Friedrich-Alexander-Universität Erlangen-Nürnberg

Friedrich-Alexander-Universität
Erlangen-Nürnberg

Master Thesis

Anomalous Event Detection in Wireless Acoustic Sensor Networks

submitted by

Chenxi Guo

submitted

November 3, 2022

Supervisor / Advisor

Prof. Dr. Nils Peters
MSc. Lorenz Schmidt

Reviewers

Prof. Dr. Nils Peters

International Audio Laboratories Erlangen
A joint institution of the
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and
the Fraunhofer-Institut für Integrierte Schaltungen IIS.

Erklärung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Erlangen, December 5, 2022

Chenxi Guo

Acknowledgements

I would like to express my gratitude to the people who supported me in the time of writing my thesis.
I would like to express my sincere thanks to Prof. Dr. Nils Peters, who gives me a clear direction for my research and patient help throughout this thesis. Thank you not only for allowing me to do this research but also for supporting my work. Thank you for giving me the opportunity to work in such a friendly and stress-free environment, the time at audio labs will be my unforgettable experience in Germany.
I would also like to extend my gratitude to my supervisor Lorenz Schmidt, who is such a nice and talented guy. Thank you for your careful guidance in my research, your patience and encouragement when I encountered problems, as well as your tolerance of my sometimes stupidity. It is a great pleasure for me to work with you. I wish you all the best in your Ph.D. program and your academic career.
Special thanks to my family, who always supports me and encourages me when I am upset and depressed. Thank you for the love you give me. Your love is my greatest spiritual strength in this country.
Thanks should also go to all of my friends. Zhang, Huang, Zou, Du...thank you all for the joy you bring me and your help.

Abstract

Human-computer interaction (HCI) appeared during the creation of personal computers and becomes increasingly ubiquitous in recent years. With the advent of small embedded devices with low computational complexity and non-intrusive sensors, many applications are possible nowadays. For example, monitoring private, business-related (e.g. industrial machines) or public goods (e.g. roads, forests) helps humans to assess their conditions. It makes human interventions only necessary when anomalies occur, examples can be burglaries, car crashes, or illegal logging. We develop an anomaly detection algorithm that is applicable to home environments. Key requirements are a wireless ad-hoc network structure, multi-room monitoring, and low computational demand. The resident should not have to worry about the computer-computer interaction and as many devices as possible should be able to participate. Therefore it consists of a low-complexity Wireless Acoustic Sensor Network (WASN) and an edge device, responsible for inferring an anomaly score. The WASN operates an algorithm representing the current acoustic scene. We use the Alternative Direction Method of Multipliers (ADMM) to derive an algorithm for Non-negative Factorization (NMF), estimating optimal codebooks for sensed audio in our WASN. The anomaly score is estimated based only on the code vectors. We train the model on the SINS dataset, which includes continuous real-life audio recordings over one week in a home environment. We integrate our approach for a distributed NMF into an anomaly detection system and evaluate the performance. The experiment results show that our detection system has a good performance under different parameters.

Contents

Chapter 1

Introduction

1.1 Motivation of the Research

Security monitoring is becoming more and more common and important in modern society caused by the advancement of information and networking technology. Many families, companies, and factories are using area monitoring systems now. The current popular security monitoring system mainly collects video information from the monitored area through cameras. By analyzing these information, they can detect anomalies in the monitored area and react in time. But video-based monitoring systems also have drawbacks. First, a camera can only monitor one direction, to achieve better real-time monitoring, it must be able to move to the area where the anomalous events occur, while omnidirectional cameras are very expensive. Next, video surveillance is easily affected by light, when the light conditions are not met, the video quality will be greatly reduced. Besides, in some private or confidential places, there are some privacy issues with video surveillance. Because individuals with known feature vectors are video-recorded in their daily lives. These videos can form information-rich datasets which can reveal sensitive personal information, including family life and daily habits, religious beliefs, etc. The stored data can be accessed by malicious users such as rogue insiders or hackers. If the leaked private information is used illegally, it can cause significant damage and harm to the victim. Finally, images are two-dimensional signals, which are complicated to process and computationally intensive. Compared with image processing, there are many advantages of using sound signals for in-area anomaly detection. They have a lower processing complexity due to sparsity of spectrogram compared to ordinary images. Microphone nodes for obtaining acoustic signals are cost-effective and can be widely distributed in the context of the application, allowing easy changes to the network structure. The sound stream also contains the sufficient information for making inference about scenarios.
Anomaly sounds can effectively reveal anomalous situations as well as unexpected events, and once it is detected, a timely response can be made to avoid greater losses. By detecting and identifying

sound information in the environment, the security system can also focus on certain areas, which can make up for the lack of video surveillance. The current research of anomaly sound detection mainly focuses on three parts, signal feature extraction, feature representation and anomaly detection. First extracting the acoustic features of normal sounds with suitable methods. Then choosing algorithms to represent these features. Finally feeding it into classification models. So that a classification model can be trained. In the detection phase, the representation of the acoustic features of the samples to be tested is fed into the trained model for the prediction. Extracting appropriate features and properly representing them can help us get good results in anomaly sound detection. With the efficient features we can train a classification model that works well, which is the ultimate goal. Depending on the choices of features and models, diverse detection schemes are established.

The features used in anomaly detection tasks are similar to those used in other common audio processing tasks, such as time domain features like short-time energy and short-time transient rate, frequency domain features like spectral center of mass and bandwidth, or cepstrum features like the Mel filterbank features or Mel-frequency cepstral coefficients[16]. However, these features are usually high dimensional and generally requires a large amount of computation for processing. To be more efficient, a practical approach is to reduce the dimensionality of these features before they are fed into the classification model. Non-negative matrix decomposition (NMF) is a popular dimensionality reduction algorithm that has the universality of modeling various types of audio sources, including speech, music, environmental sounds, etc. [22]. As a result, it is often used in cases where the extracted audio features need to be represented such as speech enhancement, speech recognition and anomaly sound detection.

With the advantages of low cost, battery-operated power supply, flexible topology, ability to sense and detect the external environment, parallel data processing, information sharing between nodes and nodes collaboration, wireless acoustic sensor networks (WASNs) are usually used for real-time monitoring and collecting acoustic information in the area of observation. Common applications are hearing aids, voice communication systems, acoustic detection, and ambient intelligence, etc.

Nevertheless, WASNs have drawbacks, either. For instance, the available energy of each node is limited, meaning that each node can only process small amount of data. In [8] A. Bertrand et al. proposed to set up a node as fusion center so that normal nodes only transmits data, the storage and processing of data is performed in the fusion center. But it requires high storage capacity and computing ability of the fusion center nodes. Besides, it is generally applicable only for stationary networks. Further for ad-hoc networks, where nodes can join and leave anytime, finding a fusion center requires additional coordination. Moreover, because of the limitation of battery energy and the transmission distance between nodes, such fusion centers are less applicable in applications covering wide areas.

Distributed algorithms do not require a fusion center and can avoid many problems occurring in a WASN context. It takes into account the node characteristics of sensor networks and is able to utilize limited energy resources and node collaboration to achieve optimal representation of signals. Flexibility is often a key challenge, in the topology the nodes should be able to join and leave anytime and only communicate with a subset of neighbors. Algorithms working in such scenarios should synchronize locally solved problems in a global context. This can be done by splitting variables into local and global variants.
As a widely used constrained problem optimization method in machine learning, the Alternating Direction Method of Multipliers (ADMM) was proved to be applicable to distributed computing systems and large-scale distributed optimization problems by Boyd et al. in [9].
For the above reasons, combining the features and functions of WASNs, we propose two distributed NMF algorithms based on ADMM framework. In these algorithms the local optimization is alternating with the global synchronization. It is a parametrized representation of the acoustical scene. We design and implement a machine learning model-based anomaly sound detection system as well, and evaluate its performance accordingly with different parameters.

1.2 Application Scenarios

The anomaly sound detection system we designed has a variety of application scenarios, such as home life, industrial monitoring, road monitoring, etc. In this thesis, we are mainly concerned with anomaly sound for indoor acoustic scenarios. The designed system can be used to detect emergencies in the home environment. For example, when a child cries such a system would alert its caretakers to react in time. Similarly, when an elderly person living alone suddenly falls and calls for help or moans for a long time, once the system detect that, it issues an alarm to prevent further harm. Integration with video surveillance systems could be considered as well, so that when an anomalous sound is detected, the video surveillance system is notified to capture and analyze the scene.

Meanwhile, if a different dataset is used as the input signal for the training process, the system can also be applied to industrial monitoring such as machinery anomaly sound detection. In this way, it is helpful to know whether the machine is malfunctioning or not, or to determine and pre-treat the possible malfunctioning status in advance to reduce maintenance costs and expenses. In addition, the system can be used for the detection and alarm of emergency events in public places as well. It can detect public security events, such as bank robberies or terrorist attacks in shopping malls.
Besides, due to different applications and practical situations, WASNs may have various topologies. Some nodes may join or leave the network at any time for environmental monitoring purposes.

Our proposed distributed algorithm can find a global solution regardless of the topology. It is independent of the number of nodes and can obtain good signal representation in different network structures. For this reason, it is suitable for acoustic sensor networks with different architectures, which satisfies the requirement of flexibility concerning topology.

