NTNU

Norwegian University of
Science and Technology

FACULTY OF ENGINEERING

DEPARTMENT OF MECHANICAL AND INDUSTRIAL ENGINEERING

Research review on Acoustic Anomaly Detection Techniques for Leakage Detection on Industrial Plants

Author:
Marianne Pettersen

Supervisors:
Christian Holden Gunleiv Skofteland

December 20, 2021

Abstract

This paper is written as part of a specialization project at the M.Sc. Engineering and ICT degree at the Norwegian University of Science and Technology during fall 2021. The project was initialized by and conducted in collaboration with the largest offshore operator of oil and gas in Norway, Equinor ASA. The specialization project lays the groundwork for the master thesis, which will be conducted in the spring of 2022. The objective of this project was to conduct a research review on the topic of using acoustic anomaly detection techniques for finding gas leakages on noisy industrial plants. Gas leakages are a big source of frustration for industrial plant owners as they lead to energy consumption, risk increase for workers, and cause pollution. The gas detection systems of today are based on gas monitors. However, gas monitors do not perform well on detecting tiny leakages and especially have trouble on outdoor plants as gas would rise upwards or be taken away with the wind and therefore create a too low concentration for tripping the alarms. Therefore, Finding a better method for early detection of gas leakages is of great value to industries dealing with gases, especially highly flammable or toxic gases. As sound is emitted from the leakages when the gas exits its compressed container, use of acoustic anomaly detection techniques is possible to use. Basic digital processing techniques have been compared to more modern machine learning techniques regarding the problem of gas leakage detection. An experiment with the utilization of the traditional Root-Mean-Square (RMS) method on a gas leakage dataset was conducted. The results clearly signalized that the rms-energy of leakage signals does not differ enough from the non-leakage signals for the RMS to separate the two classes successfully. An F1-score of only 11.76% and 21.62% was obtained for the classification attempt on signals from a dataset of respectively laboratory recordings and working site recordings. The K-means algorithm and convolutional neural networks are proposed as alternative detection techniques for which further investigations will be conducted in the spring of 2022.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Many of the flaws in industrial processes are found when workers do manual physical checks around the working site, which can reside in remote and challenging to reach locations [26]. Maintaining production and control of oil and gas plants poses a high cost related to attention, travels, and use of resources, and an unnecessary amount of checks is undesirable [26]. An operator is sometimes able to hear abnormal sounds from the machinery and thereby detect abnormalities like gas and other fluid leakages [52]. However, human hearing is limited, and small leakages can go undetected by even the most experienced operator [1]. Suppose humans can hear characteristics of abnormalities in pipes and machinery after some training. In that case, there should, in theory, be a way to extract the same characteristics with the use of software techniques. Today it exists many types of highly technological equipment for detecting sounds that humans cannot even notice [61] [9] [16]. The range of human hearing is 20Hz to 20kHz, and humans can only separate a difference of frequencies between around 200 Hz, while machines and robots do not have these limitations [1]. For the reasons mentioned above, the science topic of detecting leakages in the context of industrial plants and processes has many possibilities.

The reasons why this topic is vital to investigate are numerous. Firstly, an ongoing leakage could have massive physical destructive consequences to machinery and objects in the leakage's surroundings [23] [26], especially if the contained gas that is leaking is toxic or corrosive [49]. Secondly, gas leakages could lead to poisoning if inhaled by workers in the area where an ongoing leakage has released a large amount of toxic gas into the air [49]. Another serious consequence of leakages is pollution to the environment from leakages of fluids that could be toxic, flammable, or corrosive [51] [49]. Thirdly, all the gas is contained for a reason and is either supposed to be used in processes on the working site or be transported to a customer [26]. The loss of gas would therefore result in unnecessary energy consumption and related economic impairment [51] [46]. According to Anthony Schenck et al. [46] one-third of the power consumption in compressed air and gas networks is lost due to leakages, which entails that leakages pose a significant threat to the economic side of the business owners of the gas pipelines. Lastly, leakages of highly flammable substances like hydrogen and methane can lead to accidents, fires, and explosions [2]. In many high-risk workplaces like oil- or melting plants, there are all kinds of equipment and processes that can cause sparks and ignite nearby flammable objects if oil or gas leaks from the processes and gets ignited. One example of a recent incident like this is the fire on Equinor's methanol plant at Tjeldbergodden on the 2nd of September in 2020 [2]. Due to a malfunction in a steam turbine on the plant, chunks of metal were ejected from the turbine and created a hole in a pipe. Lube oil after that leaked from the pipe and got ignited, causing a massive fire. Luckily there were no injuries, but the subsequent investigation concluded that:

> "*... the outcome of the Tjeldbergodden incident could have been very serious, such as leakage and explosion of syngas. The potential included multiple fatalities, as several people were near the building where the fire occurred, at that particular time.*" [2]

Transportation using pipelines has become conventional in the modern industry in most countries, and a vast amount of gas is transported every day [23] [60]. Although energy consumption is the most common consequence of leakages, there can also be other more severe ones as the one just mentioned. In other words, detecting leakages as early as possible is of significant importance for firms and workers in any industry with high-risk environments. Any workplace with machinery and pipes will benefit from a system that detects leakages or other abnormalities early. The longer the leakage or fault goes undetected, the more considerable impact it makes to the machines themselves, the risk to workers, and the energy consumption of the process, eventually leading to economic consequences due to wasted resources [46]. For all of the reasons mentioned above, it is important to investigate newer and better systems for locating and stopping gas leakages early.

There has already been a wast amount of scientific research conducted on the topic of detecting leakages using different methodologies. Moreover, as more and more firms and industries are being digitized, a vast amount of digital data are becoming available. Using methods designed for big data analysis is therefore highly advantageous in this field.

1.2 Problem Description

Different alarm systems are in place in different industries to prevent incidents like the one mentioned above on Tjeldbergodden. A commonly used alarm system is one based on gas monitor instruments. For oil plants, a gas monitor system is a requirement [4]. However, the performance of these systems is varied, and not all types of gas are easily detected by gas detectors. Gas detectors monitor the gas level in the air and warn operators by alarm signals when the concentration of gas is above a specified limit [26]. This limit entails that the leakage must have taken place long enough to reach the concentration necessary for tripping the gas detector alarm system. This is a shortcoming in itself and especially serious in relation to outdoor plants. Most gas rises in the air, and on outside plants, the concentration necessary to trip the alarms may therefore take much longer time to reach as the gas disperse into the sky [26]. Furthermore, especially on windy days, the gas can be quickly removed from the outdoor industrial site, making it hard for the gas detector to detect an ongoing leakage of even large size. Small gas leakages are especially hard for the gas detectors to detect regardless of the weather, and it is, therefore, necessary to have a better leakage detection system [26]. This project aims to find a method that detects all leakages, even the small ones, thereby improving the safety of industrial plants. Luckily, there are other indications of gas leakages besides just the existence of gas in the air. Another characteristic of leakages is the change in air pressure in the area where the gas escapes out of a pipe or container [16]. The difference in pressure around the leakage, where the gas travels from a highly compressed container to the spacey room, will create small vibrations in the air that is known as mechanical waves, or more familiarly known as; sound [46].

The sound changes that alert the operator could stem from different reasons. What the operator perceives as louder noise or faster drumming could be theoretically explained as higher amplitudes, or higher frequency [12]. These features of audio signals could be analyzed in an attempt to find the qualities which are specific for leakages, thus being able to automatically detect them in the future [12]. The hardest challenge is detecting the very small leakages, which have a low signal-to-noise ratio. Therefore, the problem of detecting leakages from sound recordings is the topic at hand in this paper. In this project, an experiment was conducted on a dataset consisting of sound files taken in both a laboratory and an industrial plant using microphones. Clearly speaking, the problem or focus of this paper is:

"**Comparing acoustic anomaly detection approaches, and identifying the most applicable and best performing one for detection of leakages in noisy industrial plants**".

1.3 Paper Structure

This paper is mainly a research review related to the upcoming master thesis regarding the problem of detecting gas leakages in industrial environments. As mentioned, the focus of this paper is to look into the state-of-the-art acoustic anomaly detection (ADD) techniques and make suggestions as to further work for the master thesis. The contents of this paper firstly consist of the introduction, which included the motivation for the topic and the problem description. Thereafter, necessary theory regarding how to process audio signals will be explained in section 2, including feature extraction, analysis methods, and visualization of signals. Furthermore, section 3 will familiarize the reader with the content and features of the dataset, which have been utilized for experimentation. Methods and tools used in practical work will be discussed in section 4. The results produced during the experimentation will be displayed in section 5, including performance evaluation scores. Section 6 will consist of a discussion regarding the findings of the project, as well as a comparison of the different acoustic anomaly detection methods. Lastly, the paper ends with a short summary in section 7, followed by implications as to the focus of further master thesis work on the leakage detection problem.

2 Theory

This section of the paper will explain the theory related to sound and methods for modifying, analyzing, and visualizing sound. Further, some state-of-the-art approaches for acoustic anomaly detection will be introduced.

2.1 Signal processing

Sound is produced by vibration of objects. Vibrations cause air molecules to oscillate, thus creating a change in air pressure, which produces a mechanical wave [1]. A mechanical wave is energy or oscillation that travels through space. These waves carry information about characteristics of the vibration from the sound source, like frequency, intensity, and energy [1]. The sound is captured digitally by taking samples of the air pressure over time. The most common rate at which sound is sampled is 44100 samples per second (44.1kHz) [56]. The digital representation of a sound will be further referred to in this paper as a *signal* or *audio signal*. The process of analyzing features of signals digitally is called signal processing [12].

2.1.1 General theory on sound

A sound can be either periodic or aperiodic. This means respectively that the compression of air pressure repeats regularly or that there is no pattern in the air pressure [12]. A sound wave can be visualized by a sine graph. Typically, periodic sound like clean instrumental keys follows a pattern of sine waves that is easy to recognize. However, in sounds from industrial plants, which involve multiple objects vibrating and creating background noise, the sine waves are usually aperiodic and noisy. In figure 1 one can see the sine wave of a C major key on a piano. By comparing figure 1 to figure 2, the latter representing a leakage sound, one can see a clear difference in pattern, or rather that the first show a pattern while the latter does not. All code produced during this project is developed in the programming language *Python* [44], and relevant code snippets related to the introduced theory will be further discussed during this paper in the appropriate sections. Producing the sinus wave graphs is fairly straightforward, as can be seen from the code included below, where one only needs to plot the signal values for a finite time interval (here, 1000 frames have been used).

```
# Importing necessary libraries
import matplotlib.pyplot as plt
import IPython.display as ipd
import os
from scipy.io.wavfile import read

# Reading data from sound files (.wav format)
sound_file = read(os.path.join(BASE_DIR, leakage_file))
signal = sound_file[1]

#Plotting the Amplitude over Time (1000 samples)
plt.plot(signal[1500:2500])
plt.ylabel("Amplitude")
plt.xlabel("Time")
plt.title("Sinus Wave of small gas leakage")
plt.show()
```

The sound file used in figure 1 was retrieved from the Grand Piano single notes Collection at the Single Focus Website [3]. The sound file used in figure 2 comes from a series of recordings of artificially produced gas leakages taken at Equinor's laboratory named KLAB by a group working for Equinor ASA, for which this project was conducted in collaboration with.

The mathematical definition of a sine wave is shown in equation 1, where A represents the amplitude, f is the frequency, t is time and φ is the phase [53].

$$y(t) = A \sin(2\pi f t + \varphi) \tag{1}$$

The signal waveform can be interpreted, modified, and analyzed by signal processing techniques [12]. A signal can be modified to produce a different sound. One can adjust the frequency to make the pitch higher or lower, and one can adjust the amplitude of a waveform to make it louder or quieter [1]. The higher the frequency is, the higher sound is perceived by the listener, and the larger the amplitude of the wave is, the louder the

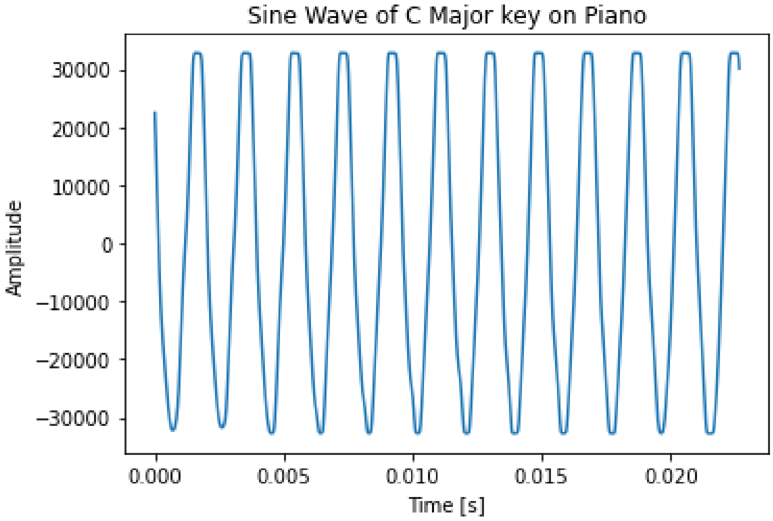


Figure 1: Visualization of the sine wave of a signal consisting of a C Major piano key.

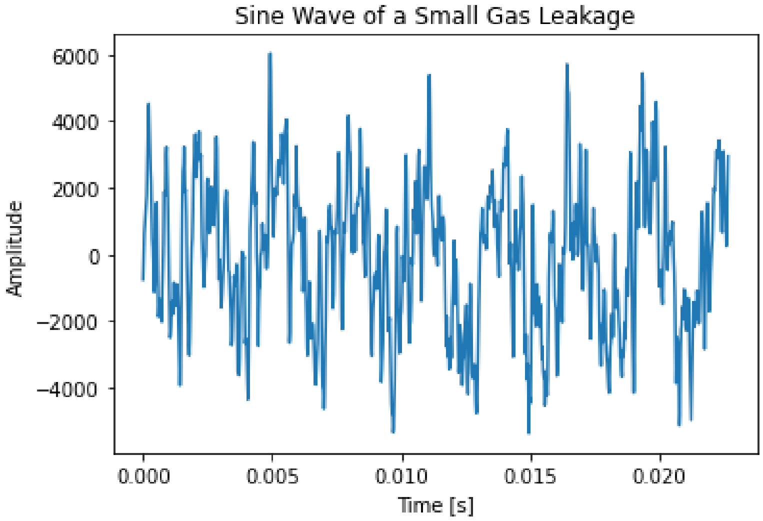


Figure 2: Visualization of the sine wave of signal consisting of a small gas leakage.

sound seams [1]. Amplitude measures the number of perturbations that happen in the air pressure. Higher perturbation means that higher energy is transferred [12]. Energy and frequency are essential features to look at when analyzing sound. It is, therefore, necessary to be acquainted with spectrograms.

2.1.2 Frequency and spectrograms

A spectrogram is a visualization of the signal strength, or "loudness", of an audio signal over time at various frequencies present in a waveform [32]. Spectrograms are two-dimensional graphs, with a third dimension represented by a color bar [32]. Time is represented on the horizontal axis, while the vertical axis represents frequency. The lowest frequencies are at the bottom of the graph, and the highest frequencies are at the top. The color represents the amplitude (or energy) of a particular frequency at a particular time. Dark blue colors correspond to low amplitudes, and brighter colors (closer to red/pink) correspond to stronger amplitudes. In figure 3 one can see a spectrogram of the C major piano key, and in figure 4 on can see a spectrogram of the gas leakage sound. By comparison, the spectrogram of the C major key has only a few frequencies present in the sound wave (the fundamental frequency of 512Hz, and the harmonic doubling frequencies), while the leakage sound has in some degree almost all frequencies, which is a characteristic of a noisy sound wave.

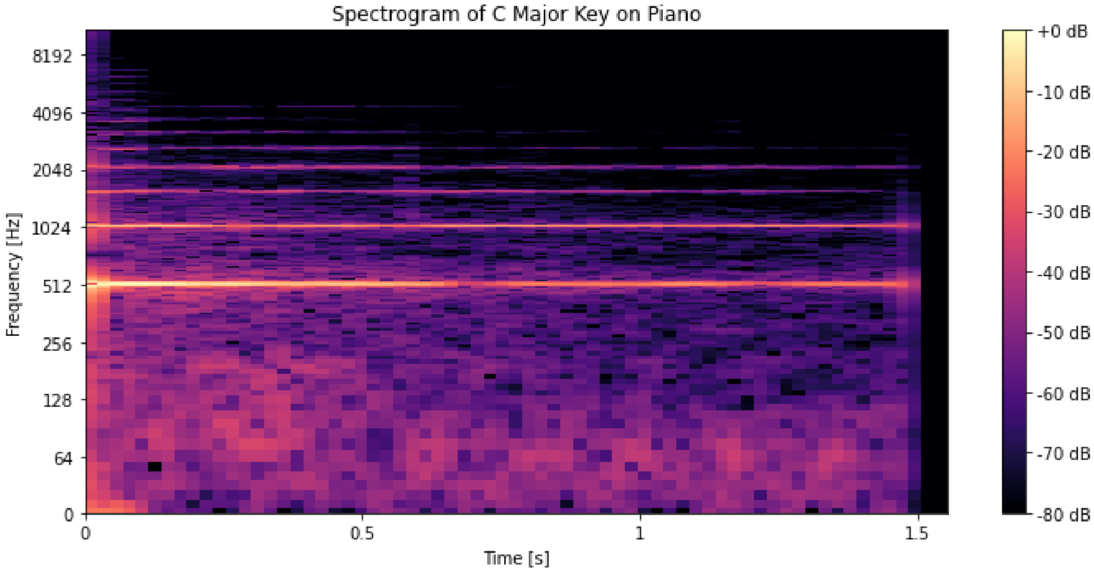


Figure 3: Frequency Spectrogram of a signal consisting of a C major piano key.

