67, 2019, doi: 10.3390/electronics8030292.

[34] T. Viehmann, Eli Stevens Luca Antiga. .

[35] Convolutional Neural Networks for Visual Recognition. (n.d.). Retrieved November 27, 2016, from http://cs231n.github.io/neural-networks-1/ .

[36] EMRE CAKIR, Deep Neural Networks for Sound Event Detection .

[37] Nitish Srivastava , Geoffrey Hinton , Alex Krizhevsky , Ilya Sutskever , Ruslan Salakhutdinov, Dropout: A Simple Way to Prevent Neural Networks from Overfitting.

[38] Pierre Baldi, Peter Sadowski, Understanding Dropout.

[39] Jacob Kasche , Fredrik Nordström , Regularization Methods in Neural Networks.

[40] Sergey Demyanov, Regularization Methods for Neural Networks and Related Models.

[41] Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10).

[42] Schmidhuber, J. Deep Learning in Neural Networks: An Overview. Neural Netw. 2015, 61, 85–117.

[43] Ackley, D.H.; Hinton, G.E.; Sejnowski, T.J. A learning algorithm for Boltzmann machines. Cogn. Sci. machines. Cogn. Sci. 1985, 9,147–169.

[44] Fukushima, K. Neocognitron: A hierarchical neural network capable of visual pattern recognition. Neural Netw. 1988, 1, 119–130.

[45] LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. Proc. IEEE 1998, 86, 2278–2324.

[46] Jianyong Tang, Ronald N Germain, and Meng Cui. Superpenetration optical microscopy by iterative multiphoton adaptive compensation technique. Proceedings of the National Academy of Sciences of the United States of America, 109(22):8434–8439, 05 2012.

[47] A. Kumar, "Deep Learning Methods for Classification with Limited Training Data Seminar Report : Spring 2017 Aviral Kumar," Bombay, 2017.

[48] Phung, V.H.; Rhee, E.J. A Deep Learning Approach for Classification of Cloud Image Patches on Small Datasets. J. Inf. Commun. Converg. Eng. 2018, 16, 173–178, doi:10.6109/jicce.2018.16.3.173.

[49] Anirudha Ghosh, Abu Sufian, Farhana Sultana, Amlan Chakrabarti, Debashis De, Fundamental Concepts of Convolutional Neural Network.

[50] Timea Bezdan, Nebojša Bačanin Džakula, Convolutional Neural Network Layers and Architectures.

[51] Hossein Gholamalinezhad , Hossein Khosravi , Pooling Methods in Deep Neural Networks, a Review.

[52] Gieseke, F.; Airola, A.; Pahikkala, T.; Kramer, O. Fast and Simple Gradient-Based Optimization for Semi-Supervised Support Vector Machines. Neurocomputing 2014, 123, 23–32.

[53] Lee, J.-S.; Du, L.-J. Unsupervised classification using polarimetric decomposition and the complex Wishart classifier. IEEE Trans. Geosci. Remote Sens. 1999, 37, 2249–2258.

[54] Pahikkala, T.; Airola, A.; Gieseke, F.; Kramer, O. Unsupervised Multi-Class Regularized Least-Squares Classification. In Proceedings of the 12th IEEE International Conference on Data Mining (ICDM), Brussels, Belgium, 10–13 December 2012; pp. 585–594.

[55] Pierre Baldi, Autoencoders, Unsupervised Learning, and Deep Architectures.

[56] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio and Pierre-Antoine Manzagol, 'Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion.'

[57] Harshini Sewani ,Rasha Kashef , An Autoencoder-Based Deep Learning Classifier for Efficient Diagnosis of Autism, Department of Electrical, Computer, and Biomedical Engineering, Ryerson University, Toronto.

[58] Kingma D. P., Welling M. Auto-Encoding Variational Bayes [J]. Stat, 2014, 1050: 10

[59] Bengio Y., Lamblin P., Popovici D., Larochelle H. Greedy Layer-Wise Training of Deep Networks [J]. Advances in Neural Information Processing Systems, 2007, 19: 153-160.