1.3 Thesis Organization

This thesis is organized as follows
In Chapter 1, we briefly introduce the basic idea of WASNs and anomaly sound detection, and the necessity of using distributed algorithms in WASNs. Then we introduce the application scenarios of the proposed NMF algorithm using ADMM framework and the anomaly sound detection system.
In Chapter 2, we detailed introduce the theoretic foundations related to this research. We talk about the features of WASNs, the basics of NMF algorithm and ADMM framework, and the widely used anomaly sound detection techniques. The audio pre-processing and feature extraction methods for anomaly sound detection are introduced, as well as the commonly used representation methods of acoustic features and some machine learning-based classification models.
In Chapter 3, we introduce the proposed methodology including the NMF based on ADMM framework, NMF using consensus ADMM for fully connected topology and NMF using consensus ADMM for all topologies. These algorithms can satisfy different demands of network structure. Then we introduce the proposed unsupervised anomaly sound detection system, including the framework for the whole system and the information exchange at one node.
In Chapter 4, we first introduce the overall experimental scheme and setup. It includes the datasets used in our experiments and the detailed settings of experimental parameters for the evaluation. Then we evaluate the performance of the proposed NMF algorithm under different settings. Finally, we analyze the overall evaluation and the problem met in the experiments.
In Chapter 5, we draw the conclusions of this thesis and give an outlook for future work. Results of different experimental settings are shown in Appendix A.

Chapter 2

Theoretic Foundations

This chapter describes the relevant concepts and techniques used in this research. Firstly, the definition and properties of wireless acoustic sensor networks are described. Then, the concepts and applications of the non-negative matrix decomposition algorithm and the framework of the ADMM algorithm are explained. Finally, the relevant techniques in the study of anomaly sound detection are introduced, including commonly used feature analysis methods for anomaly sound detection, the representation and classification methods of sounds. This chapter provides a comprehensive description of several key technical points of this research and provides the basis for the subsequent theoretical analysis and experiments.

2.1 Wireless Acoustic Sensor Networks

Wireless sensor networks (WSNs) have developed rapidly in recent years and can be applied in areas such as defense and military, human-computer interaction, and environmental monitoring. It consist of miniature, self-contained, self-powered sensor nodes, each node contains one or more sensors. These sensing nodes collect information from the observed environment and process it with distributed technology. The connectivity of WSNs relies on the principles behind wireless self-organizing networks, where nodes can communicate with each other to achieve information sharing and collaboration between nodes. When the observation object is sound, there are one or more microphone sensors at each node, and we call this kind of sensor network wireless acoustic sensor networks (WASNS). It can be used in smart cities, home automation, cars or mobile phone ad-hoc networks. Typical applications are hearing aids, hands-free telephony, acoustical monitoring and ambient intelligence.

Previously used acoustic capture and processing systems have tended to rely on microphone arrays. However, due to the fixed nature of the microphone array locations and the fact that the data processing must be finished on a central processor, these systems always have their

limitations such as the lack of flexibility and high cost. Since the central processor needs to store and process the data coming from the entire microphone array, it requires a very strong storage and computation capability, which means the cost is expensive. Besides, since the microphone array only samples the sound field locally, and the distance between it and the object sound source is usually quite far, the recorded signal often has a low signal-to-noise ratio (SNR) [8].
Compared with traditional microphone arrays, WASNs allow the use of more low-cost miniature microphone nodes to cover a larger area. It increases the probability that some microphone nodes are located close to the sound source, so that it can produces recordings with higher quality. Due to the use of wireless communication, WASNs are not limited by the size of the array and can be placed in locations that are not suitable for wired microphones. The nodes in it can join or leave the network at any time and form different topologies based on different communication protocols, which shows flexibility. Figure 2.1 shows the microphone nodes containing multiple microphones. It can be randomly distributed in the environment where the acoustic signal should be collected. The main task of microphones in the sensor nodes is to detect acoustic signals from different transmission points or different reception points, then provide the required information for acoustic environment analysis. The processing of information for different tasks will then be performed by the processor equipped at each node.
A. Bertrand proposed different topologies for WASNs in [8], including the centralized topology with a fusion center and the mesh topology. Figure 2.2 shows the scheme of the centralized topology. It requires each node to transmit the recorded microphone signals to a dedicated device called fusion center, then all data is stored and processed in the fusion center. But due to the limited communication capacity and battery energy of the nodes, this topology is not applicable in most cases. Figure 2.3 shows the ad-hoc-based WASNs with the mesh topology. Each node can locally process its own data and share information with their connected nodes, which alleviates the requirements for communication bandwidth and computational complexity. Since each node only has access to a subset of the available data, it is more challenging to design algorithms for such distributed settings.

The WASNs has a lot of advantages such as the small size and low cost of individual microphone and the flexibility in network structure and nodes location. They are generally used in acoustic processing, signal enhancement like speech enhancement, parameter estimation like representing the acoustic features, estimating the location of the speaker, etc.
However, WASNs are not perfect. It uses embedded processors and memory with very limited computation power and data storage capacity. Therefore, it requires that the designed distributed algorithms must have high efficiency and low complexity. WASNs nodes are mainly powered by batteries, which must be replaced when they run out. When the energy of the battery carried by the node is exhausted, the node becomes useless. If multiple nodes fail at the same time, it will cause a significant impact on the measurement or even lead to a complete failure. So the

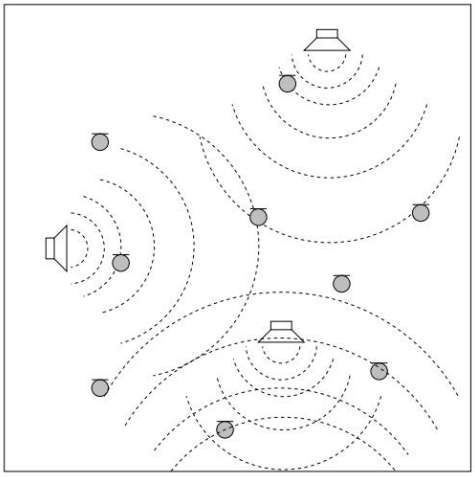


Figure 2.1: Schematic example of randomly distributed microphone nodes in WASNs[8].

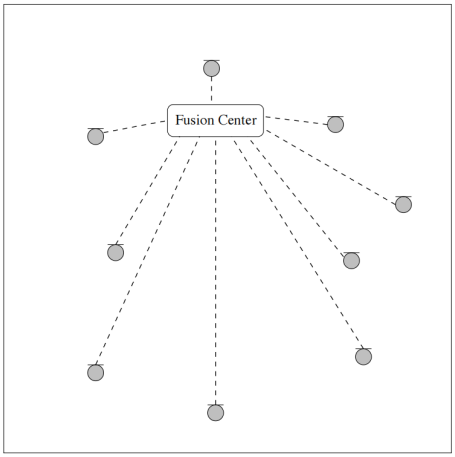


Figure 2.2: Schematic example of centralized processing by means of a fusion center [8].

designed algorithm must also consider the computational amount of each node.

2.1.1 Acoustic Feature Extraction

Due to the time-varying nature of sound, it is difficult to analyze the sound signal in the time domain. So it is required to first extract the high-dimensional features of the sound signal, which can characterize the sound and are easier to calculate. These acoustic features represent the essential properties of the audio signal. The feature extraction is a crucial step in acoustic processing tasks. Extracting the acoustic features with high discrimination and noise resistance can greatly improve the performance of audio processing tasks.
After a long time of development, the research on sound features has been well established. The

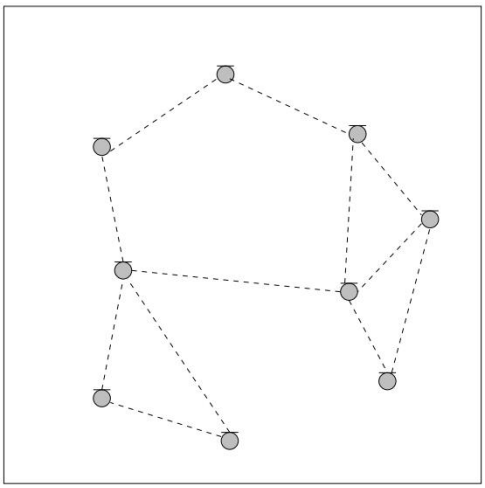


Figure 2.3: Schematic example of distributed processing in a WASN with a mesh topology [8].

common acoustic features contain time domain features, perceptual features, etc. Because of the time-varying and non-stationary characteristics, it is necessary to first sub-frame the sound and then add windows to the data frames during processing, so that the short-time sound can be considered as a time-invariant stationary signal. Commonly used time domain features include Short Time Zero Rate (ST-ZCR) [17], Short Time Energy (STE) [6], Linear Predictive Coefficient (LPC) [12], etc. In our research we choose two kinds of time-frequency domain features, the spectrogram obtained by short-time Fourier transform (STFT), and Mel filter banks features.
STFT is a simple, intuitive representation of a signal in the time and frequency domains. The basic idea is to add a window with a window function $g(\tau)$ to the original signal, the window translates with time. Then calculate the spectrogram of the original signal by segment. The short-time Fourier transform is a two-dimensional function of energy to time and frequency. For a given signal x(t), its STFT can be calculated by

$$X(t,f) = \int_{-\infty}^{+\infty} x(\tau) g \cdot (\tau - t) e^{-j2\pi f \tau} d\tau \tag{2.1}$$

For the discrete time signal X(m), its STFT is calculated by

$$X(m,\omega) = \sum_{-\infty}^{+\infty} x(n) g[n-m]) e^{-j\omega n} \tag{2.2}$$

Figure 2.4 shows the spectrogram of an audio sample obtained by STFT. The output is complex-valued.
The other chosen acoustic feature is the Mel filterbank features. It is based on the perceptual characteristics of human hearing and only focuses on the part of the information that human listeners would consider important. It introduces the Mel-scale, which describes the perceptual

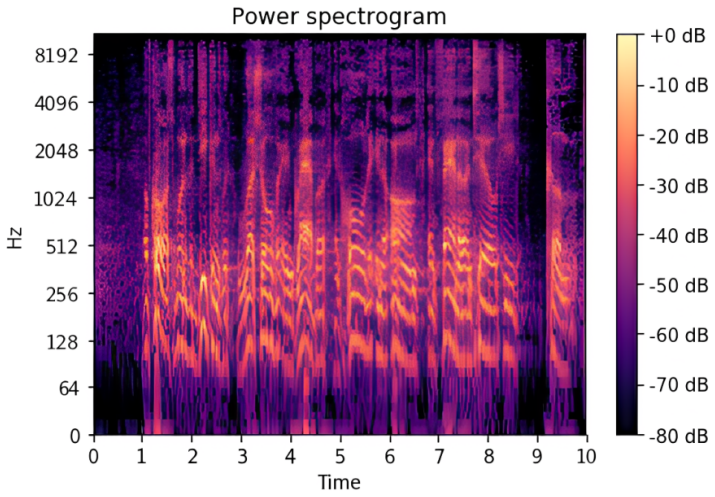


Figure 2.4: The spectrogram of an audio sample obtained by STFT.