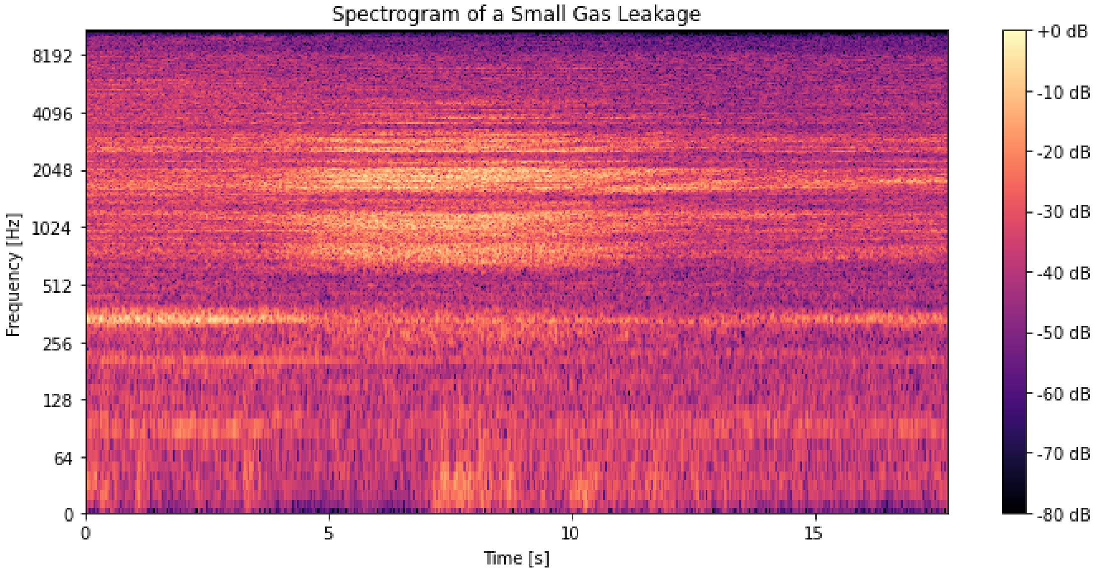


Figure 4: Frequency Spectrogram of a signal consisting of a small leakage.

Spectrograms are used in many scientific applications regarding sound analysis, like classification of animal sounds or speech recognition [32]. They are also used in relation to machine learning, where they, for example, can be fed to a neural network in order to train a model to detect specific sounds [32]. Spectrograms are therefore critical in this project, where they will be analyzed and used in an attempt to detect leakages from sound files. In order to produce the spectrograms in figure 3 and 4 a code snippet was developed with the use of the audio analysis programming library *Librosa* [34] in the programming language *Python*. From the code, one can see that the Short-Time Fourier Transform of the signal has been found using the function *stft()* embedded in the Librosa-library. Thereafter the signal was changed to decibels using function *amplitude_ to_ db()*. Thereafter, the modified signal is plotted.

```
# Importing necessary libraries
import numpy as np
import librosa
import librosa.display

# Taking the Short-Time Fourier Transform of the signal with chosen
↪  hop-length of 512
spec = np.abs(librosa.stft(signal, hop_length=512))

# Converting to Decibel
spec = librosa.amplitude_to_db(spec, ref=np.max)

# Displaying the spectrogram
librosa.display.specshow(spec, sr=sr, x_axis='time', y_axis='log');
plt.rcParams["figure.figsize"] = [10, 5]
plt.rcParams["figure.autolayout"] = True
plt.xlabel("Time [s]")
plt.ylabel("Frequency [Hz]")
plt.colorbar(format='%+2.0f dB');
plt.title('Spectrogram of a Small Gas Leakage');
```

2.1.3 Frequency Magnitude Spectrums

Changing a signal from the time domain to the frequency domain is done by using the Fourier Transform method, thereby creating a spectrum [22]. Spectrums show which frequency components are present in a sound. It is possible to have both the time and frequency representations at the same time, which then gives a time-frequency representation like the one in figure 4. Frequency magnitude spectrums help extract and visualize the main frequencies in a signal [22]. The frequencies are represented on the horizontal axis of the spectrums, while the vertical axis represents the magnitude of each frequency existing in the signal.

By comparing the frequency magnitude spectrums in figure 5 and 6, one can see a clear difference in complexity of the frequency-pattern. The piano sound only contains a few main frequencies (512Hz, 1024Hz, 1536Hz, 2048Hz, etc.). The gas leakage has in some level of magnitude almost all the frequencies visualized in the spectrum (up to around 10 000Hz). Extracting the main frequencies of a signal is instrumental in analysis, and they can be used to compare and recognize similar sounds. The spectrum in figure 5 can, for example, be used to recognize a C major key in the future. For the spectrum in figure 6 it can be difficult for the human eye to recognize a clear pattern. Alternatively, the spectrum can be used as input to a machine learning algorithm which can be trained to recognize similarities between the frequency magnitude spectrums of several gas leakage signals.

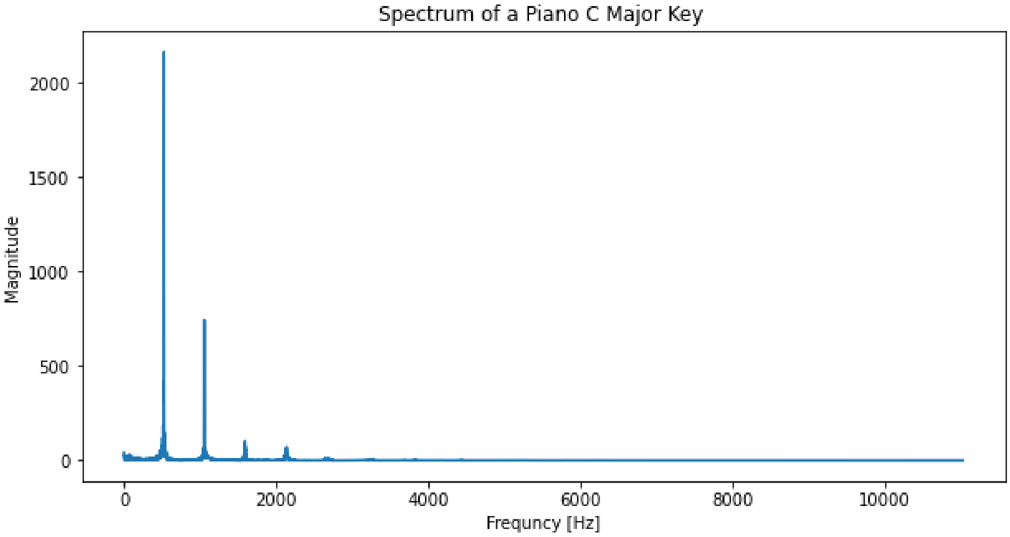


Figure 5: Frequency magnitude spectrum of a signal consisting of a C major piano key.

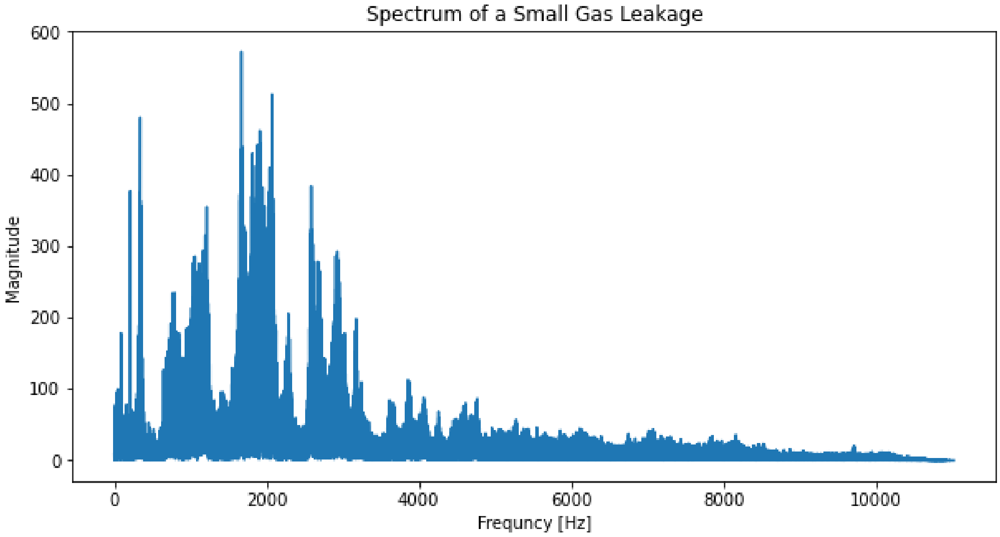


Figure 6: Frequency magnitude spectrum of a signal consisting of a small gas leakage.

2.1.4 Mel-spectrograms

A mel-spectrogram is a spectrogram where the frequencies in a signal is converted to the mel scale. The mel scale is a perceptual scale of pitches where equal distances in pitch sound are equally distant to the human listener [54]. The mel scale mimics how the human ear works. Humans do not hear in a linear way and can only perceive a very small and concentrated range of frequencies and amplitudes. The human ear can detect differences in the pitch of lower frequencies much easier than high frequencies [54]. Therefore, the frequencies in a signal are converted to a log-scale called the mel scale [54]. In other words, the mel scale is used to relate perceived frequencies to the actually measured frequencies. The formula for converting a measured frequency to the mel scaled frequency is shown in equation 2 [54] [35].

$$M(f) = 1127 * \ln(1 + \frac{f}{700}) = 2595 * \log(1 + \frac{f}{700}) \tag{2}$$

Mel spectrograms are often the feature of choice to train deep learning audio algorithms [10]. Mel spectrograms of a signal are created in the following way [35]:

1. Extract the Short-Term Fourier Transform (STFT) of the signal.
2. Convert amplitudes to Decibels (DBs).
3. Convert frequencies to mel-scale.

The last task of converting frequencies to mel scale is conducted in several steps. These steps are, in order, the following [35]:

3.1. Construct mel filter banks by converting the highest and the lowest frequency in the signal to mel frequency (by use of equation 2).

3.2. Create equally spaced-out points between the lowest and highest mel frequency (same number of points as the number of mel bands). Mel bands are a fundamental hyperparameter, and the number is dependent on the problem.

3.3. Convert the points back to frequency and round to the nearest frequency bin.

3.4. Apply the mel filter banks to the spectrogram by matrix multiplication. Mel filter banks are a set of triangular mel-weighted filters. Applying it gives the total energy present in each subband filter. Human ears act as a bank of subband filters (i.e., filterbank).

In figure 7 one can see the results of steps 1-3.2; the projection of the STFT bins onto Mel-frequency bins, which collectively make up the Mel filter bank of a signal. Since both the signal of the piano C major key and the signal of the gas leakage has a sampling rate of 22050 and the number of mel bands equal to 10, the mel filter bank looks the same for both. The code used to create the Mel Filter bank can be seen underneath.

```python
# Importing necessary libraries
import matplotlib.pyplot as plt
import librosa

# Retrieving the sound file from the local directory
signal, sr = librosa.load(os.path.join(BASE_DIR, leak_file))

#Creating the Mel filter banks by using the signals sample rate and
↪  choosing 10 number of mel bands
filter_banks = librosa.filters.mel(n_fft=2048, sr=sr, n_mels=10, fmin=0,
↪  fmax=sr / 2, norm=None)

# Plotting the Mel filter banks
plt.figure(figsize=(10,6))
plt.plot(filter_banks.T)
plt.xlabel("Frequency [Hz]")
plt.ylabel("Mel filter")
plt.title("Mel Filter Bank")
plt.show()
```

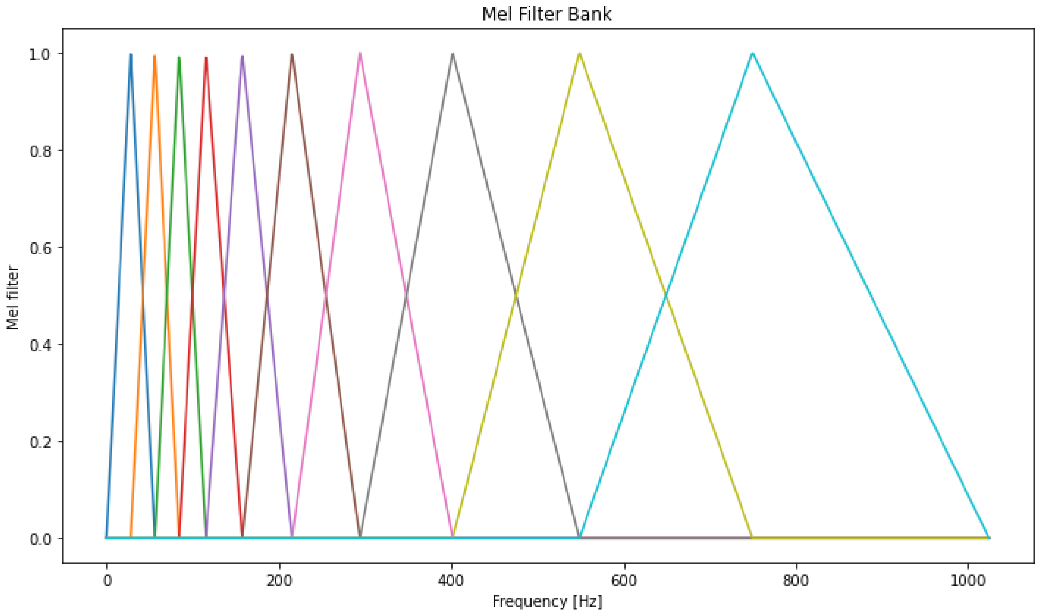


Figure 7: Visualization of the Mel filter bank extracted from a small leakage signal (and C Major Piano key).

All the steps previously outlined are usually performed by software engineers at the same time by use of embedded functions in audio analysis programming libraries such as *Librosa*. An example implementation of creating mel-spectrograms using *Librosa* in the programming language *Python* has been conducted, and the code can be found underneath. It may not be clear to see, but all the steps mentioned above is included in the *melspectrogram()*-function, where one only needs to define the following parameters: the hop length, which is most often 512, and the number of segments to take the Fast Fourier Transform of. Similar to the regular spectrograms, the signal is also here changed to decibels and plotted.

```
# Importing necessary libraries
import matplotlib.pyplot as plt
import librosa
import librosa.display

# Retrieving the features based on the sample rate, and the hop length
mel_spect = librosa.feature.melspectrogram(y=signal, sr=sr, n_fft=2048,
↪  hop_length=512)
# Changing to Decibels
mel_spect = librosa.power_to_db(mel_spect, ref=np.max)

# Plotting the mel spectrogram
plt.rcParams["figure.figsize"] = [10, 5]
plt.rcParams["figure.autolayout"] = True
librosa.display.specshow(mel_spect, y_axis='mel', fmax=sr/2,
↪  x_axis='time');
plt.title('Mel Spectrogram of a Small Gas Leakage');
plt.xlabel("Time [s]")
plt.ylabel("Mel Frequency [Hz]")
plt.colorbar(format='%+2.0f dB');
```

In figure 8 one can see a mel spectrogram of the C major piano key signal. Figure 9 displays a mel spectrogram of the small leakage signal. Both figures were produced by the code snippet above.

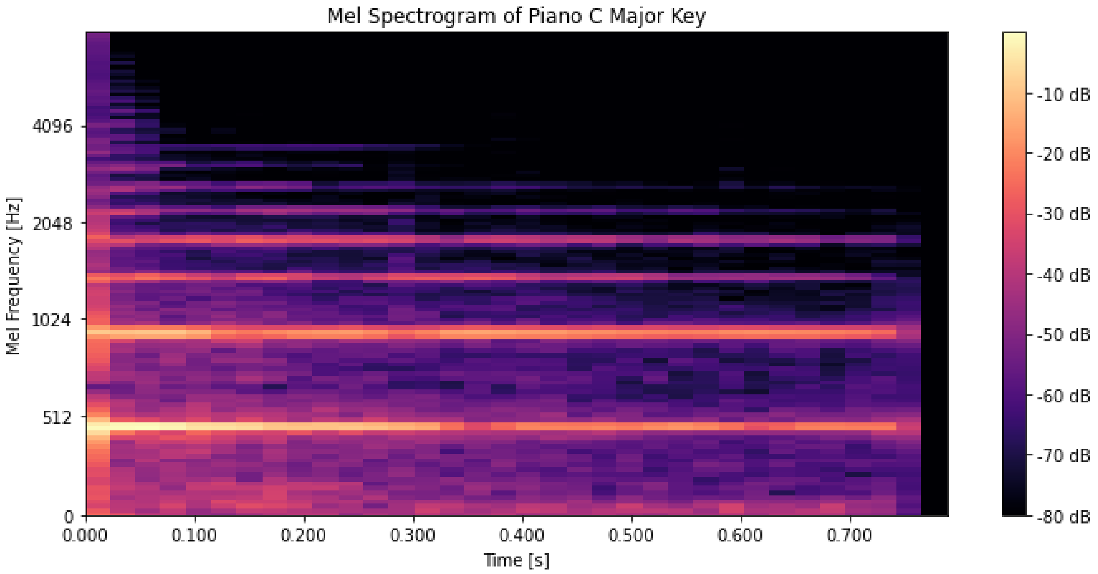


Figure 8: Mel spectrogram of a signal consisting of a C major piano key.