[60] Huajing Wei, Yuanmeng Hu, review of deep neural network based on auto-encoder.

[61] Vincent P., Larochelle H., Bengio Y., Manzagol, P. A. Extracting and Composing Robust Features with De-Noising Auto-Encoders [C]// International Conference on Machine Learning. ACM, 2008: 1096-1103.

[62] Dor Bank, Noam Koenigstein, Raja Giryes, Autoencoders.

[63] Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks, 1999.

[64] Hassan Mohamed Hassan, Abdelazim M Negm, Mohamed Zahran, Oliver Saavedra, assessment of artificial neural network for bathymetry estimation using high resolution satellite imagery in shallow lakes : case study el burullus lake.

[65] J. Demˇ, "Statistical Comparisons of Classifiers over Multiple Data Sets," vol. 7, pp. 1–30, 2006.

[66] Paul Primus , Verena Haunschmi , Patrick Praher, Gerhard Widmer , Anomalous Sound Detection As a Simple Binary Classification Problem With Careful Selection of Proxy Outlier Examples .

[67] Paul Primus, Reframing Unsupervised Machine Condition Monitoring as a Supervised Classification Task With Outlier-Exposed Classifiers .

[68] Carina Silberer, Mirella Lapata, Learning Grounded Meaning Representations with Autoencoders.

[69] Volodymyr Kovenko, Ilona Bogach, A Comprehensive Study of Autoencoders' Applications Related to Images.

[70] Thavavel Vaiyapuri, Adel Binbusayyis, Application of deep autoencoder as an one-class classifier for unsupervised network intrusion detection: a comparative evaluation.

[71] Collison, P. (1998). Of bombers, radiologists, and cardiologists: time to ROC, Heart, 80, 215–217.

[72] Lasko, T. A.; Bhagwat, J. G.; Zou, K. H. and Ohno-Machado, L. (2005). The use of receiver operating characteristic curves in biomedical informatics, Journal of Biomedical Informatics, 38, 404–415.

[73] T. Fawcett. An introduction to ROC analysis. Pattern Recognition Letters - Special issue: ROC analysis in pattern recognition, 27(8):861–874, June 2006.

[74] Luzia Goncalves, Ana Subtil, Rosario Oliveira, Patricia de Zea Bermudez, ROC Curve Estimation : an Overview .

[75] H. Narasimhan and S. Agarwal. A Structural SVM Based Approach for Optimizing Partial AUC. In Proceedings of the 30th International Conference on Machine Learning, pages 516–524, 2013.

[76] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," Inf. Sci. (Ny)., vol. 180, no. 10, pp. 2044–2064, 2010, doi: 10.1016/j.ins.2009.12.010.

[77] D. J. Sheskin. Handbook of parametric and nonparametric statistical procedures. Chapman Hall/CRC, 2000.

[78] S. L. Salzberg. On comparing classifiers: Pitfalls to avoid and a recommended approach. Data Mining and Knowledge Discovery, 1:317–328, 1997.

[79] David Gerhard, Audio Signal Classification: History and Current Techniques.

[80] M. Müller, Fundamentals of Music Processing, DOI 10.1007/978-3-319-21945-5-2.

[81] C. Clavel, T. Ehrette, and G. Richard "Events Detection for an Audio- Based Surveillance System," In Proc. of ICME, 2005.

[82] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Audio Surveillance of Roads: A System for Detecting Anomalous Sounds,"IEEE Trans. ITS, pp.279–288, 2016.

[83] P. Coucke, B. De. Ketelaere, and J. De. Baerdemaeker, "Experimental analysis of the dynamic, mechanical behavior of a chicken egg," Journal of Sound and Vibration, Vol. 266, pp.711–721, 2003.