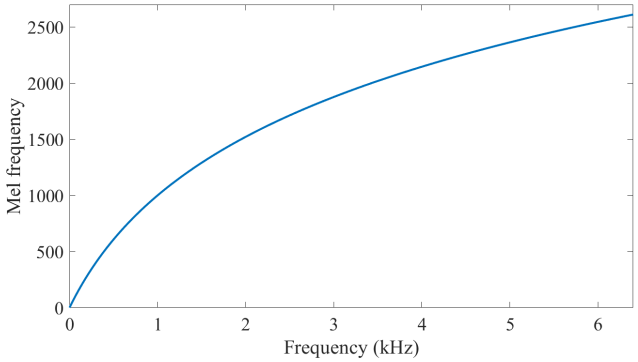


Figure 2.5: Mel Frequency versus actual frequency [3].

distance between different frequencies. A classical approximation is to define the frequency to Mel transform function for a frequency f as

$$m = 2595 \log_{10}(1 + 700f) \tag{2.3}$$

Figure 2.5 shows the relationship between the actual frequency of the sound perceived by the human ear auditory system and the Mel frequency. It can be seen that when the frequency of sound is low, the Mel frequency is approximately linear with the frequency, as the actual frequency increases, the Mel frequency is approximately logarithmic with it.

2.2 Non-Negative Matrix Factorization

Data in tensor form is often used for analysis in scientific research, such as image, text and audio. Tensors can represent high-dimensional data in a more intuitive and concrete way. For example, spectrogram with complex values is commonly used as input in acoustic tasks, each vector in it can be stored as one element in the tensor. To efficiently process these data, a common step is to decompose the tensor. This achieves two things, first the dimensionality of the matrix describing the problem is reduced, and second, features with low energy are discarded and generalization improved. Some major decomposition methods, such as principal component analysis (PCA) [5], independent component analysis (ICA) [27], and singular value decomposition (SVD) [7], all have negative elements in the decomposition results, which lacks an intuitive physical meaning in practice. For this reason, Lee and Seung proposed the non-negative matrix decomposition method (NMF) [18], which requires all elements to be non-negative in the decomposition, so that the decomposition results are more practically meaningful. For example, in image decomposition, a tensor represents an image, and each element in the tensor is a pixel in the image, the value of the pixel is non-negative. In audio representation, each tensor represents a spectrogram of audio, then all the elements in the tensor should also be non-negative. By ensuring the non-negativity of the decomposition, the NMF algorithm will retain more information reflected by the original samples after decomposition.

2.2.1 Problem Formulation

NMF is a very common factorization method, which finds applications to many fields like image feature recognition, speech recognition, etc. It aims at finding pattern-based, linear representations of non-negative data by factorizing it as the product of two low-rank non-negative matrices [18]. The objective of NMF is dimensionality reduction and feature extraction. The main idea of NMF is:
Given a non-negative matrix V $\in {\mathbb{R}_+}^{m\times n}$, find non-negative matrix factors W $\in {\mathbb{R}_+}^{m\times k}$ and H $\in {\mathbb{R}_+}^{k\times n}$ ($k > 0$ and $k \leq \min\{m, n\}$) such that

$$V \approx WH \tag{2.4}$$

As shown in Figure 2.6, V can be regarded as a set of multivariate n-dimensional data vectors, m is the number of examples in the data set. The NMF algorithm can be viewed as a linear superposition of non-negative data due to the non-negative restrictions of the matrices W and H. Thus, a column vector in the original matrix V can be interpreted as a weighted sum of all column vectors (called basis vectors) in the basis matrix or codebook W, and the weight coefficients is the element of the corresponding column vector in the coefficient matrix H. One

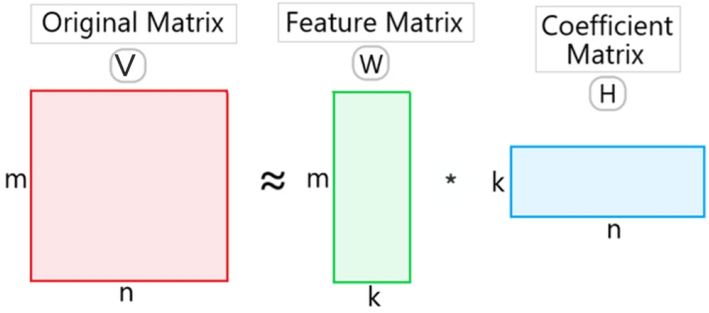


Figure 2.6: The intuitive understanding of NMF [2]. V is the original matrix with m rows and n columns, W is the feature matrix with m rows and k columns, H is the coefficient matrix with k rows and n columns.

useful method to find the optimal approximate factorization $V \approx WH$ is to use the Frobenius norm between the original matrix V and the product of W and H as the loss function. The optimum value of the loss function should be 0, which means we get the perfect factorization. In this way, the problem can be stated as

$$\begin{aligned} \text{minimize} \quad & \| V - WH \|_F^2 \\ \text{subject to} \quad & W \in {\mathbb{R}_+}^{m \times k}, H \in {\mathbb{R}_+}^{k \times n} \end{aligned} \tag{2.5}$$

Besides using the Frobenius-norm as the loss function, there are also other popular loss function such as the β-divergence, Itakura-Saito divergence, Kullback-Leibler divergence [18]. Since the matrices obtained from the decomposition are restricted to be non-negative, their product is hardly equal to the original matrix. Therefore, NMF is an NP-hard problem. The details are shown in [33].

2.2.2 Multiplicative Update Rules

A classic and popular approach called multiplicative update rules to solve the above problem is proposed by Lee and Seung in [18], it trades off the speed of convergence and ease of implementation. Its basic idea is to choose a stationary factor, W or H, then take derivatives with respect to the stationary factor to minimize the loss function. The learning rate is also needed to e chosen and all values must keep positive. The details of the proof are provided in [18], which clearly shows the iterative process and the maintenance of the non-negativity of W,

H. The update rules with respect to W and H are

$$\begin{aligned} W_{ia}^{k+1} &= W_{ia}^{k} \frac{(V(H^k)^T)_{ia}}{(W^k H^k (H^k)^T)_{ia}}, \forall i, a \\ H_{bj}^{k+1} &= H_{bj}^{k} \frac{((W^{k+1})^T V)_{bj}}{((W^{k+1})^T W^{k+1} H^k)_{bj}}, \forall b, j \end{aligned} \tag{2.6}$$

Figure 2.7 shows an example of NMF using multiplicative rules. The spectrogram in Figure 2.7a obtained by STFT is decomposed into the codebook W and coefficient matrix H. Notice that W has 10 columns, which denotes the number of features.
Because of its simplicity of implementation and the clear understandability, the multiplicative update rule is one of the most commonly used NMFs. However, numerical experiments show that in practice it often converges slowly and suffers from stability [14].

Figure 2.8 shows the convergence of decomposing the spectrogram shown in Figure 2.4 with the multiplicative NMF. The horizontal coordinate is the number of iterations, the vertical coordinate is the value of loss function. It's clear that it needs about 300 iterations to reach the approximate convergence. The slow convergence does not satisfy the requirement of fast computational speed in practical applications.

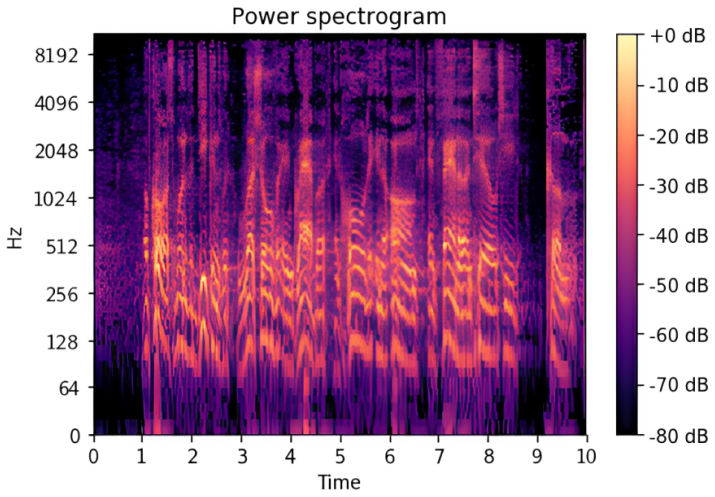
2.3 Alternating Direction Method of Multipliers

Alternating Direction Method of Multipliers (ADMM) is a method to solve decomposable convex optimization problems. It can equivalently decompose the objective function of the original problem into several solvable subproblems, then solve each subproblem in parallel, and finally coordinate the solutions of the subproblems to obtain the global solution of the original problem. ADMM is said to have the decomposability of dual decomposition and the excellent convergence of the augmented Lagrangian methods for constrained optimization [28]. ADMM is not only suitable for solving optimization problems with constrained terms, but also can be extended to large scale problems, so it is widely used for solving large scale regression, classification and other machine learning problems.