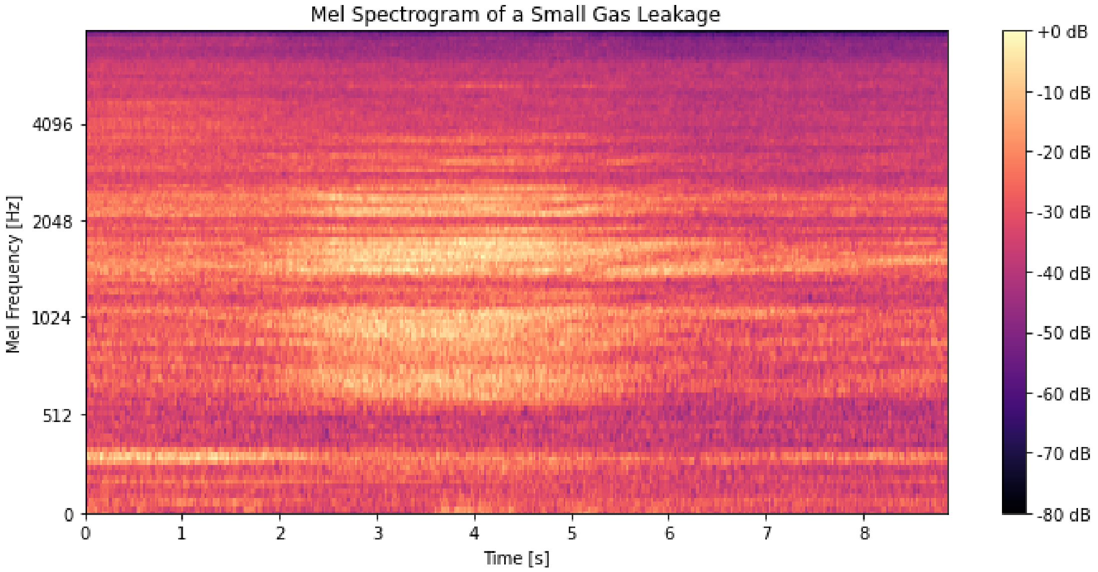


Figure 9: Mel spectrogram of a signal consisting of a small gas leakage.

2.2 State Of The Art - Hardware

There are several ways to detect sound, and different types of equipment can be used for this purpose. Further in this section, typical hardware used for sound detection will be introduced, and a discussion about which of them are most applicable for the industrial environment is also included.

2.2.1 Contact microphones

Contact microphones, also known as Piezo microphones, are microphones assembled onto an object and measure the object's vibration [9]. This kind of microphone has a good sensation and is less impacted by noise than a microphone that is not in contact with an object [9]. However, since industrial plants consist of a huge amount of machines and pipes that could stretch kilometers in length, one can only imagine the vast amount of microphones that would be necessary to assemble to the system. Besides, it is not desirable to assemble something onto the existing structure for security reasons. Another drawback of using contact microphones is that they are greatly affected by the vibrations of the object they are attached to, which could camouflage the sound of a leakage close by the microphone (but not on the attached object) [9]. In other words, contact microphones basically only recognize the sound from the solid object it is attached to. With the existence of many different pipes on the working site, the number of microphones needed would be tremendous, and naturally, therefore also expensive. Instead, a mobile robot with a directional microphone that can stroll freely around the plant is more applicable.

2.2.2 Directional Microphones

Directional microphones are microphones that are most sensitive in one direction [61]. Although it is the sound of a leakage we are interested in analyzing, we have to consider the impact of background noise. A microphone in a room is more impacted by background noise than a microphone located next to the source, like contact microphones. The processes which are being drifted on an industrial plant can be controlled differently each working day, constantly creating changes in the sound pattern of the background noise. Therefore, it can be challenging to define definite characteristics of the background noise at a particular plant, or rather a "normal" condition. This is in spite of the fact that some machines vibrate at a specific frequency. Machines that vibrate will cause other nearby objects to vibrate as well. Therefore, if different equipment is lying around, the background noise's complex sound pattern is often changed. Even though using a microphone that is standing freely in a room will result in a greater impact of background noise than if using a contact microphone [61], it is more applicable and allows for the possibility of mobility. As the directional microphone picks up sound from a specific area or direction, it can find the direction from which a leak is coming. A leakage will seem higher if the microphone is directed at it. Thus, it can be helpful in the task of not only detecting a leak in the area but also locating it. Yukinori Nagaya et al. [37] showed during their work on cavitation detection that directional microphones obtained a greater improvement of sensitivity than non-directional microphones. For all of the reasons mentioned above, the focus of this project will be the use of directional microphones for sound recordings instead of contact microphones or other recording devices.

2.2.3 Cameras

Sound is one aspect of abnormality detection, but small temperature changes are also an aspect. As a leakage provides an expansion to the air around it, it will decrease the temperature, thus indicating abnormalities at the location. Detection of these small changes could be done using thermal and infrared cameras. A research survey on the applications of thermal cameras was conducted by Rikke Gade et al. [16], where the authors concluded that gas leakage detection was a possible application. Sandsten et al. [45] published in 2000 their work on using thermal IR cameras for leakages detection of the highly flammable gasses ethylene and methane, which showed promising results. However, the use of thermal cameras is out of the scope of this paper and will not be further addressed. The reason for this is that there are some practical issues with using cameras on an industrial plant. There is a practical issue related to the need for the cameras to be capsuled and the legal issue of getting ATEX approval for the cameras, which is necessary for being able to operate a high-risk industrial plant. The ATEX directive (2014/34/EU) applies in all workplaces in the EU and EØS, and it describes minimum safety requirements for workplaces containing explosive atmospheres, such as oil and gas plants [4].

2.3 State of the Art - Basic Digital Processing Techniques

Before machine learning became popular, basic digital processing techniques (BDP) were used to analyze sound based on audio features. Audio features can be used to train intelligent audio systems like machine learning models. However, they can also be analyzed by the BDP methods that will be further introduced in this section. This section of the paper will discuss the strengths and weaknesses of BDP and machine learning approaches.

2.3.1 Audio Features

Firstly, some standard audio features of signals will be introduced. Different audio features capture different aspects of a sound. Some audio features commonly extracted and investigated during signal processing tasks are introduced below.

- **Amplitude Envelope**: Measure of the changes in the amplitude of an audio signal over time. The measure is an influential property as it affects our perception of timbre. [55]
- **Zero Crossing Rate (ZCR)**: Measure of the number of times the amplitude of a discrete-time signal in a given time interval changes from a positive to a negative value (and visa versa) [7].
- **Spectral Centroid**: Measure of the shape of the spectrum. The spectral centroid can be viewed as the spectrum's 'center of gravity'. A higher value corresponds to more energy of the signal being concentrated within higher frequencies. [17]
- **Spectral flux/spread**: Measure of the local spectral rate of change. A high value of spectral flux indicates a sudden change in spectral magnitudes. [17]
- **Spectral roll-off**: Measure of the amount of right-skewness in the power spectrum. It points to the fraction of bins in the power spectrum at which 85% of the power is at lower frequencies. [17]
- **Band Energy Ratio**: Measure of the total energy ratio of each frequency band of a signal [36].
- **Root Mean Square Energy (RMSE)**: Measures the average magnitude, or power, of a signal [24].

2.3.2 Mel Frequency Cepstrum Coefficients (MFCCs)

Another audio feature that is widely used for signal processing tasks is the Mel Frequency Cepstrum Coefficients (MFCCs) [60]. The Mel Frequency Cepstrum (MFC) is a representation of the short-term power spectrum of a signal [18]. It is based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency [11]. Mel Frequency Cepstrum Coefficients (MFCC) are coefficients that collectively make up an MFC [18]. The MFCC feature extraction technique basically includes windowing the signal, applying the Discrete Fourier Transform (DFT), taking the log of the magnitude, and then warping the frequencies on a Mel scale, followed by applying the inverse Discrete Cosine Transform (DCT) [18]. The basis of MFCCs is spectrograms, which the Short-Time Fourier Transform (STFT) has been utilized on the audio signal in order to create [11].

For obtaining the mel spectrogram, the mel filter banks have been applied on the STFT [18]. Lastly, by taking the DCT on the mel-spectrogram, one gets the MFCCs [11]. Filter banks can be implemented in both the time domain and frequency domain. For MFCC computation, filter banks are generally implemented in the frequency domain [18].

The cepstral coefficients are referred to as static features, since they mainly contain information from a single given frame [18]. Information about the dynamics of the signal is obtained by computing the first and second derivatives of the cepstral coefficients [18]. The first-order derivative is called delta coefficients, and the second-order derivative is called delta-delta coefficients [18]. In speech recognition, the delta coefficients provide information about the speech rate, while the delta-delta coefficients indicate the acceleration of speech [18].

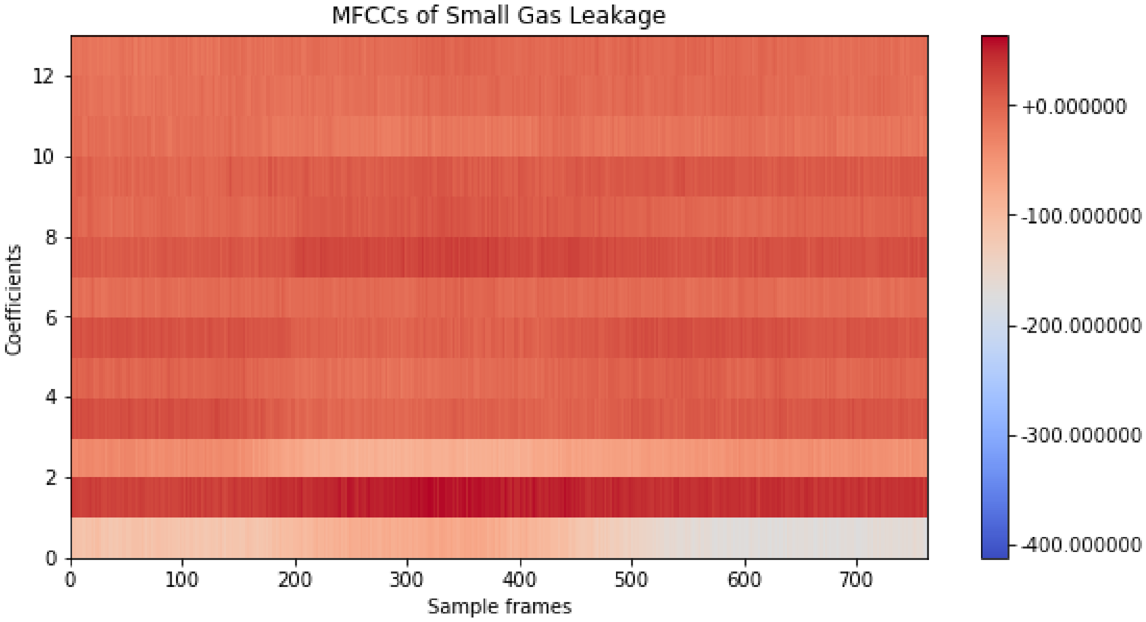


Figure 10: Visualization of the 13 MFCCs extracted from a signal of a small gas leakage.

The shape of the MFCC is a matrix with the number of rows equal to the number of coefficients and the associated number of sample (time) frames of the signal equal to the number of columns. The number of time frames is given by the length of the audio (in samples) divided by the hop length chosen by the data analyst (usually 512 as default) [15]. Extracting the MFCCs from signals is fairly easy as many programming languages have signal processing libraries with embedded functions for this purpose. Underneath, one can see part of the code used to produce the MFCCs of a signal by utilization of the programming language *Python* and the audio analysis library *Librosa*. As one can see from the code, the MFCCs are easily extracted by the *mfcc()*-function in the Librosa library. After extracting these, one simply needs to plot them, as have been done in the latter part of the code. In figure 10 the visualization of the first 13 cepstral coefficients of the MFCC extracted from a gas leakage signal is presented. Normally the first 12-20 coefficients are used for signal analysis. Selecting a large number of cepstral coefficients will result in more complexity in the models. Therefore 13 coefficients is a typical choice.

```
# Importing necessary libraries
import os
import librosa
import matplotlib.pyplot as plt

# Retrieving the sound file from the local directory
sound_file = os.path.join(BASE_DIR, leakage_file)
# Loading the signal and the sampling rate of the leakage-file
signal, sr = librosa.load(sound_file)
# Extracting the first 13 mel frequency cepstrum coefficients using Librosa
mfccs = librosa.feature.mfcc(signal, n_mfcc=13, sr=sr)

# Visualizing the MFCCs of the leakage file
plt.figure(figsize=(10,5))
librosa.display.specshow(mfccs, sr=sr)
plt.ylabel("Coefficients")
plt.xlabel("Time [s]")
plt.title("MFCCs of Small Gas Leakage")
plt.colorbar(format="%+2f")
plt.show()
```

The use of MFCCs alone to detect abnormalities based on inspection and analysis of coefficients is not very successful. However, there is a widespread use of MFCC for the training of machine learning models [60]. Both the numerical matrix representation and the visualization can be used as input to a machine learning algorithm.

2.3.3 Adaptive Digital Filtering

During basic digital processing, some filters may be used to filter background noise from the desired sound in the signal, such as a leakage sound. However, it is essential to find an appropriate filter that does not remove the leakage noise one wants to analyze but instead separates it from the background noise.

The Adaptive Digital Filter (ADF) is a method that generates a band-pass filter with the aim to pass abnormal signals through, thus separating it from the normal signals [6]. An adaptive filter is a system that has a filter and uses a transfer function to control the parameters that are used. The adaptive filter can be a combination of different types of filters, for example, single-input filters, multi-input filters, linear or nonlinear filters, and finite impulse response FIR or infinite impulse response IIR filters [6]. The algorithm used for optimizing the filter parameters is based on minimizing the mean squared error between the output and the desired signal. Two commonly used adaptation algorithms are the Recursive Least Square (RLS) and the Least Mean Square (LMS) [6]. In other words, this method can compare an input signal to the desired signal (which would be a signal without a leakage) and calculate the mean squared error. If the input signal and the desired signal are exactly the same, the method generates an all-pass filter that passes through the entire signal. A signal with a leakage should, in theory, give a high mean squared error, indicating that an abnormality is present. The ADF method creates a filter that passes normal sounds, and if an abnormal sound is present, it creates an abnormal filter [50]. The differences in characteristics of these two types of produced filters can be used to detect abnormalities such as gas leakages.

T. Shindoi et al. [50] published their work on using Adaptive Digital Filtering in comparison with Fourier analysis on diagnosing plant equipment by signal processing. The adaption algorithm they chose to measure the error between the input signal and the desired signal was the Least Mean Square (LMS) algorithm. Two different microphones were used during their study to record sounds, and these were placed 7 meters away from a drain pipe that was used to release steam for simulating leakages. Although they used this type of set-up for recording the sounds, the authors emphasized that the ADF method is easy to use with different equipment. According to the researchers, the noise that the steam leakage produced was barely perceptible to the human ear. The researchers discussed how the method outperformed traditional Fourier. They found that ADF produced more significant differences in normal and abnormal sounds and converged quickly to the optimal solution.

2.3.4 Hilbert-Huang Transform

The Hilbert Huang Transform (HHT) is a basic digital processing technique used for non-stationary and non-linear signals [40]. HHT uses the method called Empirical Mode Decomposition (EMD) to decompose a signal into Intrinsic Mode Functions (IMF) [40]. IMFs are time-varying single-frequency components, where the first IMF component represents the highest frequency in the signal. The IMFs can be used to estimate the variation in magnitude and frequency with respect to time. After the IMFs have been created, the method then takes the Hilbert transform (HT) of the IMFs to create a Hilbert spectrum [23]. The Hilbert spectrum can be used as a statistical tool for distinguishing among a mixture of moving signals [23].

Zhang Shuqing et al. [51] attempted to apply the HHT to the problem of detection leakages in gas pipelines. The researcher's experimental setup consisted of using two dynamic pressure transmitters installed at the two end-positions inside a 20000km long pipeline transferring gas. The researchers then simulated ten artificial leakages in between these two points at 12500km from the first sensor and attempted to use HHT to calculate the location of the leakages based on the point of dynamic pressure signal and time difference. Their results showed that the HHT-method could detect the pressure change accurately and find the location with good precision. Their results are based on sensors inside gas pipelines and do not relate to the use of directional microphones that will be proposed for this project. However, it shows promising results with the use of this traditional basic digital processing technique for localization and is therefore considered relevant to mention.

2.3.5 Root Mean Square (RMS)

Different acoustic signals have different values for audio features like energy and power. The Root Mean Square method (RMS) is a BDP technique that tries to take advantage of these differences by taking the rms-energy values of a signal and comparing it to a predefined rms-threshold in order to alert of any abnormalities [24]. The threshold is defined by calculating the mean and standard deviation of the rms-energy from a dataset consisting of sound recordings of the environment one wishes to observe [24]. After the threshold is defined in an environment, new signals from that same environment can be compared to the threshold in order to alert the operators of a potential abnormality [24]. The rms-values of a signal will trigger an alarm if it exceeds the threshold. In the event of a leakage, the rms-values of the signal will be consistently increasing, thus creating an alert. A detailed description of how this method could be implemented is included further down the paper in section 4.

Dimitrios Kampelopoulos et al. [24] introduced the use of an rms-based approach that relies on monitoring the rms-values of the signal and defining a threshold based on previous rms-values. The researchers used a dataset of noisy recordings from an oil refinery facility of the company Hellenic Petroleum S.A. in Greece and added a few artificial leaks to the audio files. They explained the difference between leakage and non-leakage reasons for an anomaly in rms-values, by which there is a consistent increase in rms-energy for the leakages, thus making it able to distinguish them. In their process, they used a Finite Impulse Response (FIR) filter to remove unwanted dominant frequencies. A moving median calculation was used to eliminate outliers (momentarily increases in rms-energy). They mentioned that to correctly choose the values for the parameters (since it would be different for different pipes), one must either do a careful study of the noise characteristics or by a multi-parametric process. Since a leak is only recognizable when the rms-value becomes higher than the previous windows mean, smaller leaks could go undetected. This is a drawback of the method, as the purpose of classifying audio signals into "normal" or "abnormal" is to be able to detect not only large distinct leakages but also the small ones.

2.4 Strengths and weaknesses

There are certain aspects of basic digital processing techniques that makes using these techniques for sound classification preferable over more recent and complex machine learning techniques. One benefit of using BDP techniques is that they do not necessarily require a large dataset [24]. Many machine learning models use input in the form of images to classify abnormalities [30], but BDP techniques closely monitor features of signals, which one may consider to be more detailed [24]. Some machine learning models are based on the training of only background noise, and after that adding artificial leakages, which could differ from actual leakages [25] [30]. This could, in theory, lead to them not being as well performing on actual leakages and therefore not fit in practice [25]. Some clear weaknesses about BDP techniques is that many of them are outdated and focus on only one or some of the audio features instead of many. This entails that the risk of loss of critical information is higher for these types of methods. Most importantly, it seems that more recent machine learning methods have produced better results than BDP methods, which will be discussed more in the next section.