[84] Y. Chung, S. Oh, J. Lee, D. Park, H. H. Chang and S. Kim, "Automatic Detection and Recognition of Pig Wasting Diseases Using Sound Data in Audio Surveillance Systems," Sensors, pp.12929–12942, 2013.

[85] A. Yamashita, T. Hara, and T. Kaneko, "Inspection of Visible and Invisible Features of Objects with Image and Sound Signal Processing," in Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2006), pp. 3837–3842, 2006.

[86] Y. Koizumi, S. Saito, H. Uematsu, and N. Harada, "Optimizing Acoustic Feature Extractor for Anomalous Sound Detection Based on Neyman- Pearson Lemma," in Proc. of EUSIPCO, 2017.

[87] R. Lee, Ed., Software Engineering Research, Management and Applications, vol. 578. Cham: Springer International Publishing, 2015.

[88] Jingfeng Zhang , Bo Han , Laura Wynter , Bryan Kian, Hsiang Low and Mohan Kankanhalli, Towards Robust ResNet: A Small Step but a Giant Leap.

[89] Peeters, G.: A large set of audio features for sound description (similarity and classification) in the CUIDADO project. Technical Report, IRCAM, Paris (2004).

[90] Razvan Pascanu, Tomas Mikolov, Yoshua Bengio, On the difficulty of training recurrent neural networks.

[91] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep Residual Learning for Image Recognition.

[92] A. Zaeemzadeh, N. Rahnavard, and M. Shah, "Norm-Preservation: Why Residual Networks Can Become Extremely Deep?," IEEE Trans. Pattern Anal. Mach. Intell., pp. 1–1, 2020, doi: 10.1109/tpami.2020.2990339.

[93] Mesaros, A., Heittola, T., and Virtanen, T. (2018). A multi-device dataset for urban acoustic scene classification. In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018), pages 9–13.

[94] Fonseca, E., Plakal, M., Font, F., Ellis, D. P. W., and Serra, X. (2019). Audio tagging with noisy labels and minimal supervision. In Submitted to DCASE2019 Workshop, NY, USA.

[95] Jiang, Y., Li, C., Li, N., Feng, T., and Liu, M. (2018). Haasd: A dataset of household appliances abnormal sound detection. In Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence, CSAI '18, page 6–10, New York, NY, USA. Association for Computing Machinery.

[96] 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.

[97] Koizumi, Y., Saito, S., Uematsu, H., Harada, N., and Imoto, K. (2019). Toyadmos: A dataset of miniature machine operating sounds for anomalous sound detection. In 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), pages 313–317. IEEE.

[98] Grollmisch, S., Abeber, J., Liebetrau, J., and Lukashevich, H. (2019). Sounding industry: Challenges and datasets for industrial sound analysis. In 2019 27th European Signal Processing Conference (EUSIPCO), pages 1–5. IEEE.

[99] Marchi, E., Vesperini, F., Eyben, F., Squartini, S., and Schuller, B. (2015). A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional lstm neural networks. In 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP), pages 1996–2000. IEEE.

[100] Duman, T. B., Bayram, B., and ˙Ince, G. (2019). Acoustic anomaly detection using convolutional autoencoders in industrial processes. In International Workshop on Soft Computing Models in Industrial and Environmental Applications, pages 432–442. Springer.

[101] 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.

[102] Kawaguchi, Y., Tanabe, R., Endo, T., Ichige, K., and Hamada, K. (2019). Anomaly detection based on an ensemble of dereverberation and anomalous sound extraction. In ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 865–869.

[103] Hisashi Uematsu, Yuma Koizumi, Shoichiro Saito, Akira Nakagawa, and Noboru Harada, Anomaly Detection Technique in Sound to Detect Faulty Equipment.

[104] Koizumi, Y., Saito, S., Yamaguchi, M., Murata, S., and Harada, N. (2019). Batch uniformization for minimizing maximum anomaly score of dnn-based anomaly detection in sounds. In 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), pages 6–10.