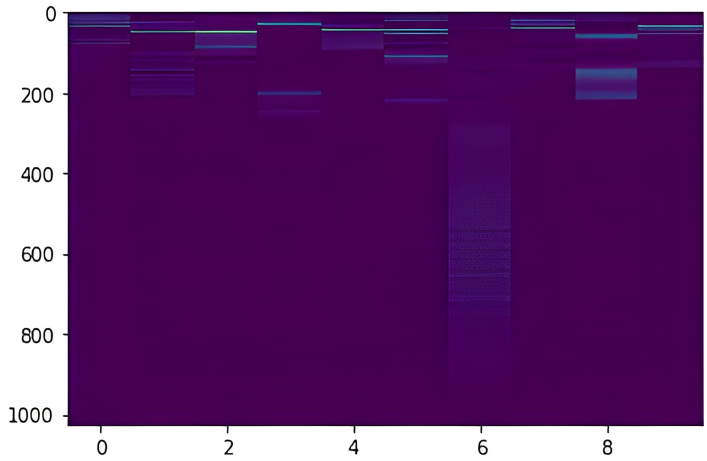
2.3.1 Definition of ADMM

The classical ADMM algorithm is suitable for solving the following 2-block (or N-block) convex optimization problems

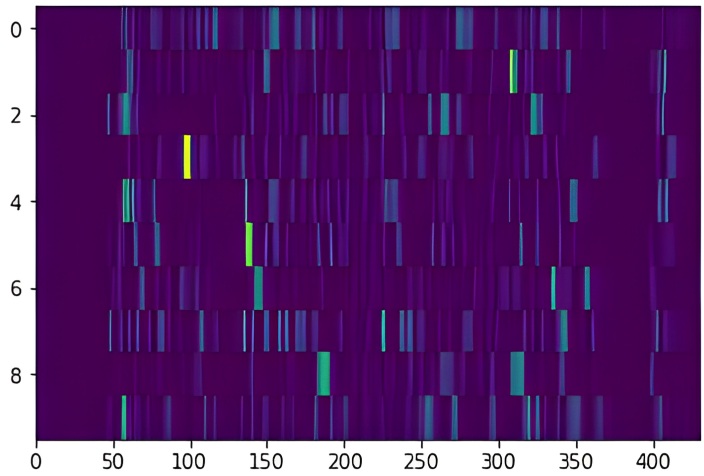
$$\begin{aligned} \text{minimize} \quad & f(x) + g(z) \\ \text{subject to} \quad & Ax + Bz = c \end{aligned} \tag{2.7}$$



(a) The input matrix V of size 1025×431, which is the spectrogram obtained by STFT.



(b) The obtained codebook W of size 1025×10.



(c) The obtained coefficient matrix H of size 10×431.

Figure 2.7: An example of NMF using the multiplicative rules.

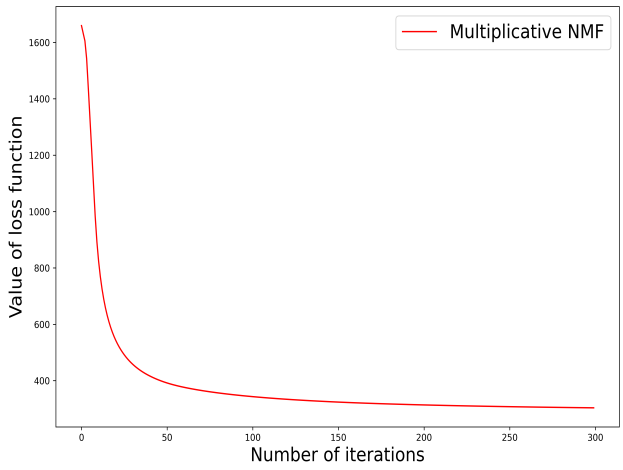


Figure 2.8: The convergence of multiplicative update rules.

Block means that the decision domain can be chunked into two groups of variables, $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$ are variables that need to be optimized. $A \in \mathbb{R}^{p \times n}$, $B \in \mathbb{R}^{p \times m}$, $c \in \mathbb{R}^p$ are convex sets, $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^m \to \mathbb{R}$ are convex functions. The objective function in Equation 2.7 is decomposable concerning two variables X and Z. ADMM decomposes the original objective function into two sub functions and optimizes the two variables alternately until the optimum solution is obtained.
The Lagrangian function of Problem 2.7 is stated as

$$L(x, z, y) = f(x) + g(z) + y^T(Ax + Bz - c) \tag{2.8}$$

In order to ensure that the Lagrangian function is strictly convex (finite value domain), the convergence of the original Lagrangian function is guaranteed even if it is a general convex function. The augmented Lagrangian methods is introduced, to yield convergence without assumptions like strict convexity of f, i.e., a squared regular term is added after the function (with a coefficient of $\frac{\rho}{2}$). The augmented Lagrangian of Problem 2.7 is

$$L_\rho(x, z, y) = f(x) + g(z) + y^T(Ax + Bz - c) + \frac{\rho}{2} \parallel Ax + Bz - c \parallel_F^2 \tag{2.9}$$

where y > 0 is the dual variable, $\rho > 0$ is the learning rate. Applying dual ascent method to the modified problem yields the algorithm

$$\begin{aligned} (x^{k+1}, z^{k+1}) &= \underset{x,z}{\operatorname{argmin}} L_\rho(x, z, y^k) \\ y^{k+1} &= y^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \end{aligned} \tag{2.10}$$

where ρ is the learning rate in dual ascent method. The ADMM iteration is shown as Equation 2.11. In which the ADMM does not optimize the main variable (x, z) jointly, but optimizes them alternatively.

$$\begin{aligned} x^{k+1} &= \underset{x}{\operatorname{argmin}} L_\rho(x, z^k, y^k) \\ z^{k+1} &= \underset{z}{\operatorname{argmin}} L_\rho(x^{k+1}, z, y^k) \\ y^{k+1} &= y^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \end{aligned} \tag{2.11}$$

where $\rho > 0$ is the learning rate for dual variables.
To make the optimization process more explicit and easy, the scaled form of augmented Lagrangian is introduced. By scaling the dual variables in the augmented Lagrangian, combining the linear and quadratic terms concerning the equation constraint, we get

$$L_\rho(x, z, u) = f(x) + g(z) + \frac{\rho}{2} \parallel Ax + Bz - c + u \parallel_F^2 \tag{2.12}$$

where $u = \frac{1}{\rho}y$, called the scaled dual variable. The scaled augmented Lagrangian objective function is equivalent to the augmented Lagrangian function, but it is more convenient to optimize and solve for the objective variables. The scaled form ADMM is

$$\begin{aligned} x^{k+1} &= \underset{x}{\operatorname{argmin}}(f(x) + \frac{\rho}{2} \parallel Ax + Bz^k - c + u^k \parallel_F^2) \\ z^{k+1} &= \underset{z}{\operatorname{argmin}}(g(z) + \frac{\rho}{2} \parallel Ax^{k+1} + Bz - c + u^k \parallel_F^2) \\ u^{k+1} &= u^k + Ax^{k+1} + Bz^{k+1} - c \end{aligned} \tag{2.13}$$

where ρ is the learning rate of dual variables. The analysis and proof of convergence can be found in [9].

2.3.2 Global Variable Consensus Optimization

The so-called global variable consensus optimization problem is a pathway for ADMM algorithms to parallel and distributed computing. In the optimization of Problem 2.7, the global objective function is the sum of objective functions with respect to two variables. In order to perform the optimization independently at each node, the global objective function needs to be decomposed for each node. In a distributed environment, if $f(x)$ is decomposable with respect to x, then the problem can be stated as

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{N} f_i(x_i) \\ \text{subject to} \quad & x_i - z = 0, i = 1, ..., N \end{aligned} \tag{2.14}$$

where $x_i \in \mathbb{R}^n$ are called local variables in the i-th node, z is often called the global consensus variable. Consensus can be viewed as a technique for turning additive objectives $\sum_{i=1}^{N} f_i(x)$, which show up frequently but do not split due to the variable being shared across terms, into separable objectives $\sum_{i=1}^{N} f_i(x_i)$, which split easily [9]. In the consensus optimization, the objective function is decomposed into N sub objective functions or subsystems, each subsystem only optimizes its local variable x_i, then all of the local variables are integrated. Combined with the dual variables, the unique global solution z is jointly optimized.
The updated global variable z will be broadcast to all nodes, and then the dual variable will be updated. The whole process is repeatedly iterated until all local variables satisfy the requirement of global variable consensus. The consensus problem can be iteratively optimized as

$$\begin{aligned} x_i^{k+1} &= \underset{x}{\operatorname{argmin}}(f_i(x_i) + (\rho/2) \parallel x_i - z^k + u_i^k \parallel_2^2) \\ z^{k+1} &= \underset{z}{\operatorname{argmin}}(g(z) + (N\rho/2) \parallel z - \bar{x}^{k+1} - \bar{u}^k \parallel_2^2) \\ u_i^{k+1} &= u_i^k + x_i^{k+1} - z^{k+1} \\ \bar{x} &= (1/N)\sum_{i=1}^{N} x_i, \quad \bar{u} = (1/N)\sum_{i=1}^{N} u_i \end{aligned} \tag{2.15}$$

The consensus optimization problem is actually distributed processing, by which the same global solution can be obtained from multiple nodes or chunks of the dataset. Therefore it is widely used in the distributed problems.

2.3.3 General Form Consensus Optimization

When we solve the global consensus optimization problem using the ADMM NMF algorithm proposed in Section 2.3.2, the local variables of each node correlate all features within the data set, i.e., both local and dual variables of each node contain the complete model parameters. When the data features are highly dimensional, the excessive amount of communication between nodes can become a bottleneck in the algorithm's performance. In many machine learning applications, each data block is associated with only partial features in the dataset, especially when the processed dataset is high-dimensional and sparse. For example, in a text classification task, each document usually consists of a subset of words or terms in the corpus, and each node only needs to be concerned about the words in its local corpus. In this case, if still using the global variable consensus optimization, there will be unnecessary communication and a waste of computational resources. Boyd et al. proposed the general form consensus optimization in [9], in which the nodes or subsystems are divided into multiple groups and each group contains only a certain node and its connected nodes. Instead of working on all model parameters, each node only needs to work on the local parameters in its block, which also can be regarded as block-wise update.