2.5 State of the Art - Machine Learning Techniques

Besides the traditional digital processing techniques, there has been a broad use of machine learning techniques, both supervised and unsupervised, for audio classification in later years [38]. Machine learning can, in general, be separated into two large categories: supervised and unsupervised learning [38]. In supervised learning, the model is fed with input values and a target value, implying that the data analyst already knows the answer but wants to make a model for future predictions [38]. However, in unsupervised learning, there are no target values. Unsupervised learning is a technique used to search for undiscovered patterns or structures in a large set of unlabeled data with minimal human supervision [38]. The goal of unsupervised learning techniques is to model the underlying structure or patterns in the data in order to learn more about it [38].

In order to classify a sound into the labels "normal" or "abnormal", any machine learning model needs input to train on [38]. Some combinations of the numerical audio features introduced in section 2.3.1 are usually extracted from audio files and fed to a model for training [25]. Also, other types of input can be used to train the machine learning models. For example, audio representations like spectrograms or MFCC-visualizations have been passed as input to models [60]. In addition, several machine learning techniques can be used to classify signals into the classes: "Leakage" or "No Leakage", and some of them will be further explained in this section.

2.5.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN) is a supervised machine learning method that contains three specific types of network layers: a convolutional layer, a pooling layer, and a fully connected layer [39]. The convolutional layers' primary purpose is to detect different patterns or features from an input image, like edges and orientation. The pooling layer simplifies the input and turns the output into a single vector, which is used as an input for the fully connected layer [39]. By sending images such as spectrograms and MFCCs as input to a CNN, the network could learn to classify different classes of sounds, like "Leakage" and "No Leakage" sounds [10].

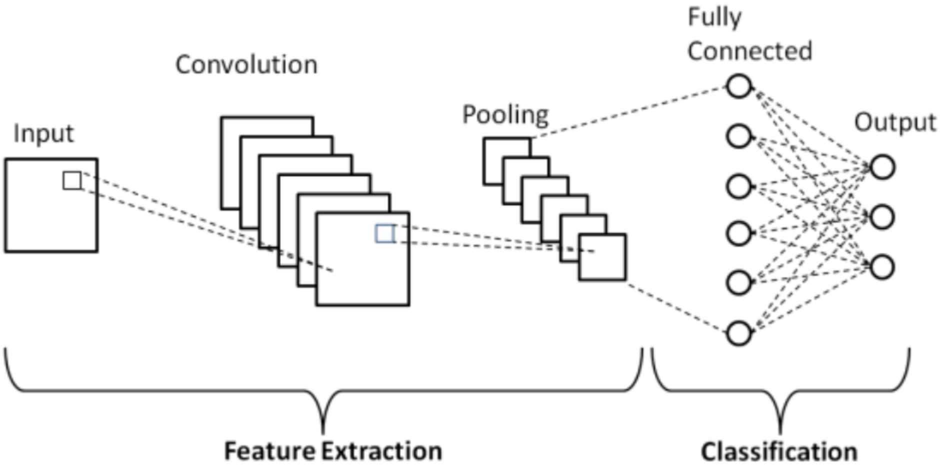


Figure 11: Visualization of the typical architecture of a Convolutional Neural Network. Source: V. H. Phung et al. [43].

For the purpose of leakage detection, the neural network would have to be trained on audio signals with and without leakages. For the network's training phase, each neuron in the layers receives the input (in the form of an image), processes it, and optionally follows it with a non-linear output using an activation function [39]. Within the network, the weights transform the input data. The weights essentially reflect how much influence the input will have on the output. Poor initialization of the weights could be one of several reasons for poor performance [41]. In addition, the choice of an objective function, or loss function, could influence learning speed and the overall accuracy of the method [47].

Zhejian Chi et al. [10] published a study on using logarithmic spectrograms as input to a deep Convolutional Neural Network (deep-CNN) for classification of environmental sounds. Similar to the procedure introduced in section 2.1.4, the researchers used STFT and Mel filterbanks to produce logarithmic Mel-Spectrograms. They developed a CNN with four convolutional layers, one global max-pooling layer, and one fully connected layer, which they trained and tested out on two datasets consisting of different kinds of environmental sounds. Both datasets contained a sampling rate of 44.1kHz, which was mentioned earlier in section 2.1.1 as the most common rate.

One issue with CNNs in relation to dynamic environments, like industrial plants, is that "normal" background noise could be very different from what the networks were initially trained on. Manabu Kotani et al. [28] looked into this issue and presented a study in 2019 on the topic of using neural networks for leakage detection in noisy industrial environments. The research paper addressed the problem regarding dynamic environments surrounding the gas pipes, which lead to different background noises over time. The purpose of their study was to introduce a modular neural network that handles the dynamic environment by adding modules to the network after a given time. The researchers conducted an experiment over the course of 18 months on an oil plant. Their model performs in the following way: It starts with an initial module and classifies the input sound as either "normal", "abnormal" or "unknown", whereas the last class indicates that the input sound is too different from the training data. After some time, a new module that has been training on the "unknown sounds" will be added. A Radial Basis Function (RBF) is a feed-forward function which they use for classification. The Gradient Descent Method was chosen as their activation function and was used in the network for training. Fast Fourier Transform was used as the pre-processing method to examine differences in the power spectrum of the leakage sound and the background noise. A discovery of their research was that the background noise's frequency properties changed over several months, which led to changes in the power spectra. They trained a second module on a background noise dataset but added the leakage sound from the first dataset, and the model classified the sounds for the second term perfectly. So essentially, their proposed method entails creating a neural network with multiple modules, where the modules can be thought of as another neural network. For this to work in the long run, one would therefore need to train another module on the "unknown sounds" when these grow larger than a given amount. It is also essential that the "unknown sounds" that are fed through the new module for training do not contain a leakage that could misguide the module. The constant changes in background noise, and therefore a need to train new modules, could imply a high computational time for training purposes. The researchers compared their model to a Gaussian Potential Function Network (GPFN) to address this issue. The computational efficiency was better for their method than the GPFN, but it may need longer computational time in diagnosis if there are too many "unknowns". However, the total computational cost is lower for their method.

Some drawbacks of neural networks are that in order for them to perform well, they need to be trained on a big dataset with adequate quality [30]. Changes in background noise, which is the essence of the industrial environment, could significantly impact the trained model's performance. However, neural networks are widely used for sound classification tasks and have shown promising results in noisy environments [10]. Therefore, this method seems applicable to this project's introduced problem and will likely be further investigated in future master thesis work.

2.5.2 Support Vector Machines

Support Vector Machines (SVM) is a supervised learning method widely used for pattern recognition and classification problems [58]. The objective of the SVM algorithm is to find a hyperplane in an N-dimensional space, where N is the number of features that manages to distinctly classify the data points in a dataset [33]. The SVM seeks to find a hyperplane that has the maximum distance between data points in the separate classes [33]. The visualization in figure 12 demonstrates how the SVM does the classification using the hyperplane.

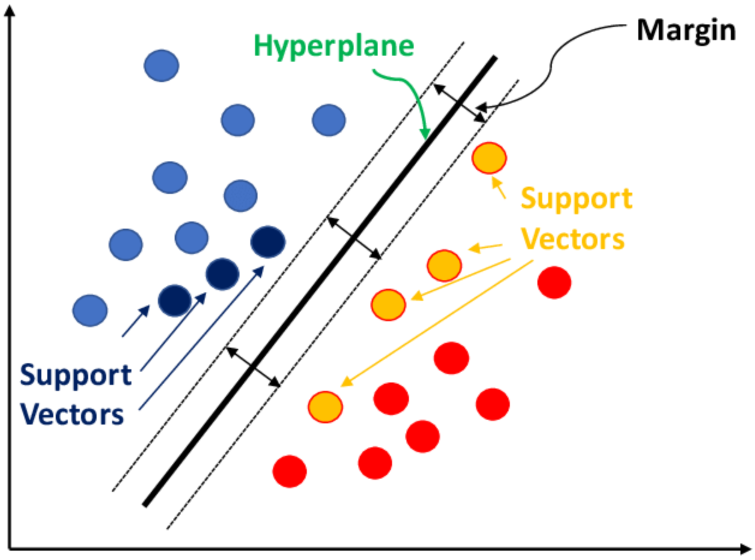


Figure 12: Visualization of a Support Vector Machine with the separation of classes using a hyperplane. Source: Onkar Manjrekar et al. [33].

Tianshu Xu et al. [60] presented in September this year (2021) a pipeline leak identification method for a Spherical Detector (SD) based on combining variational mode decomposition (VMD) and a support vector machine (SVM). The SD gathers the acoustic signals from inside the pipe. The diameter of the SD is smaller than the pipeline's diameter, which makes it easy for it to pass through the pipeline, being pushed by the flow of the liquid inside the pipes (water flow). The Mel frequency cepstral coefficients (MFCCs) were extracted and used to create a characteristic vector for the SVM-based leakage recognition. The authors mentioned a drawback of using a spherical detector: the collision and friction of the SD with the walls of the pipe can camouflage the sound of tiny leakages. Nevertheless, their experimentation using the SD in several pipelines transporting water and oil in China resulted in a classification accuracy of 93% for the SVM.

Rui Xiao et al. [58] introduced in 2019 an acoustic method for natural gas leak detection based on Wavelet Transform and Support Vector Machines (SVM). Wavelet transform was used to de-noise the signals, and a time-frequency feature was used to identify characteristics of different leakage severities. The researchers also used a Relief-F algorithm to quality check the features used in the SVM. The authors argue that SVM has excellent generalization properties, handles outliers and nonlinear boundaries simultaneously, and finds the global optimal solution. Their method received a leakage detection accuracy of 95.60%. Although this paper looks at in-pipe sound by using a microphone inside pipes, it shows positive results of using Support Vector Machines for sound classification and leakage detecting. However, the features or indexes used to establish SVM classifiers are easily affected by the changing operating pressures, which are not sufficiently robust in practical implementations on industrial plants.

2.5.3 Convolutional Autoencoders

An Autoencoder is a neural network that learns to copy its input to its output [48]. Autoencoders are considered to be unsupervised learning techniques because they do not need explicit labels to train on [14]. However, they are self-supervised since they generate their own training labels from the training data [14]. The output layer has the same number of neurons as the input, and the purpose of the algorithm is to reconstruct the inputs by minimizing the distance between the input and output [8]. Autoencoders consist of an encoder and a decoder. The encoder maps the input to a compact lower-dimensional vector, referred to as a latent variable [8]. It does the mapping by using an element-wise activation function such as the Sigmoid function. Backpropagation is used during training to update weights and biases [48]. After that, the decoder maps the latent variable to the reconstruction going from a lower to a higher dimensional space [8]. There are different types of Autoencoders, but Convolutional Autoencoders are typically used in the context of acoustic anomaly detection [8] and take images as input. For the purpose of detecting abnormalities in the form of leakages, the input could be different types of spectrograms. Figure 13 displays the general architecture of a Convolutional Autoencoder.

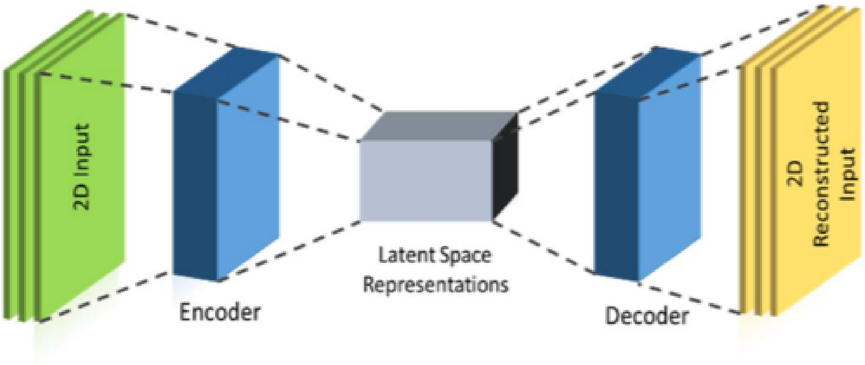


Figure 13: Visualization of the typical architecture of a Convolutional Autoencoder. Source: Shaikh Akib Shahriyar et al. [48].

Taha Berkay Duman et al. [14] used Convolutional Autoencoders (CAE) for feature extraction and as an end-to-end anomaly detector for industrial plants. In the researchers' publication, they compare the performance of a CAE to a One-Class Support Vector Machine (OCSVM). For the implementation they used the machine learning libraries *Librosa* for Mel-spectrograms, *Keras* for CAE and *Scikit-learn* for the OCSVM. The dataset used in their work consisted of a lot of sounds recordings recorded in industrial plants, with normal sounds and abnormal sounds mixed together. These audio files were used during the training of the CAE. Their results demonstrated the superiority of the CAE, where the CAE outperformed the OCSVM-based method. They did not focus on leakages but instead on other abnormal sounds: explosions, fire, and glass breaking. They concluded that the CAE is more successful than other methods when the sounds stem from industrial plants where cutting and welding processes exist.

2.5.4 K-Means Clustering

Clustering is in the category of unsupervised learning methods, and it is a suitable technique for audio classification [42]. Clustering involves dividing a dataset into subsets (i.e., clusters). Each cluster contains data points that are more similar to each other than the data points in the alternative clusters [5]. K-Means is the most common clustering algorithm, and it is efficient for numerical problems. Since the audio features from signals are represented by numerical values, this algorithm is applicable for the leakage detection problem.

The K-Means Algorithm's main objective is to assign each data point in a dataset to one of the k clusters, where k is a number chosen by the data analyst [31]. Specifically, the algorithm's input is a training set of data points $X_1, ..., X_n \in \mathbb{R}^d$, and a number k which represents how many clusters one wants the algorithm to produce [31].

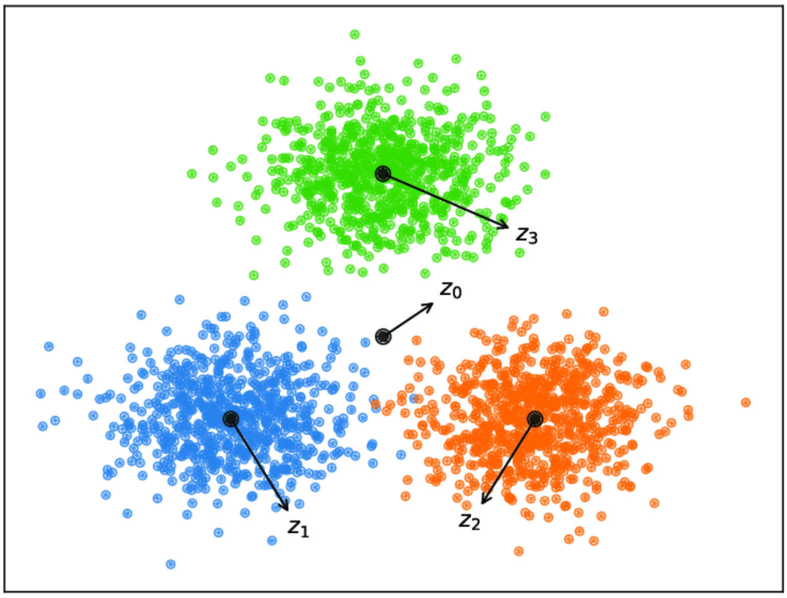


Figure 14: Visualization of clustering with clusters of data points separated by different colors and the related centroids represented by the black middle points (z_1, z_2 and z_3). Source: Liyan Xiong et al. [59].

The K-means algorithm starts with a randomized selection of centroids $m_1^{(0)}, ..., m_k^{(0)}$ [31]. An iterative optimization process then begins. For each iteration i, all data points are assigned to the nearest centroid by use of a distance function, expressed in equation 3, which in turn creates the clusters $C_1^{(i+1)}, ..., C_k^{(i+1)}$ [20]. Thereafter new centroids, $m_1^{(i+1)}, ..., m_k^{(i+1)}$, are defined by calculating the mean vector of each cluster, expressed in equation 4 [20]. This iterative process runs until it converges, meaning the centroids remains the same and the clusters are completely formed [31]. The K-means algorithm is popular because of its simplicity and generality, as one can see from the pseudocode in algorithm 1 [20].

Algorithm 1 K-means algorithm

```
procedure K-MEANS($X_1, ..., X_n$, $k$)    ▷ Input: set of data points, # of desired clusters
    Randomly initialize centroids $m_1^{(0)}, ..., m_k^{(0)}$
    while not converged do
        Assign each data point to the closest centroid by use of equation 3:
            $X_s \in C_k^{(i+1)} \iff ||X_s - m_k^{(i)}||^2 \leq ||X_s - m_l^{(i)}||^2, l = 1, ..., K$    (3)
        Compute new centroids, $m_1^{(i+1)}, ..., m_k^{(i+1)}$, by use of equation 4:
            $m_k^{(i+1)} = \frac{1}{|C_k^{(i+1)}|} \sum_{s \in C_k^{(i+1)}} X_s$    (4)
    end while
    return $C_1, ..., C_k$    ▷ Output: k final clusters
end procedure
```

As mentioned, the K-means clustering algorithm is an applicable method for audio classification and anomaly detection by use of a combination of numerical audio features. Cheng-Yu Peng et al. [42] proposed a system for detection of abnormal vibrating sounds from the milling process of a Computer Numerical Control (CNC) machine. Their system consisted of recordings of the CNC machine by using an Intopic JAZZ-016 mini-PC desktop microphone and classification using the K-means algorithm. Their pre-processing consisted of filtering the sound using a bandpass filter and analyzing the resonant frequency with Fast Fourier Transform (FFT). The frequency results were then passed to the K-means clustering algorithm for classification, which yielded satisfactory results. The researchers' choice of audio features for input to the K-means algorithm was just the frequencies from the signals. Other audio features can also be used as input, like the ones mentioned in 2.3.1, and by using the MFCCs.