[105] Hayashi, T., Komatsu, T., Kondo, R., Toda, T., and Takeda, K. (2018). Anomalous sound event detection based on wavenet. In 2018 26th European Signal Processing Conference (EUSIPCO), pages 2494–2498. IEEE.

[106] Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A. Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499.

[107] Almgren, M., Jonsson, E. (2004). Using active learning in intrusion detection. In Proc. of IEEE Computer Security Foundation Workshop, pp. 88–89.

[108] Stokes, J. W., Platt, J. C. (2008). Aladin: Active learning of anomalies to detect intrusion. Tech. rep., Microsoft Research.

[109] Blum, A., Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In COLT' 98: Proc. of the eleventh annual conference on Computational learning theory, pp. 92–100, New York, NY, USA. ACM.

[110] Mao, C.-H., Lee, H.-M., Parikh, D., Chen, T., Huang, S.-Y. (2009). Semi-supervised co-training and active learning based approach for multi-view intrusion detection. In SAC '09: Proc. of the 2009 ACM symposium on Applied Computing, pp. 2042–2048, New York, NY, USA. ACM.

[111] Abeßer, J. (2020). A review of deep learning based methods for acoustic scene classification. Applied Sciences, 10(6).

[112] Rushe, E. and Namee, B. M. (2019). Anomaly detection in raw audio using deep autoregressive networks. In ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 3597–3601.

[113] Papadimitriou, I., Vafeiadis, A., Lalas, A., Votis, K., Tzovaras, D. (2020). Audio- Based Event Detection at Different SNR Settings Using Two-Dimensional Spectrogram Magnitude Representations. Electronics, 9(10), 1593.

[114] Koizumi, Y., Yasuda, M., Murata, S., Saito, S., Uematsu, H., Harada, N. (2020, May). SPIDERnet: Attention Network For One-Shot Anomaly Detection In Sounds. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 281-285). IEEE.

[115] Janjua, Z. H., Vecchio, M., Antonini, M., Antonelli, F. (2019). IRESE: An intelligent rare-event detection system using unsupervised learning on the IoT edge. Engineering Applications of Artificial Intelligence, 84, 41-50.

[116] Kong, Q., Xu, Y., Sobieraj, I., Wang, W., Plumbley, M. D. (2019). Sound event detection and time–frequency segmentation from weakly labelled data. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 27(4), 777-787.

[117] Provotar, O. I., Linder, Y. M., Veres, M. M. (2019, December). Unsupervised Anomaly Detection in Time Series Using LSTM-Based Autoencoders. In 2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT) (pp. 513-517). IEEE.

[118] Koizumi, Y., Saito, S., Uematsu, H., Kawachi, Y., Harada, N. (2018). Unsupervised detection of anomalous sound based on deep learning and the Neyman–Pearson lemma. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 27(1), 212-224.

[119] Koizumi, Y., Saito, S., Yamaguchi, M., Murata, S., Harada, N. (2019, October). Batch uniformization for minimizing maximum anomaly score of dnn-based anomaly detection

in sounds. In 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA) (pp. 6-10). IEEE.

[120] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," ACM computing surveys (CSUR), vol. 41, no. 3, pp. 1–58, 2009.

[121] Droghini, D., Vesperini, F., Principi, E., Squartini, S., Piazza, F. (2018, August). Few-shot Siamese neural networks employing audio features for human-fall detection. In Proceedings of the International Conference on Pattern Recognition and Artificial Intelligence (pp. 63-69).

[122] Antonini, M., Vecchio, M., Antonelli, F., Ducange, P., Perera, C. (2018). Smart audio sensors in the internet of things edge for anomaly detection. IEEE Access, 6, 67594-67610.

[123] Foggia, P., Petkov, N., Saggese, A., Strisciuglio, N., Vento, M. (2015). Reliable detection of audio events in highly noisy environments. Pattern Recognition Letters, 65, 22-28.