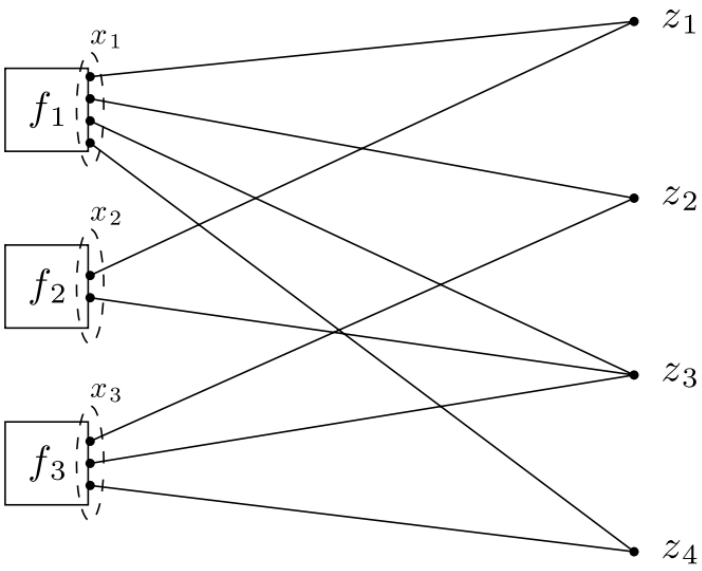


Figure 2.9: General form consensus optimization. Local objective terms are on the left; global variable components are on the right. Each edge in the bipartite graph is a consistency constraint, linking a local variable and a global variable component [9].

As shown in Figure 2.9, local objective functions $f_i(x_i)$ are on the left, global variables z_j are on the right. The edges linking them represent a constraint on consistency. The global variable consists not only of a single vector, but of $Z = \{z_1, z_2, z_3, ...z_j, j = 1, 2, 3...N\}$. z_j is the global variable for the j-th group, it is responsible for the intercommunication of nodes or subsystems within this group. For instance, z_1 in Figure 2.9 is responsible for the first and second systems, z_2 is responsible for the second and third systems. The second subsystem $f_2(x_2)$ belongs to the first and third groups, which means it needs both of z_1 and z_3 to optimize its local variables.
In a distributed system, all primal variables, dual variables, and global variables can be decomposed into M groups. Let z_j denote denote the global consensus variable in the j-th group for j $= 1, ...M$, x_i denote the local variable for the i-th node. Let Λ denote the mapping relationship between the local variables and the global variables, it contains all the (i, j) pairs such that the local variables for node x_i depends on the global variable z_j. Let $\Phi(i) = j|(i, j) \in \Phi$, which denote that node i only needs to pull the relevant groups z_j for $j \in \Phi(i)$. Let $\Phi(j) = i|(i, j) \in \Phi$, which denote that group j concerning about the nodes x_i for $i \in \Phi(j)$. The general form consensus

problem in [9] is stated as

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{N} f_i(x_i) \\ \text{subject to} \quad & x_i - z_j = 0, \quad i = 1, ..., N \\ & j = 1, ...M \quad \forall (i,j) \in \Lambda \end{aligned} \tag{2.16}$$

The augmented Lagrangian for Problem 2.16 is

$$L_\rho(x, z, y) = \sum_{i=1}^{N} f_i(x_i) + \sum_{(i,j)\in\Lambda} y_{ij}^T (x_i - z_j) + \sum_{(i,j)\in\Lambda} (\rho/2) \parallel x_i - z_j \parallel_F^2 \tag{2.17}$$

where y_{ij} is the dual variable for i-th node and j-th group. ρ is the learning rate for the dual variables. Then ADMM consists of the iterations

$$\begin{aligned} x_i^{k+1} &= \underset{x}{\operatorname{argmin}} (f_i(x_i) + \sum_{j\in\Phi(i)} (y_{ij}^{kT} (x_i - z_j^k) + \sum_{j\in\Phi(i)} (\rho/2) \parallel x_i - z_j^k \parallel_F^2 \\ z_j^{k+1} &= (1/n_j) \sum_{i\in\Phi(j)} x_i^{k+1} \\ y_i^{k+1} &= y_i^k + \sum_{j\in\Phi(i)} (\rho/2)(x_i^{k+1} - z_j^{k+1}) \end{aligned} \tag{2.18}$$

where n_j is the number of local variable entries corresponding to the global variable entry z_j. It can be seen that z-update is the local average for each group but not the global average.

2.4 Anomaly Sound Detection

The purpose of anomaly sound detection is to detect sounds that should not occur in a normal environment such as gunshots, screams, explosions, glass breakage, etc. Its tasks can be classified into two types, supervised anomaly sound detection and unsupervised anomaly sound detection. Supervised anomaly sound detection is the task of detecting defined anomaly sounds, such as gunshots or screams, which is a kind of sound event detection. While unsupervised anomaly sound detection is the task of detecting unknown anomaly sounds outside of the known dataset. In real life, anomaly sounds are diverse and occur rarely, so that it is impossible to collect an exhaustive set of anomaly sounds. So we need to detect anomaly sounds that are not present in the training data set. Therefore, anomaly sound detection tasks are usually solved as unsupervised problems. The main process of anomaly sound detection generally consists of three parts. First, the distinguished acoustic features of the normal input signal are extracted, then the features are

modeled using a suitable algorithm, which is fed into the machine learning models for training. Finally, use the trained model to classify the normal and anomalous features of audio. That is, the model learns the representation of normal sounds from existing data, and then uses the trained model to detect the unknown sound data.

2.4.1 Representation Methods of Anomaly Sound Detection

At present, two methods are mainly used for representing the acoustic features of sound. One is based on the dictionary learning, the other is based on the embedding method. To calculate the normal score, three commonly used methods are based on Gaussian mixture models, support vector machines and neural networks.
Depending on the choices of features and models, different detection schemes are developed. An example is the sound recognition by using statistical models of some classical machine learning algorithms. The Mel frequency spectrum coefficient feature is used in [20], then the anomalous sound is identified by a OC-SVM. Meanwhile, there has been widespread interest in neural network-based approaches that uses normal data to train an auto-encoder [25] or long short term memory recurrent neural networks [21].

2.4.1.1 Representation of Acoustic Features based on Dictionary Learning

The idea of dictionary learning comes from the concept of the dictionary in real life. When a given reconstruction error is satisfied, the audio features can be represented by a set of over-complete bases, i.e., Y = DX. Y is the audio feature parameter to be modeled, D is a sparse matrix called a codebook, X is the coefficient of representation of the audio feature parameter Y with respect to the sparse matrix D. Dictionaries can extract the essential features of audio signals. In the dictionary learning process, obtaining suitable codebooks is the key to the success of the algorithm. In addition to satisfying the demand of small reconstruction error, the representation coefficients X have to be as sparse as possible, so that to obtain a more concise representation of the audio features.

2.4.1.2 Representation of Acoustic Features based on Embedding Method

Neural network is a computational model that mimics the structure and function of biological neural networks. It can be used for prediction or approximate estimation of input data. It is capable of implementing complex logic operations and can handle highly nonlinear relationships, and has shown good performance in the processing of audio signals. Deep neural networks have

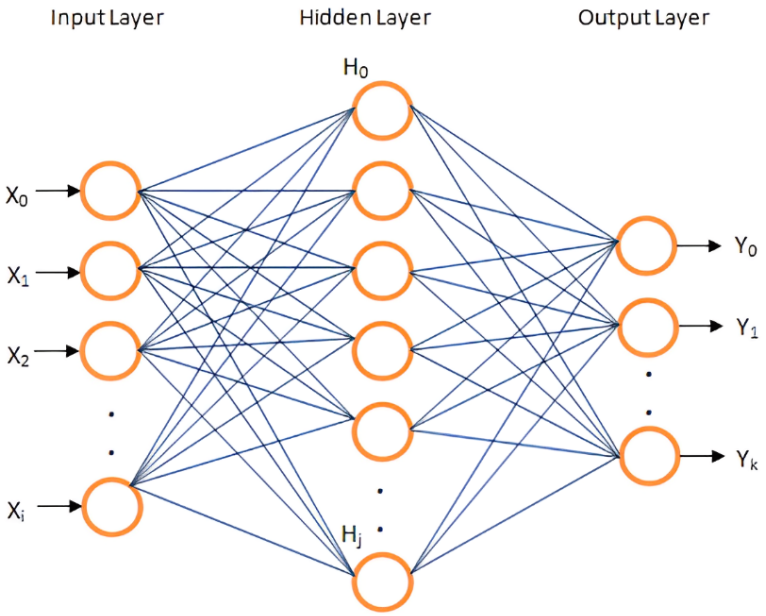


Figure 2.10: The structure of a common deep neural network [4].

a powerful feature learning capability and can learn the essential features of data from a large number of samples.

Figure 2.10 shows the structure of a common neural network, the first layer is the input layer and the last layer is the output layer. There are multiple hidden layers in between and each hidden layer contains a large number of neurons for computation. After a series of operations on the input data X inside the neural network, output the result Y.

The neural network simulates the functional relationship between the input X and the output Y without the need to know the specific details of the operations. In this way, when a certain reconstruction error is satisfied, the neural network can be used to learn the normal audio features. First, input normal audio feature X, the activation value of each neuron in each layer between the input layer and the output layer is obtained by forward propagation, until obtain the result Y. Then compute its error from the expected value, and the error gradient of each layer is calculated by back propagation, the parameters are updated from back to front. That is, the parameters inside the network are continuously corrected so that the output Y gradually approximates the input normal audio features X. The details can be found in [11].

2.4.2 Classification Methods of Anomaly Sound Detection

There are several popular methods of classifying anomaly sounds, including Gaussian mixture model [24] based on Gaussian probability density function, deep clustering methods [15] based on neural networks, and classification methods based on traditional machine learning algorithms,

such as One-Class support vector machine (OC-SVM) and Isolation Forest , which will be explained in the following.