The performance of the K-means algorithm is highly dependent on the dataset, or more specifically, the dependencies or similarities between points in the dataset [31]. Furthermore, different results of the algorithm may also appear depending on the algorithm's initialization of centroids, and in the worst case, the result may fall into the local optimum instead of the global optimum [31].

The focus of this project paper is not diagnostics of machines or classification of the degree of leakages in a system based on sound. However, it is right to mention that clustering could be an applicable method for this purpose. Since the data analyst can choose the number of clusters for the algorithm to generate, it is, in theory, possible for the algorithm to separate different severity degrees of leakage in different clusters.

2.6 Strengths and weaknesses

With the use of machine learning techniques, some requirements have to be met in order for them to work properly. For once, there is a need for a large and relevant dataset with good balance and quality to produce a good model performance [43]. Machine learning models produce better results that generalize properly to new cases if the data it has been trained on was of large scale and diverse [43]. In other words, the classification performance is highly dependent on the dataset the model has been trained on. If the collection of a perfectly balanced and large dataset is already taken care of, this should not be a big problem. However, "perfection" is often not reachable. Nevertheless, previous work with machine learning models on acoustic anomaly detection tasks has shown that they can reach well above satisfactory results [42] [58] [60]. They would thereby provide more correct leakage alerts than the gas monitor systems used today.

As previously mentioned, the models could be greatly impacted by the differences in background noise from day to day. There is a risk of the model being trained on background noise (i.e., normal state) that will not be normal on the industrial plant some days, weeks, or months later [28]. The possibility of using a modular neural network and training the model again on new data whenever the differences become large is one possible solution to this issue [28]. Training of machine learning models can require much computational complexity, and therefore take much time to train, especially if the model will create new modules constantly [28]. They may most likely also require a different dataset for each plant where it will be used unless it does an exceptional job on generalizing [28]. There is a risk that large differences in background noise will be falsely classified as leakages since the machine learning technique could consider it to be an abnormality. A possible solution for this issue could be to set limits for how "abnormal" the sound should be for the algorithm to classify it as a leakage or add it to a group of "unknown" sounds. However, it would have to be up to the system's users to decide if the risk of wrongly classifying non-leakage signals as "Leakage" should be prioritized over the risk of letting a leakage go undetected because of a possibly too high limit. As unnecessary checks result in high costs used on travel and resources, it is beneficial to not have false alerts [26], but at the same time, there is a significant cost involved when a leakage goes detected due to absent alerts [46]. The benefit of using machine learning techniques is that one does not necessarily need to understand everything about the data and the characteristics of the sound [10]; the algorithm will learn this by itself. It can be very difficult for humans to examine spectrograms and see the differences between leakage and no-leakage signals. However, machine learning algorithms specifically trained to solve this task can handle the classification of these complex patterns. Machine learning techniques have shown themselves as exceptional for many different science applications in recent years; one just have to make sure to feed them with proper data [8] [10] [28] [42] [48] [58] [60].

3 Data

In order to look further into the problem of detecting leakages from sound, some acoustic data were needed. At the beginning of this project phase, the data used for experimenting was self-produced .wav-files of sounds taken from around the home, like from the microwave, coffee kernel, and kitchen fan. These were mainly used at the beginning of the project to test code and create plots. Further, some recordings of artificial leaks taken at a testing laboratory called KLAB in one of Equinor's working sites were made available. These recordings were made into .wav-files used to investigate leakages' characteristics on spectrograms. The audio file with a small gas leakage that was utilized in the creation of the spectrograms/plots in figure 2, 4, 5, 9 and 10 was taken from the Equinor-recordings. However, a much larger set of data was needed to test out signal processing methods to produce results with appropriate quality.

3.1 Collection of data

With almost all machine learning classifiers, an adequately balanced data set is crucial for achieving optimal performance [29]. In order to have a large and balanced dataset to work with, a public open-access dataset from the *Fraunhofer Institute for Digital Media Technology* was collected [13]. The dataset consists of 5592 files, which contains recordings of different industrial background noises, with and without leakages, and of different types and sizes. Earthworks M30 Omnidirectional Measurement microphones were used as recording devices, which is similar to the earlier mentioned directional microphone that was proposed for this project in section 2.2. The name of the dataset is *IDMT-ISA-COMPRESSED-AIR*, but it will be further referenced to in this paper as the *leakage dataset*.

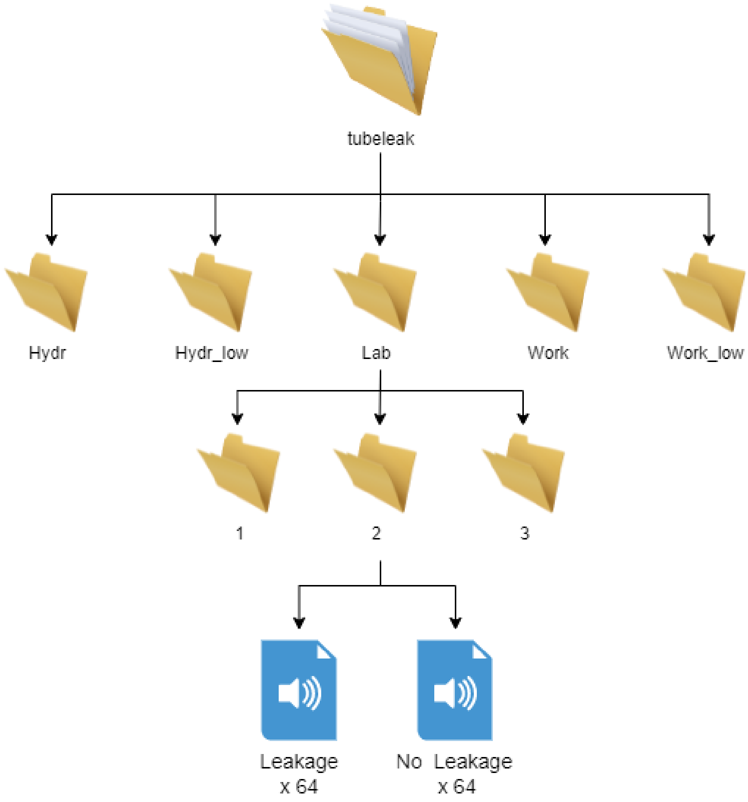


Figure 15: Visualization of the leakage dataset's structure.

3.2 Structure

In figure 15 a visualization of the leakage dataset's structure is presented. The structure consists of to main folders: *ventleak* and *tubeleak*. The folder consisting of recordings from tube leakages is the one presented in figure 15, but the folder with vent leakages looks identical. These main folders have five sub-folders with different types of leakages and environments at focus. These five types are: *Hydraulic machine noise with high volume*, *Hydraulic machine noise with low volume, Laboratory* (no added background noise), *General factory workshop noise with high volume*, and *General factory workshop noise with low volume*. In each of the five sub-folders, there have been three recording sessions labeled in their own folder as *1, 2,* or *3*. In all recording sessions, 64 of the audio files are with a leakage, while an additional 64 are without a leakage.

The dataset was already perfectly balanced, with 50% of the files containing leakages, so no editing of the dataset's content was necessary. Another positive note on this dataset is that the researchers had taken recordings both in a laboratory and on a working site. Recordings from the laboratory make it possible to hear a leakage without any noise, thus creating the possibility to analyze a leakage sound by itself. The recordings on the working site are especially valuable as it contains actual industrial workplace background noises such as welding, cutting, and banging noises. The environment for which the recordings have been taken is highly similar to the environment for which the leakage detection methods discussed in this project have their intended usage. This data, therefore, fits the problem description of this paper very well.

4 Methodology

This section of the paper will address the approach and technologies used for early-stage experimentation on the gas leakage dataset mentioned in section 3. The aim of the experimentation was to implement one of the state-of-the-art methods for acoustic anomaly detection introduced in section 2, and analyze the performance of the method.

4.1 Experimentation of the traditional Root Mean Square Method

Although this paper is mostly meant as a pre-study of leakage detection approaches, some early-stage experimentation has been conducted. In order to have a performance baseline to measure more recent anomaly detection methods against in the future master thesis, some initial experimentation using the traditional RMS method mentioned in section 2.3.5 has been conducted. As the method simply evaluates the rms-values of the signal against a pre-defined rms-threshold, it is a relatively uncomplicated method to test on a dataset for getting a baseline. The RMS method was utilized on the leakage dataset mentioned in section 3.

4.1.1 Technologies and programs

To implement the RMS method on the leakage dataset, *Python* was chosen as the programming language as it contains useful audio analysis libraries. *Jupyter notebook*, which is a web-based interactive computing platform, was used as the integrated development environment (IDE) [27]. Jupyter Notebook is handy for smaller code snippets with the intent to display visualizations and plots, as it combines common developer tools into a single graphical user interface (GUI). As mentioned in section 2.1.4, a common Python-library for audio analysis is *Librosa*. This library contains a function for easily extracting the rms-energy feature of a signal throughout its duration.

4.1.2 Parameter Choice

In the proposed method by Dimitrios Kampelopoulos et al. [24], the rms-threshold is compared to the mean rms-energy over intervals of 1-4 seconds of the new signal. This is to limit the chance of leakage alerts on momentarily increases in rms-energy (i.e., peaks), which other reasons than leakages can cause. As a leakage is continuous until it is stopped, it will create a continuous increase in the rms-energy [24]. Thus, taking the mean rms-energy over a larger time interval will limit the influence of sudden rms-peaks due to noise. Using large intervals could potentially decrease the system's responsiveness, thus increasing the likelihood of letting leakages go undetected. The optimal interval window is not explicitly known, and it can be discussed which undesirable scenario to prioritize; false alerts vs. undetected leakages. During Kampelopoulos and his research partner's study, they weighted detection of leakages more than the risk of false alerts, as they used an interval of 1 second. For these reasons, an interval of 3 seconds is used during this project's implementation in order to prevent undetected leakages, but at the same time aim to limit the number of false alerts.

4.1.3 Threshold Calculation

The main focus of the RMS method is to compare a moving mean of a new audio signal to a predefined threshold. The mean (μ), or average, of a set of values is calculated by equation 5, and during programming it can be found by using the *mean()*-function embedded in the Python-library called *Numpy* [19]. In equation 5, N represents the total number of samples, and X_i represents the rms-value at sample number i [57].

$$\mu = \frac{\sum_{i=1}^{N} X_i}{N} \tag{5}$$

Since all the audio files in the dataset have a duration of 30 seconds, the mean is taken ten times for each file over 3-second intervals. Creating these ten sub-intervals of the signal is not very complicated, but as the code snippet is relatively long, it is placed in appendix A.1. These mean values of rms-energy are then measured against the threshold and will create a leakage alert if the moving mean at some point exceeds it.

The standard deviation (σ) of the rms-values is found by equation 6, where μ represents the mean [57].

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(X_i - \mu)^2} \tag{6}$$

The threshold is easily found by calculating the overall mean and standard deviation of a signal without leakage and defining the threshold value by equation 7. The mean indicates the normal rms-values of background noise without a leakage, which with the standard deviation raises the threshold to a point where it is more likely not to classify a non-leakage signal as a leakage. The reason for using three times the standard deviation is because that is the value proposed by Dimitrios Kampelopoulos et al. [24], which they got adequate results by using. There are many alternative ways of defining the threshold, and in section 6 the issue of finding the optimal threshold will be discussed.

$$Threshold = \mu + 3\sigma \tag{7}$$

As previously mentioned, the threshold for rms-energy is the main focus of the RMS method. As each file in the dataset has a duration of only 30 seconds, it would not be adequate to use only one signal for the threshold calculation. Instead, all the 64 files in one recording session were utilized, adding up to 32 minutes in total. To exploit all files, the mean rms-energy and standard deviation was firstly calculated for each file individually, using functions *mean()* and *std()* in *Numpy*. Thereafter, the average value of all files' mean and standard deviation was found, using the *mean()*-function again on the list of all 64 files' μ and σ, as seen in equation 8 and 9. This was done in order to obtain the overall mean and standard deviation of the whole subset of non-leakage files, i.e., μ_{total} and σ_{total}.

$$\mu_{total} = \frac{\sum_{i}^{N} \mu_i}{N}, N = 64 \tag{8}$$

$$\sigma_{total} = \frac{\sum_{i}^{N} \sigma_i}{N}, N = 64 \tag{9}$$

The function *Calculate_ Threshold()* was created and applied to the dataset in order to determine the threshold. The code for the function can be seen underneath. As can be seen from the code, the function calculates a mean rms-energy for all the 64 non-leakage files in a recording session by equation 5, and then finds the overall one-value mean by equation 8. The same is done for the standard deviation with equation 6, followed by equation 9. Then equation 7 is used to find the resulting threshold. The end part of the function makes a plot will all 64 means, the total mean, and the calculated threshold.

```python
def Calculate_Threshold(files_no_leak, BASE_DIR):
    rms_mean = []
    rms_std = []
    list_x = []
    # For each file in the subset on non-leakage files
    for i in range(len(files_no_leak)):
        file = files_no_leak[i]
        list_x.append(i)
        # Obtaining the signal from the local directory
        signal, sr = librosa.load(os.path.join(BASE_DIR, file))
        # compute magnitude and phase content
        S, phase = librosa.magphase(librosa.stft(signal))
        # compute root-mean-square for each frame in magnitude
        rms = librosa.feature.rms(S=S)
        # Calculating the mean rms-energy
        mean = np.mean(rms)
        # Calculating the standard deviation
        std = np.std(rms)
        # Creating a list of all files' mean and std-values
        rms_mean.append(mean)
        rms_std.append(std)
    # Calculating the average mean and average std
    mean = np.mean(rms_mean)
    std = np.mean(rms_std)
    # Defining the threshold
    threshold = mean+3*std
    # Plotting the threshold with each files' mean rms-energy
    plt.figure(figsize=(10, 6))
    plt.scatter(list_x, rms_mean)
    plt.axhline(y = mean, color = 'green', label='Mean', linestyle = '--')
    plt.axhline(y = threshold, color = 'red', label='Threshold', linestyle
    ↪   = '-')
    plt.xlabel("Sample Files")
    plt.ylabel("Mean RMS Energy")
    plt.title("Mean RMS Energy for Each File in Dataset")
    plt.legend()
    return threshold
```

The non-leakage files from recording session nr.1 in the laboratory folder were utilized firstly. Secondly, the same process was conducted with recording session nr.3 from the working site folder. The result of using this function can be seen in section 5.

4.1.4 Classification of Signals

After the non-leakage signals have defined the threshold, the classification of new signals can begin. As already mentioned, the moving mean of each signal's rms-energy will be compared to the threshold instead of using the rms-energy at all times. In order to conduct this classification process, a function called *Classify_ leak()* was created. The code for this can be found underneath.

```python
def Classify_leak(threshold, files, base_dir):
    leak_alerts = []
    rms_means = []
    list_x = []
    # For each new signal
    for i in range(len(files)):
        list_x.append(i)
        # Leakage Alert is set to "off" in the beginning
        found_leak = False
        file = files[i]
        # Retrieving
        signal, sr = librosa.load(os.path.join(base_dir, file))
        # Compute magnitude and phase content
        S, phase = librosa.magphase(librosa.stft(signal))
        # Compute root-mean-square (rms) for each frame in magnitude
        rms = librosa.feature.rms(S=S)
        # Finding the mean rms-energy of the signal
        mean = np.mean(rms)
        means = get_means(rms)
        rms_means.append(mean)
        # For each mean of the 10 chunks of 3 second intervals
        for mean in means:
            # The leakage alert in "turned on" if the moving mean exceeds
            ↪  the threshold
            if mean > threshold:
                found_leak = True
                break
        # Adding to list to be used for creating confusion matrix
        if found_leak == True:
            leak_alerts.append("Leakage")
        else:
            leak_alerts.append("No Leakage")
    # Plotting each files' mean along with the threshold
    plt.figure(figsize=(10, 6))
    plt.scatter(list_x, rms_means)
    # Adding divider to separate the files that contains a leakage from the
    ↪  others
    if(len(files)>64):
        divider = 64
        plt.axvline(x = divider, color = 'purple', linestyle = '--')
    plt.axhline(y = threshold, color = 'red', label='Threshold', linestyle
    ↪  = '-')
    plt.xlabel("Sample Files")
    plt.ylabel("Mean RMS Energy")
    plt.title("Mean RMS Energy for Each File in Dataset")
    plt.legend()
    plt.show()
    return leak_alerts
```

The *Classify_leak()*-method has the goal to distinguish the input signals into two classes: "Leakage" and "No Leakage". It does the classification based solely on checking whether each signal's moving mean has reached the threshold value. As can be seen from the last part of the code snippet above, the function also creates a plot of all the signals' mean rms-values. It also separates the actual leakage signals from the others to clearly visualize the possible differences between the two classes. In section 6 there will be a discussion about these differences and if they are big enough for an adequate classification to be possible. To evaluate the performance of this leakage classification procedure, some evaluation metrics were utilized.

4.2 Evaluation Metrics

For the performance evaluation, the following metrics were used: Accuracy, Specificity, Sensitivity (Recall), Precision and F1-score [21]. These metrics are defined as the following:

$$Accuracy = \frac{TP + TN}{P + N} \tag{10}$$

$$Sensitivity/Recall = \frac{TP}{P} \tag{11}$$

$$Precision = \frac{TP}{TP + FP} \tag{12}$$

$$Specificity = \frac{TN}{N} \tag{13}$$

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{14}$$

In the equations above, TP and TN respectively stands for *True Positive* and *True Negative*, and it means that a sample, in this case, an audio signal, have been correctly classified as "abnormal" or "normal" (in this case "Leakage" or "No Leakage"). The symbols for *Positive (P)* and *Negative (N)* simply means the total amount of samples classified as "normal" and "abnormal". All of these metrics indicate the performance of the method. Precision indicates how many possible false leakage alerts can happen, and Recall/Specificity shows how many "abnormal" sounds, in this case, *leakages*, could go undetected.