[124] Sammarco, M., Detyniecki, M. (2018). Crashzam: Sound-based Car Crash Detection. VEHITS.

[125] R. (2008, September 21). Pack: KITCHEN common sounds. Freesound.Org.

[126] Salamon, J., Jacoby, C., Bello, J. P. (2014, November). A dataset and taxonomy for urban sound research. In Proceedings of the 22nd ACM international conference on Multimedia (pp. 1041-1044).

[127] Aleksandr Diment, Annamaria Mesaros, Toni Heittola, Tuomas Virtanen. (2018). TUT Rare sound events, Evaluation dataset [Data set]. Zenodo. http://doi.org/10.5281/zenodo.1160455.

[128] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," in Proc. Detect. Classif. Acoust. Sce. Events 2018 Workshop, 2018, pp. 9–13.

[129] E. Fonseca et al., "General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline," 2018,arXiv:1807.09902.

[130] Various artificial signal datasets, https://github.com/numenta/NAB/tree/master/data

[131] DCASE2016 Challenge - DCASE.(2016).http://dcase.community/challenge2016/index

[132] Yuma Koizumi, Shoichiro Saito, Masataka Yamaguchi, Shin Murata and Noboru Harada, "Batch Uniformization for Minimizing Maximum Anomaly Score of DNN-based Anomaly Detection in Sounds," in Proc of Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), 2019.

[133] "Series 6000 general sound effects library," http://www.sound-ideas.com/sound-effects/series-6000- sound-effects-library.html, [Accessed: 02- Jan- 2021].

[134] Principi, E., Droghini, D., Squartini, S., Olivetti, P., Piazza, F. (2016). Acoustic cues from the floor: a new approach for fall classification. Expert Systems with Applications, 60, 51-61.

[135] System In Frontier Inc. (http://www.sifi.co.jp/system/modules/pico/index.php?content id=39ml lang=en).

[136] H. Purohit, R. Tanabe, T. Ichige, T. Endo, Y. Nikaido, K. Suefusa, and Y. Kawaguchi, "MIMII Dataset: Sound dataset for malfunctioning industrial machine investigation and inspection," in Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019).

[137] Y. Koizumi, S. Saito, H. Uematsu, N. Harada, and K. Imoto, "ToyADMOS: A dataset of miniature-machine operating sounds for anomalous sound detection," in Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)

[138] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: an ontology and human-labeled dataset for audio events," in Proc. of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp.776–780, 2017.

[139] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, "Freesound datasets: a platform for the creation of open audio datasets," in Proc. of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017), pp.486–493, 2017.

[140] M. M. Jam and H. Sadjedi, "Identification of hearing disorder by multi-band entropy cepstrum extraction from infant's cry," in 2009 International Conference on Biomedical and Pharmaceutical Engineering, 2009, pp. 1–5.

[141] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Receptivefield- regularized CNN variants for acoustic scene classification," CoRR, vol. abs/1909.02859, 2019.

[142] Shruti Jadon, A survey of loss functions for semantic segmentation.

[143] R. Serizel et al., "Acoustic Features for Environmental Sound Analysis To cite this version : HAL Id : hal-01575619 Chapter 4 : Acoustic Features for Environmental Sound Analysis .," 2017.

[144] Yuma Koizumi et al, "description and discussion on Dcase2020 challenge task2: unsupervised anomalous sound detection for machine condition monitoring".

[145] Sohrab Mokhtari, Alireza Abbaspour, Kang K. Yen and Arman Sargolzaei , A Machine Learning Approach for Anomaly Detection in Industrial Control Systems Based on Measurement Data.

[146] Kohavi, R. and Provost, F. (1998) Glossary of terms. Machine Learning—Special Issue on Applications of Machine Learning and the Knowledge Discovery Process. Machine Learning, 30, 271-274.

[147] Guansong pang et al, "Deep Learning for Anomaly Detection: A Review".