2.4.2.1 One-Class Support Vector Machines

As mentioned above, OC-SVM is one of the commonly used traditional machine learning algorithms for the audio classification. It is mainly applicable when there is only one class of data for training and the goal is to test whether the new data is similar to the training data. As a result, it suits the anomaly sound classification a lot.
Support vector machine (SVM) is developed in the mid-1990s based on statistical learning theory proposed [32], which can be used for pattern classification and regression estimation [10].
It was originally designed for solving various two-class classification problems. Given a set of independently and identically distributed samples $X = \{(x_i, y_i), i = 1, 2, ..., n\}$, $x_i \in \mathbb{R}^d$, $y_i \in \{-1, 1\}$. If the sample x_i belongs to the first class, then it's regarded as positive, namely $y_i = +1$. If the sample x_i belongs to the second class, it is labeled as negative namely $y_i = -1$, the sample set X is called the training set.
The goal of training is to construct a decision function such that each sample in the test dataset can be classified as correctly as possible. SVM are essentially designed to solve binary classification problems, and in order to achieve this goal, both positive and negative samples are needed in the training set. However, in real life, there are many classification problems that are not binary, or even if they are binary, it is not easy to collect a training set containing both positive and negative samples. For example, one-class classification, such as outlier identification or anomaly detection.
For these reasons, OC-SVM was proposed in [26], which is suitable for data only containing positive classes. It treats the one-class problem as a special two-class classification problem, where the OC-SVM seeks a hyperplane in the feature space such that the distance between the sample points and the origin is maximized. For the anomaly audio detection, it models the distribution of normal data and the kernel function maps the input data to a higher dimensional feature vector space to provide a clearer separation between normal and anomalous data.
Given the training set $X = \{x_1, x_2, x_3, ..., x_n\}$, where x_i corresponds to the samples, the OC-SVM model will generate the mapping function from the sample space to a high dimensional feature space. It is able to build a hyperplane in the high-dimensional feature space, separating the sample points and the origin by interval. The optimal hyperplane would be found by maximizing the distance between the origin and the sample points.

As shown in Figure 2.11, ϕ is the non-linear mapping function that maps x_i to the high dimensional feature space, ρ is the interval between the origin and the sample points, w is the normal vector of the hyperplane and p is the intercept of the hyperplane. The goal is to find the optimal

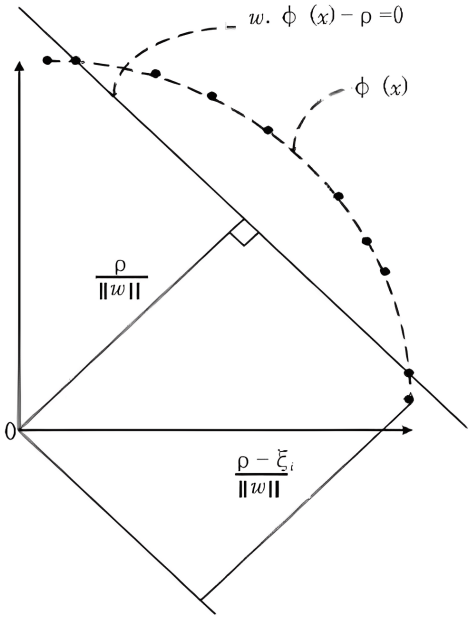


Figure 2.11: A hyperplane of the OC-SVM in 2D space.

hyperplane $w \cdot \phi(x) - \rho = 0$ that makes the distance $\frac{\rho}{\|w\|}$ maximal.
When the data used for training is linearly separable, there are infinite number of separation hyperplanes to correctly separate the two classes of data. The perceptron or neural network use mis-classification minimization to find the separation hyperplane, but there are infinite number of solutions. While SVM find the optimal separation hyperplane by maximizing the interval, which leads to a unique solution. The key points that support the separation hyperplane are called support vectors in SVM. That means, a few support vectors determine the final result, which allows one to catch the key samples, the addition or deletion of non-support vector samples does not have much impact on the model. But when the amount of noise is too large, or the noise appears in new distribution form, which differs greatly from the noise distribution in original sample set, it is probable that the noise will fall in the maximum classification interval, thus becoming a support vector, which greatly affects the model performance.

2.4.2.2 Isolation Forest

Traditional anomaly detection usually requires first constructing a normal model, then determining whether the instance conforms to this model, and whether it is anomalous. But in the real world, scenarios become more and more complex and new anomaly samples are generated all the time. The models obtained by traditional algorithms are unable to accurately identify anomalies with new types, so that the accuracy cannot be guaranteed. Thus, Liu et al. proposed the Isolation Forest algorithm in [19]. It is a fast anomaly detection algorithm based on Ensemble Learning,

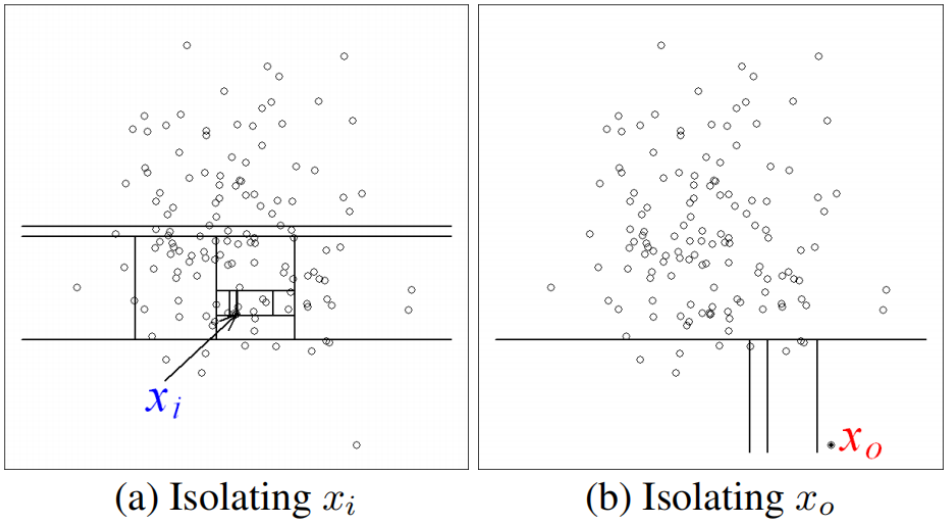


Figure 2.12: Segmentation of the random space (a) a normal point x_i requires twelve random partitions to be isolated; (b) an anomaly x_o requires only four partitions to be isolated [19].

which calculates the number of hyperplane required to isolate an instance by dividing them, and then determines the anomaly of that instance. The Isolation Forest algorithm is widely used because of its high computational efficiency and linear time complexity.
The term isolation means 'separating an instance from the rest of the instances' [19]. The core idea is to use the random hyperplane to separate data space to obtain two data sub-spaces, then use the random hyperplane to separate the data sub-spaces, and repeat the operation until there is only one data point in each subspace. Since anomalies are 'few and different', they are more susceptible to isolation [19]. As shown in Figure 2.12, the normal point x_i requires more separations than the anomaly point x_o.

The Isolation Forest algorithm uses binary trees for segmentation of data, and its samples selection, feature selection, split point selection are all randomized. The implementation of the algorithm can be divided into two steps.
First, construct an isolation forest consisting of t isolated binary trees (Isolation Tree)
(i) Randomly select n sample points from the training set as a sub-sample set and put them into the root node of the tree. (ii) Randomly select a data dimension and generate a random cut point p and a feature f in the current node. If the maximum and minimum values of all samples contained in this node correspond to the feature f are f_{max} and f_{min} respectively, then $p \in [f_{min}, f_{max}]$. (iii) Form a hyperplane based on this cut point, divide the data space of the current node into two sub-spaces, put the data whose samples have values less than p about feature f on the left side of the current node, and put the data greater than or equal to p on the right side of the current node. (iv) Repeat step (ii) and (iii) in the sub-nodes and continuously

construct new sub-nodes, when the data is not divisible or has reached the maximal depth $\log_2 n$ of the tree, the recursive process ends.
Then calculate the anomaly score for the tested samples. The training process ends after the isolation forest is formed. Since the formation of isolated binary trees is random, the result of a single tree is not reliable. So for the data sample to be tested, it is made to traverse each tree in the isolation forest. Finally the average depth of sample in each tree is obtained. The smaller the sample depth in the isolated binary tree, the higher the anomaly score, i.e., the higher the probability that the sample is an anomaly. By normalizing the length of the isolated binary tree, a number between $[0, 1]$ can be obtained as the anomaly score of the detected sample.
It can be seen that the Isolation Forest algorithm is suitable to be employed when the proportion of anomalies to the total amount of samples is small and there are significant difference between the features of anomalies and normal samples. In the anomaly sound detection, the occurrence of anomaly sounds is also rare, and their features differ significantly from those of normal sounds, so we select Isolation Forest as one of the models for training.

2.4.3 Classifier Performance Evaluation Criteria

In order to properly evaluate the performance of classification models, different metrics are proposed. Examples are Accuracy and Receiver Operating Characteristic (ROC) metric. Accuracy is the number of samples whose predictions are correct divided by the total number of samples, and measures the ability of the model to identify positive and negative samples. Accuracy can directly characterize the performance of a classifier, however, when using a dataset with unbalanced categories, this metric cannot reflect the true classification ability. For example, 1000 people are having a liver cancer diagnosis and 10 of them are confirmed to have cancer. If negative is the statistical criterion, then that accuracy rate could be as high as 99% anyway, such an evaluation criterion is meaningless. Meanwhile, ROC metric replaces accuracy as a better measure due to its insensitivity to class distribution, low error cost, intuitiveness and good comprehensibility.
An ROC curve is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters, True Positive Rate (TPR) and False Positive Rate (FPR) [1]. The prediction for binary classification problems usually produces a binary output, i.e. N or P, representing prediction as negative and prediction as positive, respectively. FPR is defined as $\frac{TP}{TP+FN}$, which is the number of samples in the true negative class that are predicted to be positive divided by the number of samples in all true negative classes. TPR is defined as $\frac{FP}{FP+TN}$, which is the number of samples in the true positive class that are predicted to be positive divided by the number of samples in all true positive classes. An ROC curve plots TPR vs. FPR at different classification thresholds. In other words, it plots the trade-off between TPR and FPR by varying the classification threshold.