One drawback of using the F1-score is that precision and recall have equal importance. By the assumption that the RMS method alone result in alerts or not, the consequences of failing to alert a leakage due to false negatives are greater than the risk of unnecessary in-person checks due to false positives. The recall is, therefore, a more significant metric for this problem. However, both recall and precision in collaboration with specificity and accuracy are included to compare the different metrics' results. The code used to retrieve these evaluation metrics is included underneath. As can be seen from the functions in the code snippet, the exact implementation of equations 10, 11, 12, 13 and 14 has been conducted.

```python
# Function for calculating accuracy
def Get_Accuracy(TP, TN, P, N):
    accuracy = (TP+TN)/(P+N)
    return accuracy

# Function for calculating recall
def Get_Recall(TP, P):
    recall = TP/P
    return recall

# Function for calculating specificity
def Get_Specificity(TN, N):
    specificity = TN/N
    return specificity


# Function for calculating precision
def Get_Precision(TP, FP):
    precision = TP/(TP+FP)
    return precision

# Function for calculating F1-score
def Get_F1(Precision, Recall):
    F1_score = (2*Precision*Recall)/(Precision+Recall)
    return F1_score
```

5 Results

During the experimentation of the RMS method, some results have been obtained. These results will be presented and examined in this section. The different types of results are; calculated threshold values, confusion matrices, and the evaluation scores found by the metrics introduced in section 4.2.

5.1 Threshold Values

The threshold value for the laboratory recordings was found to be **0.0094**, while for the recordings from the working site, the threshold value was **0.1578**. Calculation of the threshold from another recording session conducted in a working site (session nr.2) produced a threshold value of **0.1019**. This entails that the average rms-energy of signals without leakage differs depending on the environment. It is not surprising that the laboratory signals have less rms-energy as it is supposed to have approximately zero background noise (i.e., silent signal). These large differences in calculated threshold values reveal the fact that various background noises have a significant impact on the rms-energy. As follows, it is not only the existence of a leakage that creates increased rms-energy. Consequently, it could be the case that using only rms-energy as a feature for detecting leakages could be insufficient. A deeper abbreviation of this finding is included in section 6. In figure 16 one can see a visualization of the mean rms-value for all 64 no-leakage files from laboratory session nr.1. The green line represents the total mean of all the files combined. As previously specified, this total mean plus three times the standard deviation produced the threshold, which is represented by the red line on the plot. In figure 17 one can see the same visualization for all 64 no-leakage files from working site session nr.3.

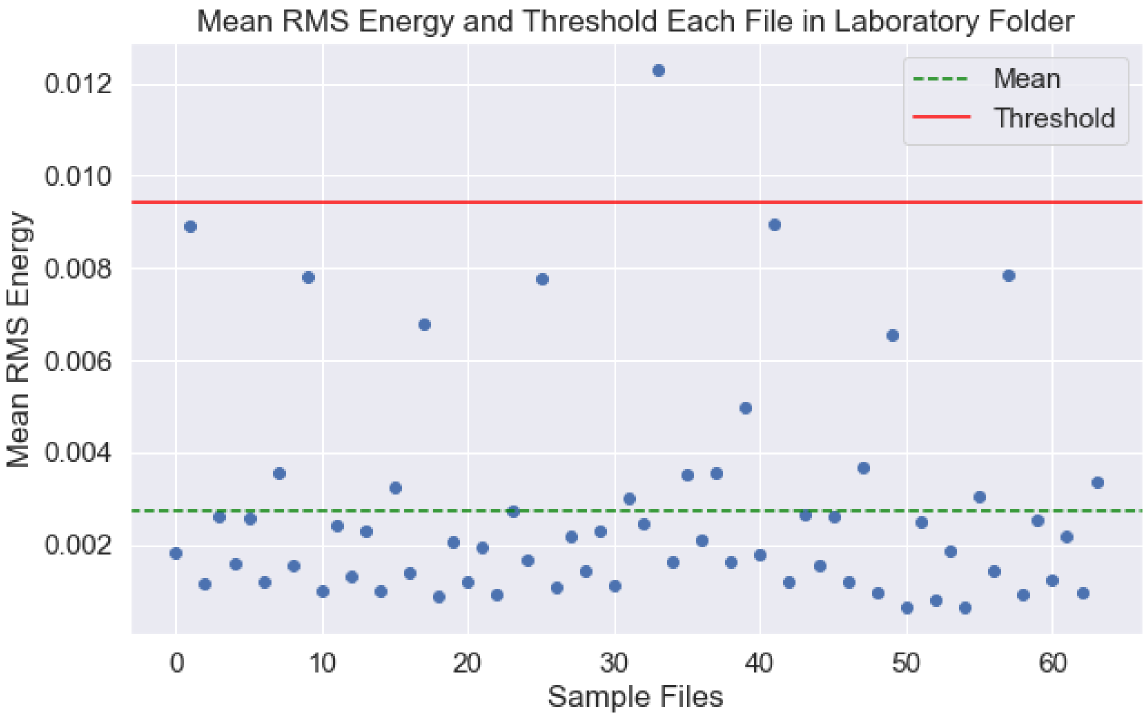


Figure 16: Visualization of the calculated Threshold and Mean rms-energy of the 64 audio signals with only background noise, from the laboratory recording session nr.1.

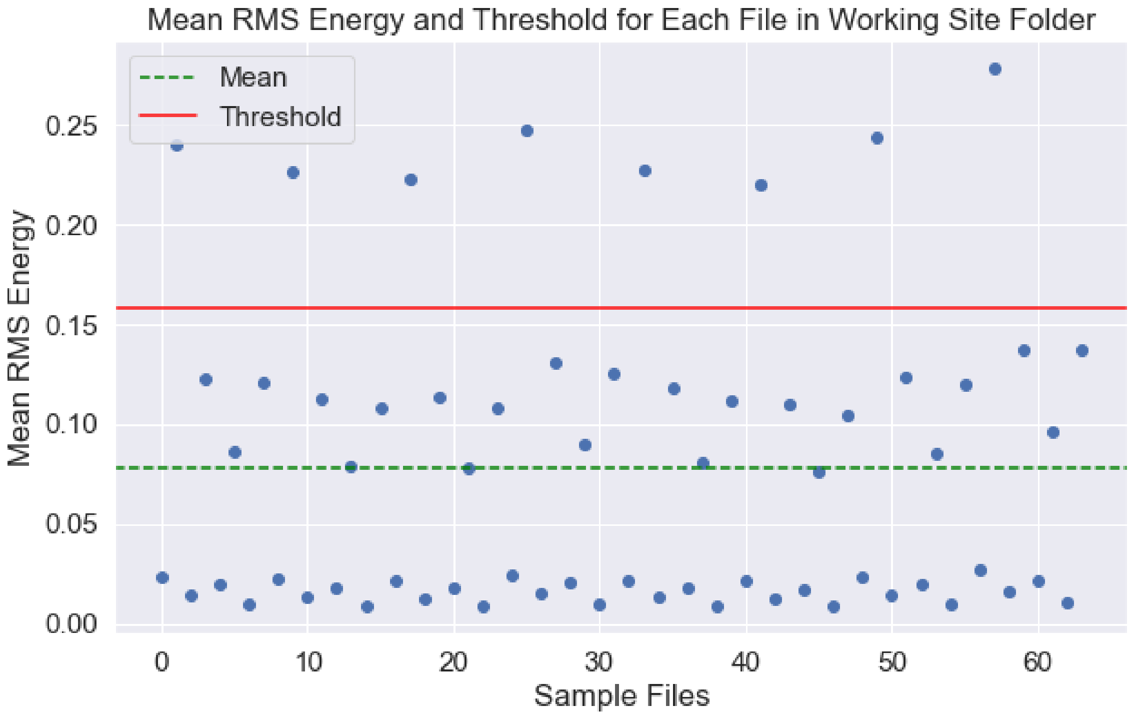


Figure 17: Visualization of the calculated Threshold and Mean rms-energy of the 64 audio signals with only background noise, from the working site recording session nr.3.

The total mean value for the laboratory session was **0.0027** and the average standard deviation was **0.0022**. This mean value can be clearly seen in figure 16 by the green dotted line. For the working site session, the total mean was **0.0781**, while the average standard deviation was **0.0265**. Likewise, for the working site session, the mean value is displayed on figure 17 in the green dotted line.

5.2 Classification of leakages

As the threshold was calculated for laboratory session nr.1 and working site session 3, the classification performance was tested out on other recording sessions, namely laboratory session nr.2 and working site session 1. The result of the classification of the laboratory signals based on the laboratory threshold can be seen in the confusion matrix in figure 18. The most important value to point attention to is the value in the upper right corner of the confusion matrix, which shows the false negatives (FN). A high number of leakages seem to have been wrongly labeled as "No Leakage" by the RMS-method. The value at the upper left represents the true negatives (TN), and the value shows that the total rightfully labeled leakages was no more than 8 out of 64 actual leakages in the dataset.

The values from the confusion matrix in figure 18 gives a recall score of disappointingly **12.5%** for the laboratory signals. The accuracy is only at **54.69%** and is elevated by the fact that almost all the signal without leakage is correctly classified, giving a high amount of true negatives (TN). The total F1-score is **21.62%** (see table 1), which is not a satisfactory result for being able to use this method as the sole method for leakage detection.

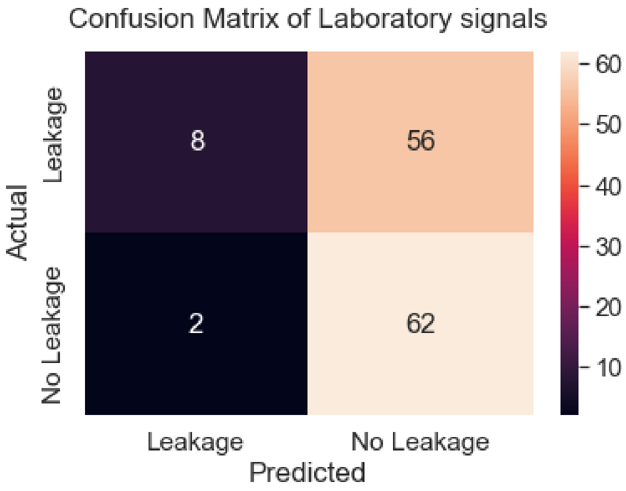


Figure 18: Confusion matrix for the RMS method on signals recorded in a laboratory.

The classification results of the signals from working site session 1, utilizing the working site threshold, are displayed in the confusion matrix in figure 19. The performance of the RMS method on signals with background noise was even worse, as it only managed to classify 4 out of 64 signals correctly as "Leakage". The high number of false negatives (60 out of 64 possible) gives an extremely low recall score of **6.25%**. Likewise, as the laboratory session, the classification of this session's signals gave a high number of true negatives. Hence, a similar accuracy score of **53.12%**. Looking at the low F1-score of only **11.76%** in table 1, it is clear that the RMS method performs even worse in environments like industrial working sites with a lot of background noise.

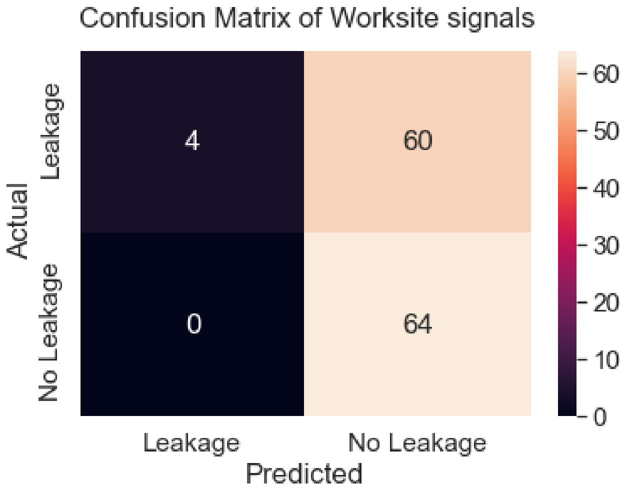


Figure 19: Confusion matrix for the RMS method on signals recorded on a working site.

The scores of all evaluation metrics introduced in section 4.2 is displayed in table 1 for both classification trials. The results, as well as possible improvements to the method, will be further discussed in section 6.

Table 1: A table displaying the results of different evaluation metrics for the two recording sessions.

Recording session	**Accuracy**	**Recall**	**Specificity**	**Precision**	**F1-score**
Laboratory 2	54.69%	12.50%	98.88%	80.00%	21.62%
Working Site 1	53.12%	6.25%	100.00%	100.00%	11.76%

6 Discussion

6.1 Performance Evaluation

Examination of the scores in table 1 reveals that the RMS method performs well on classifying signals without leakages correctly as the specificity score is **98.88%** for the laboratory dataset and **100.00%** for the working site dataset. This could be explained by the fact that the threshold was calculated by non-leakage signals, which provided a threshold higher than the mean rms-energy (in addition to three times the standard deviation) of no-leakage signals. In other words, the threshold was designed to fit the non-leakage signals and should therefore perform well on classifying true negatives. Nevertheless, the high specificity scores do not make up the remarkably low recall and F1-scores. The evaluation scores shows conclusive evidence that the RMS method alone is not adequate for leakage detection in this type of dynamic environment with considerable changes in background noise.

6.2 Interpretations of results

The RMS method is based on the assumption that the rms-energy of a signal with leakage will be significantly higher than a signal without one [24]. However, looking a the plots in figure 20 and 21, one can clearly see that the rms-energy is somewhat the same, and also under the defined threshold, for signals both with and without leakage. One can also see that the rms-energy has one peak exceeding the threshold in both signals. If the method did not use a moving mean and only looked at the rms-value at any given time, both of these signals would have been labeled as a "Leakage". The green line on the plots represents the moving mean, i.e., the mean for every 3-second interval of the audio signal.

To visualize the comparison even more clearly, figure 22 shows the mean rms-energy of all 128 audio files in the *lab*-folder. This folder was recorded with no background noise and should, in theory, show a big difference when a leakage is present. The purple line indicates the separation between the files without a leakage (on the left side) and the files with a leakage (on the right). The same visualization is included for the *work*-folder in figure 23. For this method to work properly, all the 64 audio signals on the right side of the purple line should have had a greater mean-value than the signals on the left side, given that the basic assumption was correct. However, by these results, it is clear that no matter what the threshold value was, the result of the system would be that a way too high number of leakages would go undetected.

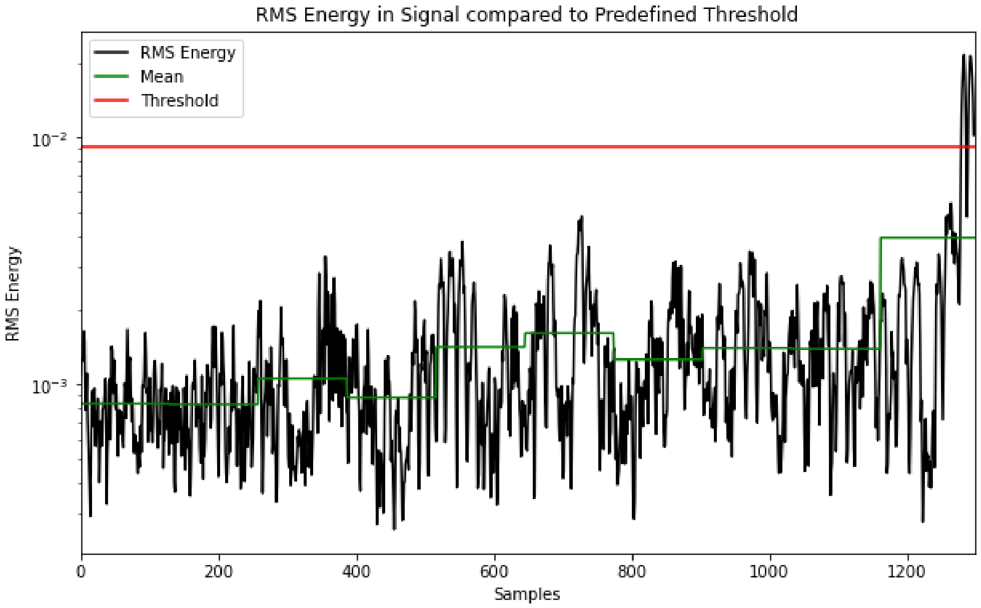


Figure 20: Plot of the rms-energy in a signal containing a gas leakage recorded in a laboratory.

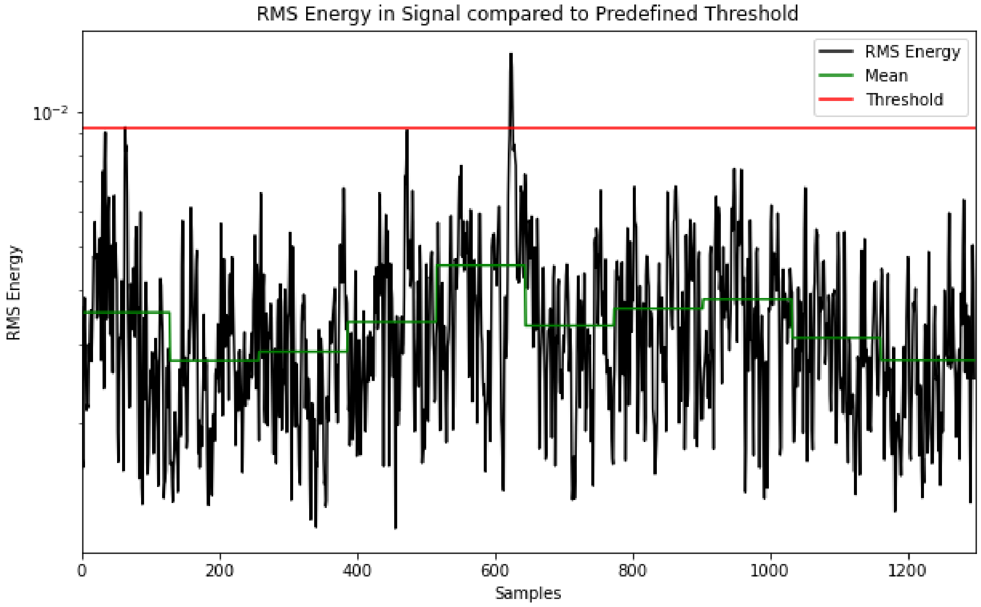


Figure 21: Plot of the rms-energy in a signal containing no background noise or leakage, recorded in a laboratory.