[148] Michael Biehl, "Supervised Learning – An Introduction".

[149] Agnieszka Ławrynowicz ,Volker Tresp , "Introducing Machine Learning".

[150] X. Goldberg, Introduction to semi-supervised learning, vol. 6. 2009.

[151] S. Vluymans, "Multi-label Learning," Stud. Comput. Intell., vol. 807, pp. 189–218, 2019, doi: 10.1007/978-3-030-04663-7-7.

[152] Stuart Rosen and Peter Howell. Signals and systems for speech and hearing. Vol. 29. Brill, 2011.

[153] Alla, S., Adari, S. K. (2019). Beginning Anomaly Detection Using Python-Based Deep Learning. Apress.

[154] Rodriguez, M.A.; Kotagiri, R.; Buyya, R. Detecting performance anomalies in scientific workflows using hierarchical temporal memory. Future Gener. Comput. Syst. 2018, 88, 624–635.

[155] Amini, A.; Saboohi, H.; Herawan, T.; Wah, T.Y. MuDi-Stream: A multi density clustering algorithm for evolving data stream. J. Netw. Comput. Appl. 2016, 59, 370–385.

[156] Osvaldo Simeon, "A Brief Introduction to Machine Learning for Engineers"

[157] Shaeke Salman, Xiuwen Liu ,"Overfitting Mechanism and Avoidance in Deep Neural Networks".

[158] J. D. Dignam, P. L. Martin, B. S. Shastry, and R. G. Roeder, "Eukaryotic gene transcription with purified components," Methods Enzymol., vol. 101, no. C, pp. 582–598, 1983, doi: 10.1016/0076-6879(83)01039-3.

[159] Weibo Liua, Zidong Wanga, Xiaohui Liua, Nianyin Zengb, Yurong Liuc, Fuad E. Alsaadid, "A Survey of Deep Neural Network Architectures and Their Applications".

[160] Hanan Qassim Jaleel, , "An overview of Neural Networks and Deep Learning".

[161] Ravil I. Mukhamediev et al , " From Classical Machine Learning to Deep Neural Networks: A Simplified Scientometric Review".

[162] Andy Turner , "Artificial Neural Networks".

[163] Tomasz Szandała , Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks.

[164] Larochelle, H. (2013). Neural networks class [6.2] : Autoencoder - loss function. Hugo Larochelle.

[165] K. Koutini, H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "The receptive field as a regularizer in deep convolutional neural networks for acoustic scene classification," in 27th European Signal Processing Conference, EUSIPCO 2019, A Coruña, Spain, September 2-6, 2019, 2019, pp. 1–5.

[166] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci, and A. Sarti, "Scream and gunshot detection and localization for audio-surveillance systems," in 2007 IEEE Conference on Advanced Video and Signal Based Surveillance, IEEE, 2007, pp. 21– 26.

[167] Lutz, M. (2010). Programming Python: powerful object-oriented programming. " O'Reilly Media, Inc.".

[168] B. Mcfee et al., "Librosa - audio processing Python library," PROC. 14th PYTHON Sci. CONF,no.Scipy,pp.18–25,2015.

[169] G. Maguolo, M. Paci, L. Nanni, and L. Bonan, "Audiogmenter : a MATLAB Toolbox forAudioDataAugmentation,"pp.2–8,2019.

[170] J. Moolayil, Learn Keras for Deep Neural Networks. 2019.

[171] S. Chu, S. Narayanan, C. C. Jay Kuo, and M. J. Matarić, "Where am I? Scene recognition for mobile robots using audio features," 2006 IEEE Int. Conf. Multimed. Expo, ICME 2006 - Proc., vol. 2006, pp. 885–888, 2006, doi: 10.1109/ICME.2006.262661.

[172] E. Cakir, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection," IEEE/ACM Trans. Audio Speech Lang. Process., 2017, doi: 10.1109/TASLP.2017.2690575.