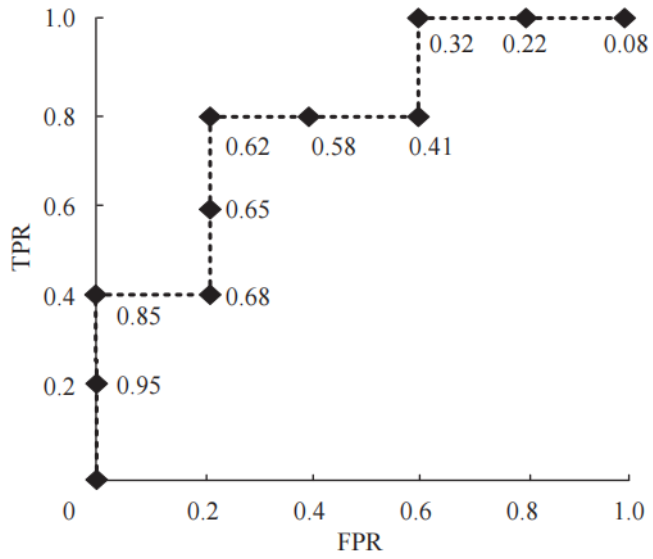


Figure 2.13: An example of ROC curve.

Each threshold represents a classifier, corresponding to a point on the ROC curve. Figure 2.13 plots the thresholds corresponding to different TPR and FPR with TPR as the y-axis and FPR as the x-axis. As the threshold decreases, the number of instances classified as positive increases, and the number of negative instances classified as positive classes also increases. That would lead to an increase in both TPR and FPR, meaning that lowering the classification threshold would classify more instances as positive. If the threshold is too low, it will result in an overly optimistic estimate of the performance, and a large number of negative instances will be misclassified as positive, which is very problematic in practice. For example, if patients are diagnosed with breast cancer and many cancer patients are misclassified as healthy, the treatment of the disease will be delayed and irreparable damage will be caused. Therefore, a sensible threshold value is important for the outcome. But it's inefficient to evaluate a logistic regression model many times to choose a perfect threshold. The area under the ROC curve (AUC) is proposed, which makes it possible to clearly and directly show the performance of the classifiers. It provides an aggregate measure of performance across all possible classification thresholds. The value of AUC ranges from zero to one. When it takes the value of zero, it proves that the model's prediction is 100% wrong, and when it takes the value of one, it proves that the model's prediction is 100% correct. The AUC measures the quality of the model predictions regardless of the classification threshold chosen. For this reason, it was chosen as the performance evaluation criterion in our experiments. Criteria for evaluating the performance of classifiers from AUC is as following. AUC $= 1$, the classifier is perfect; AUC $\in [0.85, 0.95]$, the classifier is very good; AUC $\in [0.7, 0.85]$, the classifier is good; AUC $\in [0.5, 0.7]$, the classifier is inefficient; AUC $= 0.5$, the performance of the classifier is same as random guess, it has no predictive value; AUC < 0.5, there might be a flipped prediction.

Chapter 3

Proposed Methodology

In this chapter, based on the optimization strategy of ADMM, we design and implemente the ADMM-based NMF, NMF using consensus ADMM and NMF using general form consensus ADMM for distributed problems. These algorithms are able to decompose the original problem into several sub-problems, and the global consistency constraint between nodes ensures the optimization of the global solution. While the NMF using consensus ADMM is applicable to fully connected topologies, the NMF using general form consensus ADMM is more flexible and also applicable to partially connected topologies. Then the proposed unsupervised anomaly sound detection system is presented.

3.1 Non-Negative Matrix Factorization using ADMM Algorithm

Because of the stability and ease of implementation of multiplicative NMF, it is very popular in research areas such as pattern recognition, image engineering, etc. However, the multiplicative NMF also has shortcomings, such as slow convergence, especially tail convergence; asymptotic convergence to zeros, and are especially susceptible to becoming trapped in poor local optima [29]. Figure 3.1 shows the difference of convergence between multiplicative NMF and ADMM-based NMF, when the input and number of features in the codebook are the same. The vertical coordinate is the value of the loss function during the iteration, and the horizontal coordinate is the number of iterations. The red curve shows the convergence of multiplicative NMF, it requires nearly 300 iterations to reach convergence, which results in slow computation when performing large-scale operations.

In [29], D.L. Sun et al. applied the alternating direction multiplier method to non-negative matrix factorization and proposed ADMM based Non-negative Matrix Factorization with the

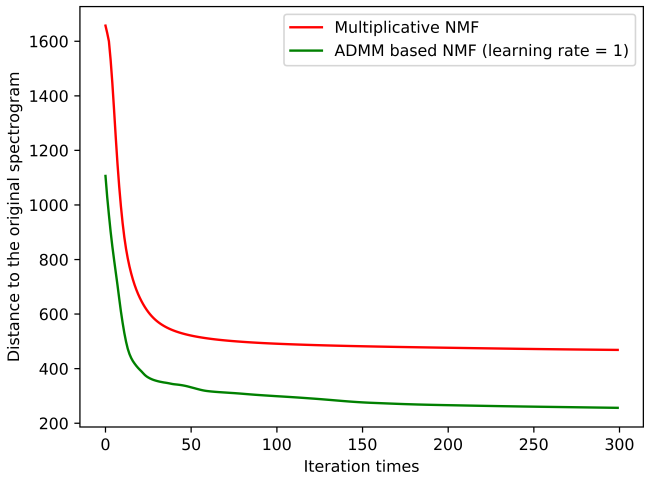


Figure 3.1: Comparison of the convergence between multiplicative NMF and ADMM based NMF.

β-divergence. The algorithm solves the problem of slow convergence, and the decomposed matrix is strongly sparse. The solution of the iterative optimization process of NMF is a convex problem, and the alternating direction multiplier method can decompose the convex problem with a divisible structure into several sub-problems to solve alternately. Based on the algorithm in [29], we replace the divergence with a simpler norm of the input matrix and the product of sub-matrices as the objective function, and derive the ADMM-based NMF update rules. The problem can be stated as

$$\begin{aligned} \text{minimize} \quad & L(X|WH) = \frac{1}{2} \parallel X - WH \parallel_F^2 + I_+(W_+) + I_+(H_+) \\ \text{subject to} \quad & W = W_+, H = H_+ \end{aligned} \tag{3.1}$$

where X is the input signal, W and H are the decomposed matrices, I_+ are the indicator functions of $\mathbb{R}^+$, which enforces the non-negativity of the variable. Our algorithm decomposes the optimization of Problem 3.1 into alternating optimization of multiple sub-problems. By adopting dual variables, it establishes the augmented Lagrangian equation of the original objective function and then alternatively update the primal and dual variables to find the optimal solution. The augmented Lagrangian is

$$\begin{aligned} L_\rho(W, H, W_+, H_+, U_W, U_H) = L(X|WH) + \rho U_W(W - W_+) + \frac{\rho}{2} \parallel W - W_+ \parallel_F^2 + \\ \rho U_H(H - H_+) + \frac{\rho}{2} \parallel H - H_+ \parallel_F^2 \end{aligned} \tag{3.2}$$

where W, H, W_+, H_+ are the primal variables, to make the iteration process more clear, we use the scaled form of dual variables U_W, U_H. ρ is the learning rate of the dual variables. The

iterative optimization process is shown in Algorithm 1.

Algorithm 1 Local ADMM NMF routine

Input X$\in \mathbb{R}_+^{n\times m}, \rho, o, M$
Output W$\in \mathbb{R}_+^{n\times o}$,$H \in \mathbb{R}_+^{o\times m}$

procedure NMF
  W^0,H^0,W_+^0,$H_+^0 \leftarrow \mathcal{U}(0,1), \quad U_W^0$,$U_H^0 \leftarrow \mathbf{0}$
  while $k < M$ **do**
    $W^{(k+1)} = [XH^{kT} + \rho(W_+^k - U_W^k)][H^k H^{kT} + \rho I]^{-1}$ ▷ Primal update
    $H^{(k+1)} = [W^{(k+1)T}W^{(K+1)} + \rho I]^{-1}[W^{(k+1)T} + \rho(H_+^k - U_H^k)]$
    $W_+^{(k+1)} = \max(W^{k+1} + U_W^k, 0)$ ▷ Projection to positives
    $H_+^{(k+1)} = \max(H^{k+1} + U_H^k, 0)$
    $U_W^{(k+1)} = U_W^k + (W^{(k+1)} - W_+^{(k+1)})$ ▷ Dual update
    $U_H^{(k+1)} = U_H^k + (H^{(k+1)} - H_+^{(k+1)})$
  end while
  return $\max(W^M, 0), \max(H^M, 0)$
end procedure

The green curve in Figure 3.1 shows us the convergence of ADMM-based NMF with a learning rate of 1. Compared to the red curve, which is the convergence curve of the multiplicative rules, its optimal solution is lower objective in the end.

3.2 Non-Negative Matrix Factorization using Consensus ADMM for Distributed Problems

In Section 3.1, ADMM-based NMF update rules are proposed. Although it has the advantage of faster convergence compared to the multiplicative update rules, it is only suitable for the case where the input contains only a single node. In audio processing tasks, sensor networks composed of multiple sensor nodes are often used to capture sounds. For example, a WASN consists of multiple nodes, each of which contains one or more microphones. All nodes are relatively independent and non-primary, and there is not a fusion center node with strong computational and storage capabilities. All nodes must be able to process information, the information exchange and sharing between them is carried out through direct connections, that is, each node needs to receive and process the information coming from its neighbors, and needs to send the processed information out. Obviously, the local ADMM NMF algorithm is not suitable for such a situation, which requires us to develop a novel algorithm designed for distributed information processing. Therefore, based on Section 2.3.2, we propose a novel NMF algorithm based on ADMM with global variable consensus optimization.