Multiple audio files with different background noises and leakage types were used to assess the RMS method during this project. In Kampelopoulos's study, they only used a single no-leakage recording with a duration of 3.5 hours for finding the threshold, and then compared the threshold to another dataset of recordings of two artificial leakages [24]. When these artificial leakages were recorded at another time and potentially at a different place, it is not very surprising that they produced a significant change in rms-energy, thereby providing good results. However, it is desirable to have a system that detects leakages in dynamic environments. The researchers in the mentioned study have not prioritized this as they used only one recording in one environment.

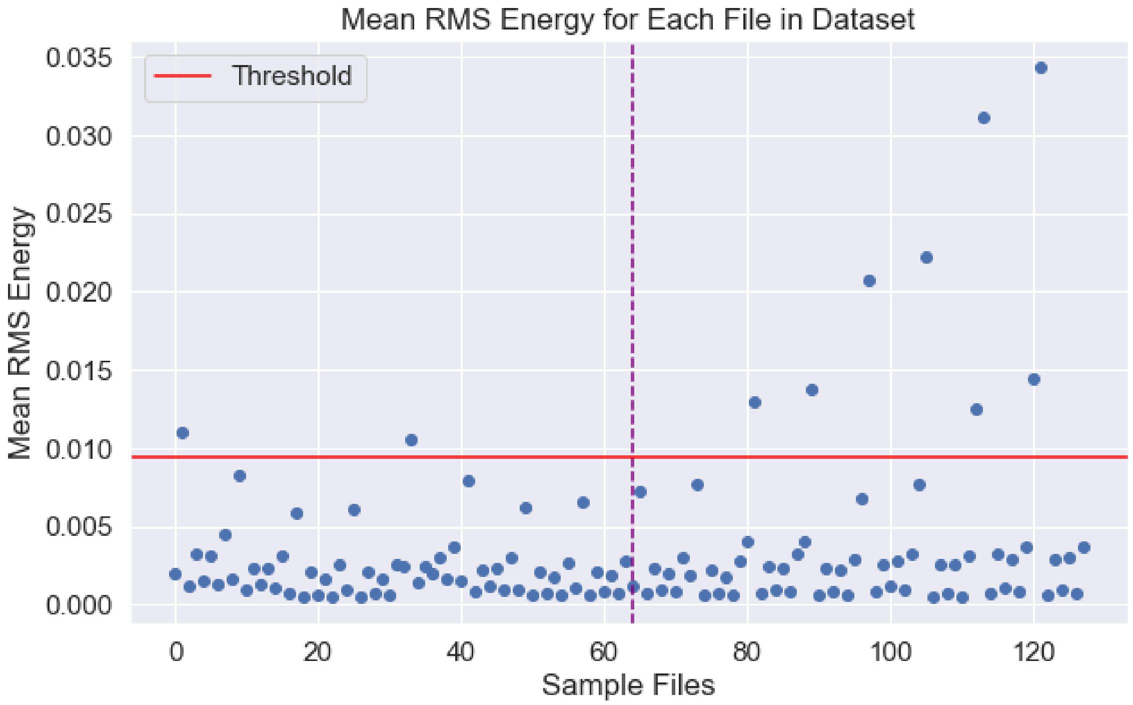


Figure 22: Visualization of the mean rms-energy in each of the 128 files from the laboratory recording session nr.2.

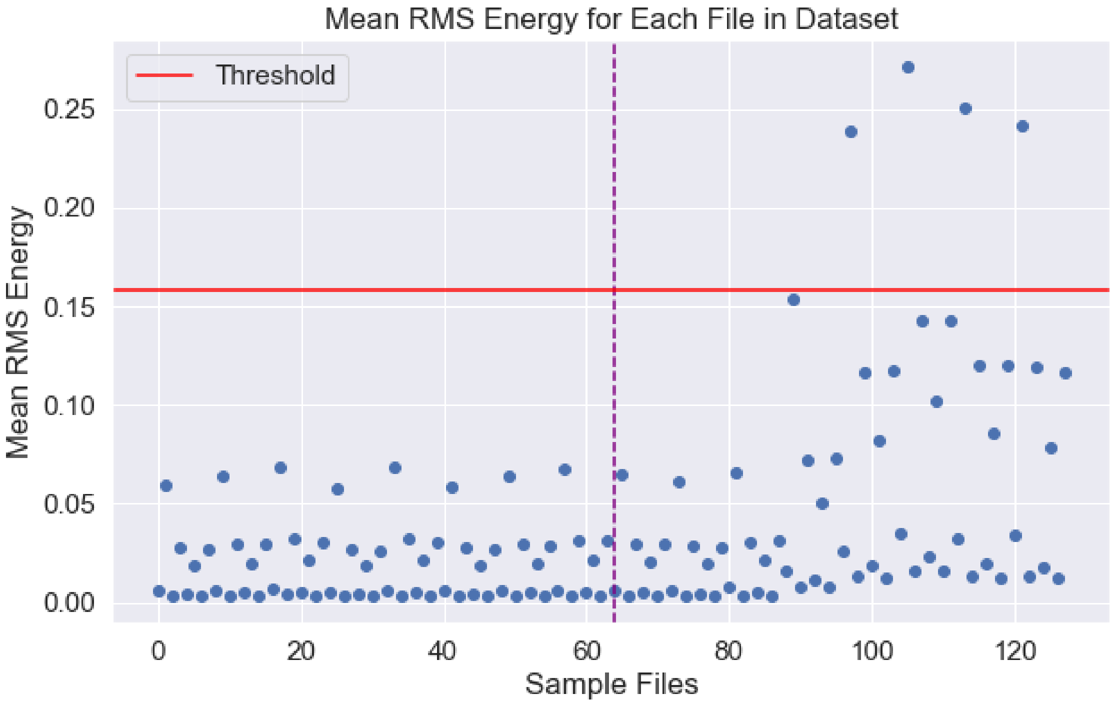


Figure 23: Visualization of the mean rms-energy in each of the 128 files from the working site recording session nr.1.

6.3 Possible Improvements

Regarding the matter of the threshold calculation, it is evident by looking at figure 22 and 23 that using a different equation for calculating the threshold would not have mattered much. As the rms-energy of the leakages is not that significantly different from the non-leakage, a different threshold would still not separate the two classes ("Leakage" and "No Leakage") adequately. Through examining figure 22 one can determine that lowering the threshold to, for instance, 0.005 instead of 0.010 would result in more true positives, but at the same time also result in more false positives. Lowering the threshold in figure 23 to a value around 0.08 would provide a greater number of true positives while simultaneously keeping the false negatives at a stably low number. Alternatively, the calculated threshold for the working site session nr.2, which was found to be **0.1019**, would have provided a few more true positives. Doing this would have produced slightly better performance scores. However, the method would still not be satisfactory as both classes' rms-energy is too similar, and the full separation of classes is impossible.

6.4 Choice of alternative methods

There is a slightly higher rms-energy in a few of the leakage signals than the non-leakage ones, which implies that some types and degrees of leakages will be possible to separate. The fact that the rms-energy is different for some of the leakage signals suggests that the rms-energy should not be completely disregarded as a useful audio feature for leakage detection. Alternatively, this feature could be used for leakage classification in combination with some of the other audio features. For this reason, the use of the rms-energy will be considered looking into along with the audio features mentioned in 2.3.1.

After a comparison of basic digital processing (BDP) techniques compared to machine learning techniques, and at the same time judging the results of the experimentation of the RMS method, it is implied that the BDP is perhaps too outdated. After considering the strengths and weaknesses of these two types of techniques, which were discussed in section 2.4 and 2.6, the decision to use machine learning techniques for further research has been made. The methods that show the greatest promise is the K-means clustering algorithm and Convolutional Neural Networks. These two methods use different types of input. The K-means algorithm uses numerical input, which makes the use of a combination of the numerical audio features from 2.3.1 a possibility. The Convolutional Neural Networks usually train on images in the form of spectrograms, MFCC-visualization, or frequency magnitude spectrums [10]. It could be interesting to compare the results of these different inputs on the performance of the CNN after the different training sessions. The other mentioned machine learning techniques, namely Support Vector Machines and Convolutional Autoencoders, are also good candidates for acoustic anomaly detection. However, the author has landed on using K-means and CNN, as they have displayed noticeable results and could also potentially be used for classification into different degrees of leakages.

7 Conclusion

7.1 Summary

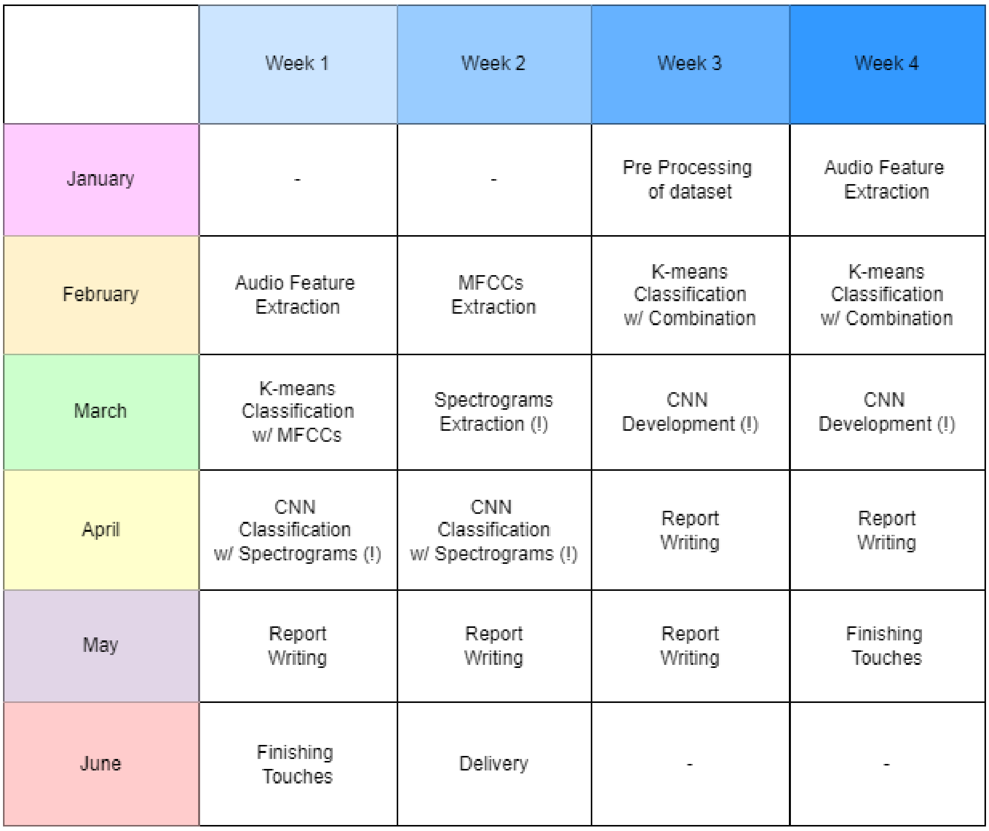
There is a big focus on safety measures in industries that deal with high-risk substances like oil, methane, hydrogen, and other gases. Leakages of highly flammable substances can lead to massive economic and environmental consequences and increase the safety risks of workers at an industrial plant. The problem of detecting gas leakages as early as possible is therefore of great value to industries dealing with these substances. Detection of tiny gas leakages is especially important. If they go undetected for too long, they will lead to high energy consumption losses and possible poisoning of workers if the gas is toxic. Therefore, the objective of this paper has been to do a research review on the possibilities of using the sound that gas leakages create when exiting their containers to detect them as early as possible. Useful theory on sound, signal processing, and different ways of visualizing audio features was presented in section 2. Further down in section 2.2 the State-of-the-art hardware used for the purpose of recording or capturing sound was introduced, where the emphasis of the practical use of directional microphones as supposed to contact microphones or cameras was discussed. Traditional basic digital processing techniques were mentioned, and different commonly used audio features used during signal processing tasks were presented in section 2.3. Some deeper explanation of the components of state-of-the-art machine learning approaches for anomaly detection, and sound classification was conducted in section 2.5. The structure and features of the *IDMT-ISA-COMPRESSED-AIR*-dataset utilized for experimentation with the traditional RMS method was gone through in section 3. During section 4 the methodology used during experimentation was explained in detail. The results of utilizing the RMS method on the gas leakage dataset were presented in section 5. The evaluation scores were discussed in section 6. The scores demonstrated that the RMS method does not work sufficiently on acoustic anomaly detection in dynamic environments like working sites as the rms-energy is too similar for the signals with and without a leakage present. However, there is a slight increase in rms-energy for some of the leakage signals. This implies that there is a possibility of using the rms-energy feature in combination with other audio features for classification purposes, and this should be investigated further.

7.2 Further Work

For further work in the master thesis, the focus will likely be on machine learning techniques. One point of worry for machine learning approaches based on supervised or unsupervised learning is the availability of an adequate dataset to train models on. It is the author's believes that the dataset introduced in section 3 is satisfactorily balanced and possesses a quality that enables the practicability of using machine learning approaches for this problem. In addition to the mentioned dataset, the author has access to resources such as a directional microphone and a laboratory. These resources facilitate the opportunity to create a new dataset consisting of leakage recordings, should this turn out to be necessary. To create a new dataset, a directional microphone will be used because of its practicality, as mentioned in section 2.2. For now, the idea is to use only one directional microphone. On the other hand, the idea of using two microphones in order to consider the benefits of looking at phase shifts is not completely disregarded. Using a directional microphone also allows for the possibility of locating leakages, as the signal-to-noise ratio will be higher when the microphone is pointed directly at the leakage source.

A combination of audio features mentioned in section 2.3.1 in addition to the MFCC's will be used as input to the machine learning algorithms as this is a standard in the field and has shown prominent results. The K-means clustering algorithm will be prioritized for the classification method. However, it is a goal of the author to be able to experiment with more of the mentioned approaches in section 2.5, and compare the performance results of several approaches on the same dataset. Therefore, if time allows it, a Convolutional Neural Network will also be developed and experimented with. Experimenting with using a combination of the audio features as input to the K-means classification algorithm will show which features lead to the best separation of the two classes, "Leakage" and "No Leakage". For the input to the Convolutional Neural Network, images in the form of Mel-spectrograms and MFCC-plots will be prioritized (see section 2.3.2), as they are often used in deep neural networks and other machine learning techniques. Furthermore, these input images will possibly be compared with the use of frequency magnitude spectrum, which was discussed in section 2.1.3.

A preliminary plan for the future master thesis work has been created and can be seen in figure 24. The exclamation marks indicate the work that might be substituted with a continuation of the work on the K-means clustering classification. Another topic of research that could be interesting to look into is using machine learning methods to not only detect if there exists a leakage, but also classify the severity or level of the leakage. One possible adaption of the K-means algorithm as discussed in section 2.5.4 is to define a higher number of clusters in order to try to solve this exact problem. However, This research topic requires a dataset with a clearly separated difference in levels of leakages and will not be prioritized in the master thesis work unless there is a high amount of extra time left over from the other experimentation work.



	Week 1	Week 2	Week 3	Week 4
January	-	-	Pre Processing of dataset	Audio Feature Extraction
February	Audio Feature Extraction	MFCCs Extraction	K-means Classification w/ Combination	K-means Classification w/ Combination
March	K-means Classification w/ MFCCs	Spectrograms Extraction (!)	CNN Development (!)	CNN Development (!)
April	CNN Classification w/ Spectrograms (!)	CNN Classification w/ Spectrograms (!)	Report Writing	Report Writing
May	Report Writing	Report Writing	Report Writing	Finishing Touches
June	Finishing Touches	Delivery	-	-

Figure 24: Preliminary plan of the master thesis working period in spring 2022.

Looking at the contents of the plan in figure 24, one can see that the first step of the practical work will include pre-processing and feature extraction of the audio signals in the dataset. Thereafter, the focus will be on developing the K-means clustering algorithm and testing it out on different combinations of the audio features mentioned in section 2.3.1. The last part of the work concerning the K-means algorithm will be to compare the use of MFCCs as input against the use of the other audio features. If there is enough time left of the working period, some experimentation with the use of a CNN will be conducted. This experimentation should include extraction of spectrograms and frequency magnitude spectrums from the audio files and using them as input to train the neural network for classification of signals into "Leakage" or "No Leakage". After that, the findings of the experimentation with one or both methods, along with other essential factors of the project, will naturally be written into a final report.

Recordings taken during a long period of time will not be used. Thus, the problem of detecting the decay of machinery over the years in order to determine if objects are about to reach their life limit will not be investigated. As the discussed leakage detection solution using daily robot scans of an industrial plant could be conducted over several years in the future, this could be an interesting thing to look further into in the years to come. Daily scans over a period of several years will produce a vast amount of data on the same system, which will be valuable for further research on long-term machine decay.

To summarize, the use of audio signal data and acoustic anomaly detection (AAD) techniques opens up many possibilities in the sound analysis field, from detecting leakages of gas and fluid, classifying degrees of leakages to locating them, and ultimately diagnosing machinery conditions. Some of these problems have been broadly investigated, and some should be explored even deeper. What should be memorized from this paper is that the applications and possibilities are vast when it comes to the use of AAD techniques and that a lot more interesting research on this topic will likely and hopefully be seen in the future.

Bibliography

[1] CUNY Brooklyn College. *"Elements of Sound"*, https://human.libretexts.org/@go/page/51172, May 29th 2020.

[2] Equinor ASA. *"Investigation of the fires at Tjeldbergodden and Hammerfest now concluded"*. https://www.equinor.com/en/news/20210512-investigation-fires-tjeldbergodden-hammerfest-concluded.html, 2020.

[3] Public Domain. *"Piano C - Grand Piano single notes Collection"*. https://samplefocus.com/samples/piano-c (retrieved 09.11.2021).

[4] European Commission. *"Equipment for potentially explosive atmospheres (ATEX)"*. https://ec.europa.eu/growth/sectors/mechanical-engineering/equipment-potentially-explosive-atmospheres-atex_en, Official Website of the European Union, 2014.

[5] K. Alsabti, S. Ranka, and V. Singh. *"An efficient k-means clustering algorithm"*, https://surface.syr.edu/eecs/43/, Electrical Engineering and Computer Science, Vol.43, 1997.

[6] J. G. Avalos, J. C. Sanchez, and J. Velazquez. *"Adaptive Filtering Applications - Chapter.1: Applications of Adaptive Filtering"*, https://www.intechopen.com/chapters/16112, 2011.

[7] R. Bachu, S. Kopparthi, B. Adapa, and B. Buket. *"Voiced/Unvoiced Decision for Speech Signals Based on Zero-Crossing Rate and Energy"*. https://www.researchgate.net/publication/259823741_VoicedUnvoiced_Decision_for_Speech_Signals_Based_on_Zero-Crossing_Rate_and_Energy, Advanced Techniques in Computing Sciences and Software Engineering, pp.279-282, 2010.

[8] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau. *"Autoencoder-based network anomaly detection"*, https://doi.org/10.1109/WTS.2018.8363930, 2018 Wireless Telecommunications Symposium (WTS), 2018.

[9] C.-F. Cheng, A. Rashidi, M. A. Davenport, and D. V. Anderson. *"Evaluation of Software and Hardware Settings for Audio-Based Analysis of Construction Operations"*, https://link.springer.com/article/10.1007/s40999-019-00409-2#citeas, International Journal of Civil Engineering, Vol.7, Nr.9, pp.1469–1480, 2019.

[10] Z. Chi, Y. Li, and C. Chen. *"Deep Convolutional Neural Network Combined with Concatenated Spectrogram for Environmental Sound Classification"*, https://ieeexplore.ieee.org/document/8962462, 2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT), 2019.

[11] W.-Y. Chuang, Y.-L. Tsai, and L.-H. Wang. *"Leak Detection in Water Distribution Pipes Based on CNN with Mel Frequency Cepstral Coefficients"*, https://doi.org/10.1145/3319921.3319926, ICIAI 2019: Proceedings of the 2019 3rd International Conference on Innovation in Artificial Intelligence, pp. 83–86, 2019.

[12] J. Daintith and E. Wright. *"A Dictionary of Computing"*, https://www.oxfordreference.com/view/10.1093/acref/9780199234004.001.0001/acref-9780199234004, Oxford University Press, 2008.

[13] S. G. David Johnson, Jakob Kirner and J. Liebetrau. *"IDMT-ISA-Compressed-Air"*. https://www.idmt.fraunhofer.de/en/publications/isa-compressed-air.html, Fraunhofer Institute for Digital Media Technology, 2020.

[14] T. B. Duman, B. Bayram, and G. Ince. *"Acoustic Anomaly Detection Using Convolutional Autoencoders in Industrial Processes"*. http://dx.doi.org/10.1007/978-3-030-20055-8_41, pp.432-442, 2020.

[15] Z. Fang, Z. Guoliang, and S. Zhanjiang. *"Comparison of different implementations of MFCC"*, https://doi.org/10.1007/BF02943243, Journal of Computer Science and Technology, Vol.16, pp.582-589, 2001.

[16] R. Gade and T. B. Moeslund. *"Thermal cameras and applications: a survey"*, https://doi.org/10.1007/s00138-013-0570-5, Machine Vision and Applications, Vol.25, Nr.1, pp.245-262, 2013.

[17] T. Giannakopoulos and A. Pikrakis. *"Introduction to Audio Analysis - A MATLAB Approach"*, https://doi.org/10.1016/C2012-0-03524-7, 2014.

[18] W. Han, C.-F. Chan, C.-S. Choy, and K.-P. Pun. *"An efficient MFCC extraction method in speech recognition"*, https://doi.org/10.1109/ISCAS.2006.1692543, 2006 IEEE International Symposium on Circuits and Systems (ISCAS), 2006.

[19] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. *"Array programming with NumPy"*, https://doi.org/10.1038/s41586-020-2649-2, Springer Science and Business Media LLC, Nature, Vol.585, pp.357-362, 2020.

[20] J. A. Hartigan and M. A. Wong. *“Algorithm AS 136: A K-Means Clustering Algorithm”*, http://www.jstor.org/stable/2346830, Journal of the Royal Statistical Society. Series C (Applied Statistics) Vol.28, No.1, pp.100–108, 1979.

[21] M. Hossin and M. N. Sulaiman. *"A review on evaluation metrics for data classification evaluations"*. https://www.researchgate.net/publication/275224157_A_Review_on_Evaluation_Metrics_for_Data_Classification_Evaluations, International Journal of Data Mining Knowledge Management Process (IJDKP) Vol.5, No.2, 2015.

[22] X. Huifa and L. Feng. *"Spectrum Estimation of Pseudo-random Nonuniformly Sampled Signals in the Fractional Fourier Transform Domain"*. https://ieeexplore.ieee.org/document/5571392, 2010 WASE International Conference on Information Engineering, 2010.

[23] Y. Jia, B. Gao, C. Jiang, and S. Chen. *"Leak diagnosis of gas transport pipelines based on Hilbert-Huang transform"*, https://ieeexplore.ieee.org/abstract/document/6273368, Proceedings of 2012 International Conference on Measurement, Information and Control, 2012.

[24] D. Kampelopoulos, N. Karagiorgos, G. P. Kousiopoulos, D. Porlidas, V. Konstantakos, and S. Nikolaidis. *"An RMS-based Approach for Leak Monitoring in Noisy Industrial Pipelines"*. https://ieeexplore.ieee.org/document/9460014, 2021 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), 2021.

[25] D. Kampelopoulos, G.-P. Kousiopoulos, N. Karagiorgos, V. Konstantakos, S. Goudos, and S. Nikolaidis. *"Applying One Class Classification for Leak Detection in Noisy Industrial Pipelines"*, https://ieeexplore.ieee.org/document/9493355, 2021 10th International Conference on Modern Circuits and Systems Technologies (MOCAST), 2021.

[26] W. Z. Khan, M. Y. Aalsalem, W. Gharibiand, and Q. Arshad. *"Oil and Gas monitoring using Wireless Sensor Networks: Requirements, issues and challenges"*. https://ieeexplore.ieee.org/abstract/document/7849577, 2016 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET), pp.31-35, 2016.

[27] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, and C. Willing. *"Jupyter Notebooks – a publishing format for reproducible computational workflows"*, https://jupyter.org/, Positioning and Power in Academic Publishing: Players, Agents and Agendas, pp.87-90, 2016.

[28] M. Kotani, M. Katsura, and S. Ozawa. *"Detection of gas leakage sound using modular neural networks for unknown environments"*. http://dx.doi.org/10.1016/j.neucom.2004.06.002, Neurocomputing, Vol.62, pp.427-440, 2019.

[29] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas. *"Handling imbalanced datasets: A review"*. https://www.researchgate.net/publication/228084509_Handling_imbalanced_datasets_A_review, GESTS International Transactions on Computer Science and Engineering, Vol.30, 2006.

[30] H. Lee, N.-W. Kim, un Gi Lee, and B.-T. Lee. *"A Study on Distance Measure for Effective Anomaly Detection using AutoEncoder"*, https://doi.org/10.1109/ICTC49870.2020.9289177, 2020 International Conference on Information and Communication Technology Convergence (ICTC), 2020.

[31] A. Likas, N. Vlassis, and J. J. Verbeek. *"The global k-means clustering algorithm"*, https://doi.org/10.1016/S0031-3203(02)00060-2, Pattern Recognition, Vol.36, Nr.2, pp.451-461, 2003.

[32] C. Lin and D. Wang. *"Spectrogram Image Encoding based on Dynamic Hilbert Curve Routing"*. https://ieeexplore.ieee.org/document/5586805, 2010 2nd International Conference on Image Processing Theory, Tools and Applications, pp.107-111, 2010.

[33] O. Manjrekar and M. Duduković. *"Identification of flow regime in a bubble column reactor with a combination of optical probe data and machine learning technique"*, http://dx.doi.org/10.1016/j.cesx.2019.100023, Chemical Engineering Science X Journal, Vol.2, 2019.

[34] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto. *LibROSA: Audio and music signal analysis in Python*, https://librosa.org/doc/latest/index.html, Proceedings of the 14th Python in Science Conference, Vol.8, 2015.

[35] H. Meng, T. Yan, F. Yuan, and H. Wei. *"Speech Emotion Recognition From 3D Log-Mel Spectrograms With Deep Learning Network"*, https://doi.org/10.1109/ACCESS.2019.2938007, IEEE Access, Vol.7, pp.125868-125881, 2019.

[36] Y. V. Mikhlin, W. Wang, Q. Chen, X. Liang, X. Yue, and J. Dou. *"A Novel Multidimensional Frequency Band Energy Ratio Analysis Method for the Pressure Fluctuation of Francis Turbine"*. Mathematical Problems in Engineering, Hindawi, https://doi.org/10.1155/2018/3494785, 2018.

[37] Y. Nagaya and M. Murase. *"Detection of Cavitation with Directional Microphones placed Outside Piping"*, https://doi.org/10.1016/j.nucengdes.2011.08.045, Nuclear Engineering and Design, Vol.249, pp.140-145, 2012.

[38] S. Omar, M. A. Ngadi, H. H. Jebur, and S. Benqdara. *"Machine Learning Techniques for Anomaly Detection: An Overview"*, http://dx.doi.org/10.5120/13715-1478, International Journal of Computer Applications, Vol.79, Nr.2, 2013.

[39] K. T. O'Shea and R. Nash. *"An Introduction to Convolutional Neural Networks"*. https://www.researchgate.net/publication/285164623_An_Introduction_to_Convolutional_Neural_Networks, 2015.

[40] P. F. Pai. *"Space Wavenumber and Time–Frequency Analyses for Vibration and Wave-Based Damage Diagnosis"*, https://www.sciencedirect.com/science/article/pii/B9780081001486000147, Structural Health Monitoring (SHM) in Aerospace Structures, pp.393–426, 2016.

[41] P. Patel, M. Nandu, and P. Raut. *"Initialization of Weights in Neural Networks"*. https://www.researchgate.net/publication/330875010_Initialization_of_Weights_in_Neural_Networks, 2019.

[42] C.-Y. Peng, U. Raihany, S.-W. Kuo, and Y.-Z. Chen. *"Sound Detection Monitoring Tool in CNC Milling Sounds by K-Means Clustering Algorithm"*, https://www.researchgate.net/publication/352702642_Sound_Detection_Monitoring_Tool_in_CNC_Milling_Sounds_by_K-Means_Clustering_Algorithm, Sensors, Vol.21, 2021.

[43] V. H. Phung and E. J. Rhee. *"A High-Accuracy Model Average Ensemble of Convolutional Neural Networks for Classification of Cloud Image Patches on Small Datasets"*, http://dx.doi.org/10.3390/app9214500, MDPI Journal, 2019.

[44] G. V. Rossum and F. L. Drake. *Python 3 Reference Manual*, version 3.8, http://www.python.org, CreateSpace, 2009.

[45] J. Sandsten, P. Weibring, H. Edner, and S. Svanberg. *"Real-time Gas-correlation Imaging Employing Thermal Background Radiation"*, https://www.osapublishing.org/oe/fulltext.cfm?uri=oe-6-4-92&id=63460, Optics Express Journal, Vol.6, Nr.4, pp.92, 2000.

[46] A. Schenck, W. Daems, and J. Steckel. *"AirLeakSlam: Automated Air Leak Detection"*, http://dx.doi.org/10.1007/978-3-030-33509-0_70, pp.746-755, 2020.

[47] A. Senior, G. Heigold, M. Ranzato, and K. Yang. *"An empirical study of learning rates in deep neural networks for speech recognition"*. https://static.googleusercontent.com/media/research.google.com/no//pubs/archive/40808.pdf, 2013.

[48] S. A. Shahriyar, M. A. H. Akhand, N. Siddique, and T. Shimamura. *"Speech Enhancement Using Convolutional Denoising Autoencoder"*, https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8679106, 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), 2019.

[49] V. Sharma, Taksh, K. Srivastav, Priyam, and N. A. Siddiqui. *"A Critical Study on Role of Sensor-Based Electronic System for Toxic Gas Identification in the Mining (Coal) Industry"*, https://doi.org/10.1007/978-981-10-5903-2_157, Advances in Intelligent Systems and Computing Journal, pp. 1511-1521, 2018.

[50] T. Shindoi, T. Hirai, K. Takashima, and T. Usami. *"Plant equipment diagnosis by sound processing"*, https://ieeexplore.ieee.org/document/816552, IECON'99 - Conference Proceedings, 25th Annual Conference of the IEEE Industrial Electronics Society (Cat. No.99CH37029), Vol.2, pp.1020-1026, 1999.

[51] Z. Shuqing, G. Tianye, H. Xu, J. Jian, and W. Zhongdong. *"Research on Pipeline Leak Detection Based on Hilbert-Huang Transform"*, https://ieeexplore.ieee.org/abstract/document/5363780, 2009 International Conference on Energy and Environment Technology, 2009.

[52] K. Suefusa, T. Nishida, H. Purohit, R. Tanabe, T. Endo, and Y. Kawaguchi. *"Anomalous Sound Detection Based on Interpolation Deep Neural Network"*, https://doi.org/10.1109/ICASSP40776.2020.9054344, ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp.271-275, 2020.

[53] T. Sundstrom and S. Schlicker. *"Applications and Modeling with Sinusoidal Functions"*. https://math.libretexts.org/@go/page/7106, Grand Valley State University, Mathematics LibreTexts, 2021.

[54] R. N. Tak, D. M. Agrawal, and H. A. Patil. *"Novel Phase Encoded Mel Filterbank Energies for Environmental Sound Classification"*, http://dx.doi.org/10.1007/978-3-319-69900-4_40, International Conference on Pattern Recognition and Machine Intelligence, pp.317-325, 2017.

[55] G. Vallet, D. Shore, and M. Schutz. *"Exploring the Role of the Amplitude Envelope in Duration Estimation"*, http://dx.doi.org/10.1068/p7656, Perception Journal, Vol.43, pp.613-630, 2014.

[56] R. J. van der Vleuten and F. Bruekers. *"Lossless compression of binary audio signals"*, https://ieeexplore.ieee.org/document/672321, Proceedings DCC '98 Data Compression Conference, 1998.

[57] F.-W. Wellmer. *"Standard Deviation and Variance of the Mean"*, https://link.springer.com/chapter/10.1007/978-3-642-60262-7_6, Statistical Evaluations in Exploration for Mineral Deposits, pp.41-43, 1998.

[58] R. Xiao, Q. Hu, and J. Li. *"Leak Detection of Gas Pipelines using Acoustic Signals based on Wavelet Transform and Support Vector Machine"*, https://doi.org/10.1016/j.measurement.2019.06.050, Measurement Journal, Vol. 146, pp.479-489, 2019.

[59] L. Xiong, C. Wang, X. Huang, and H. Zeng. *"An Entropy Regularization k-Means Algorithm with a New Measure of between-Cluster Distance in Subspace Clustering"*, http://dx.doi.org/10.3390/e21070683, MDPI Journal, 2019.

[60] T. Xu, Z. Zeng, X. Huang, J. Li, and H. Feng. *"Pipeline leak detection based on variational mode decomposition and support vector machine using an interior spherical detector"*, https://doi.org/10.1016/j.psep.2021.07.024, Process Safety and Environmental Protection Journal, Vol.153, pp.167-177, 2021.

[61] X. Zhang. *"Benefits and Limitations of Common Directional Microphones in Real-World Sounds"*, http://dx.doi.org/10.11648/j.cmr.20180705.12, Clinical Medicine Research, Vol.7, 2018.

Appendix

A Source code

A.1 Calculation of the mean rms-energy of the signal

```
# Method for calculating mean values for each 3 second interval of the
↪  input rms_values
# The output is a list of the mean for each sample in the signal
def get_means(rms_values):
    means = []
    # Calculating the mean rms-energy for each 3 seconds of the signal
    tree_sec_chunk = int(np.floor(len(rms_values.T)/10))
    # All files in the dataset is 30 seconds long (30/10 = 3 second
    ↪  intervals)
    mean1 = np.mean(rms_values.T[0:tree_sec_chunk])
    mean2 = np.mean(rms_values.T[tree_sec_chunk+1:tree_sec_chunk*2])
    mean3 = np.mean(rms_values.T[tree_sec_chunk*2:tree_sec_chunk*3])
    mean4 = np.mean(rms_values.T[tree_sec_chunk*3:tree_sec_chunk*4])
    mean5 = np.mean(rms_values.T[tree_sec_chunk*4:tree_sec_chunk*5])
    mean6 = np.mean(rms_values.T[tree_sec_chunk*5:tree_sec_chunk*6])
    mean7 = np.mean(rms_values.T[tree_sec_chunk*6:tree_sec_chunk*7])
    mean8 = np.mean(rms_values.T[tree_sec_chunk*7:tree_sec_chunk*8])
    mean9 = np.mean(rms_values.T[tree_sec_chunk*8:tree_sec_chunk*9])
    mean10 = np.mean(rms_values.T[tree_sec_chunk*9:len(rms_values.T)])
    # Adding the means data to a list (to be able to plot it later)
    for i in range(tree_sec_chunk):
        means.append(mean1)
    for i in range(tree_sec_chunk):
        means.append(mean2)
    for i in range(tree_sec_chunk):
        means.append(mean3)
    for i in range(tree_sec_chunk):
        means.append(mean4)
    for i in range(tree_sec_chunk):
        means.append(mean5)
    for i in range(tree_sec_chunk):
        means.append(mean6)
    for i in range(tree_sec_chunk):
        means.append(mean7)
    for i in range(tree_sec_chunk):
        means.append(mean8)
    for i in range(tree_sec_chunk):
        means.append(mean9)
    for i in range(len(rms_values.T)-tree_sec_chunk*9):
        means.append(mean10)
    return means
```