As described in Section 2.3.2, the basic idea of consensus ADMM is to optimize the individual sub-problems separately in parallel, and then aggregate the solutions of each sub-problem to obtain their mean value, obtain the solution of global variables, and update the dual variables as a whole. The global variable contains information from all nodes, which enables the communication between them. Each node has an optimal local solution, but the global solution is unique.
Now we introduce the global variables Z_W, Z_H and state equality for all sub-problems. Problem 3.1 can be stated as

$$\begin{aligned} &\text{minimize} \sum_{i=1}^{N} L(X_i|W_iH_i) + I_+(Z_W) + I_+(Z_H) \\ &\text{subject to} \quad W_i - Z_W = 0, H_i - H_{i+} = 0, i = 0, 1, 2...N \end{aligned} \tag{3.3}$$

where X = $\{X_1, X_2, X_3, ..., X_n\}$, $X_i \in X$, X represents the input data, X_i is its subset in the i-th node. W_i and H_i are the decomposed matrices in the i-th node, Z_W and Z_H are the global variables. I_+ are the indicator functions of $\mathbb{R}^+$, which enforces the non-negativity of variables. H_{i+} is non-negative. The augmented Lagrangian is

$$\begin{aligned} Ł_\rho(W_i, H_i, Z_W, Z_H, U_{Wi}, U_{Hi}) = &\sum_{i=1}^{N} L(X_i|W_iH_i) + \frac{\rho}{2} \parallel W_i - Z_W + U_{Wi} \parallel_F^2 \\ &- \frac{\rho}{2} \parallel U_{Wi} \parallel_F^2 + \frac{\rho}{2} \parallel H_i - Z_H + U_{Hi} \parallel_F^2 - \frac{\rho}{2} \parallel U_{Hi} \parallel_F^2 \end{aligned} \tag{3.4}$$

where W_i, H_i are local variables, Z_W, Z_H are global variables, U_{Wi}, U_{Hi} are the scaled form of dual variables, ρ is the learning rate of the dual variables. The scaled form of the augmented Lagrangian is

$$\begin{aligned} W_i^{(k+1)} &= \arg\min_{W_i}(L(X_i|W_iH_i)) + \frac{\rho}{2} \parallel W_i - Z_W^k + U_{Wi}^k \parallel_F^2 \\ H_i^{(k+1)} &= \arg\min_{W_i}(L(X_i|W_iH_i)) + \frac{\rho}{2} \parallel H_i - Z_H^k + U_{Hi}^k \parallel_F^2 \\ Z_W^{(k+1)} &= \arg\min_{Z_W}(I_+(Z_W)) + \frac{N\rho}{2} \parallel Z_W - \bar{W}^{(k+1)} - \bar{U}_W^k \parallel_F^2 \\ Z_H^{(k+1)} &= \arg\min_{Z_H}(I_+(Z_H)) + \frac{N\rho}{2} \parallel Z_H - \bar{H}^{(k+1)} - \bar{U}_H^k \parallel_F^2 \\ U_{Wi}^{(k+1)} &= U_{Wi}^k + W_i^{(k+1)} - Z_W^{(k+1)} \\ U_{Hi}^{(k+1)} &= U_{Hi}^k + H_i^{(k+1)} - Z_H^{(k+1)} \end{aligned} \tag{3.5}$$

where N is the number of nodes. The iterative optimization process is shown in Algorithm 2.

With this algorithm, each node in the network can communicate with each other and perform parallel computations independently. Through iterative updates, it can converge to a unique

Algorithm 2 Global ADMM NMF routine

Input X$\in \mathbb{R}_+^{n\times m}, \rho, o, M$
Output W$\in \mathbb{R}_+^{n\times o}$, $H \in \mathbb{R}_+^{o\times m}$

procedure NMF
  W_i^0,H_i^0,Z_W^0,$Z_H^0 \leftarrow \mathcal{U}(0,1)$, U_{Wi}^0,$U_{Hi}^0 \leftarrow \mathbf{0}$
  while $k < M$ **do**
    $W_i^{(k+1)} = [X_i H_i^{kT} + \rho(Z_W^k - U_{Wi}^k)][H_i^k H_i^{kT} + \rho I]^{-1}$
    $H_i^{(k+1)} = [W_i^{(k+1)T} W_i^{(k+1)} + \rho I]^{-1}[W_i^{(k+1)T} X_i + \rho(Z_H^k - U_{Hi}^k)]$ ▷ Primal update
    $\bar{W} = \frac{1}{N}\sum_{i=1}^N W_i$
    $\bar{H} = \frac{1}{N}\sum_{i=1}^N H_i$ ▷ Average Primal Variables
    $Z_W^{(k+1)} = max(\bar{W}^{(k+1)} + \bar{U}_W^k, 0)$
    $Z_H^{(k+1)} = max(H_i^{(k+1)} + U_{Hi}^k, 0)$ ▷ Global update
    $U_{Wi}^{(k+1)} = U_{Wi}^k + (W_i^{(k+1)} - Z_W^{(k+1)})$
    $U_{Hi}^{(k+1)} = U_{Hi}^k + (H_i^{(k+1)} - Z_H^{(k+1)})$ ▷ Dual update
    $\bar{U}_W = \frac{1}{N}\sum_{i=1}^N U_{Wi}$
    $\bar{U}_H = \frac{1}{N}\sum_{i=1}^N U_{Hi}$
  end while
  return $\max(W^M, 0), \max(H^M, 0)$
end procedure

global optimal solution. However, this algorithm is only applicable to fully connected topology because the global variables Z_W and Z_H contain information on all nodes. The actual sensor network contains various topologies, besides the fully connected topology, there are also partial connected topologies such as ring topology, star topology, etc. The nodes in these topologies only exchange information with part of other nodes. In this case, each node and the nodes it communicates with form different groups, and the global variables should not contain information about all nodes, but only concern about the nodes inner the group. Therefore, based on section 2.3.3, we develop a more flexible NMF algorithm based on general form ADMM consensus, which is also applicable to partially connected topologies.

3.3 Non-Negative Matrix Factorization using General Form Consensus ADMM for Distributed Problems

The algorithm described in Section 2.3.3 optimized the global variables by grouping all variables. Its basic idea is to divide the nodes in the network topology into several groups. Each group has a global variable, whose update is only related to the local variables and dual variables in this group. Each node belongs to several groups, which means the update of its local variables depends on the global variables of these groups. Figure 3.2 shows an example of grouping the nodes in a ring topology. The degree of it is 2, each node only communicates with its left and

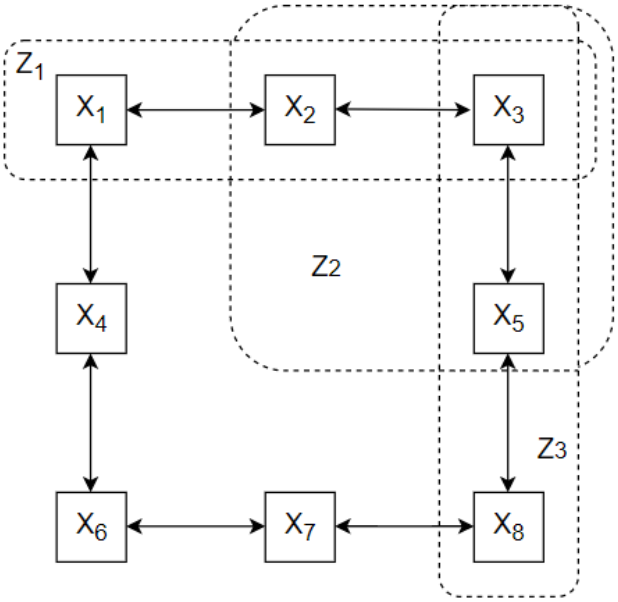


Figure 3.2: Interconnection in the partial connected topology.

right neighbors. Each three neighboring nodes are divided as a group with a global consensus variable z_j. Three groups are marked in Figure 3.2. In each group there is a certain node and its connected neighboring nodes, for example in the group with global variable z_2, x_2 can communicate with both x_1 and x_3. Similarly, each node belongs to multiple groups. For instance, x_3 belongs to z_1, z_2 and z_3. The NMF problem in such a topology can be stated as

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{N} L(X_i|W_iH_i) + I_+(Z_{Wj}) + I_+(Z_{Hj}) \\ \text{subject to} \quad & W_i - Z_{Wj} = 0, \quad i = 1, ..., N \\ & j = 1, ...M \quad \forall (i,j) \in \Lambda \end{aligned} \tag{3.6}$$

where Z_{Wj} and Z_{Hj} denote the global consensus variables in the j-th group for j $= 1, ...M$, W_i and H_i denote the local variables for the i-th node. Λ denote the mapping relationship between the local variables and the global variables, it contains all the (i, j) pairs such that the local variables for node X_i depends on the global variable Z_j. I_+ are the indicator functions of $\mathbb{R}^+$, which enforces the non-negativity of variables. Let $\Phi(i) = j|(i, j) \in \Phi$, which denote that node i only needs to pull the relevant groups z_j for $j \in \Phi(i)$. Let $\Phi(j) = i|(i, j) \in \Phi$, which denote that group j concerning about the nodes x_i for $i \in \Phi(j)$. The scaled form augmented Lagrangian of

Problem 3.6 is

$$\begin{aligned} L_\rho(W_i, H_i, Z_W, Z_H, U_{Wi}, U_{Hi}) = \sum_{i=1}^{N} L(X_i|W_iH_i) + \frac{\rho}{2} \sum_{(i,j)\in\Lambda} \parallel X_i - Z_{Wj} + U_{Wij} \parallel_F^2 \\ - \frac{\rho}{2} \sum_{(i,j)\in\Lambda} \parallel U_{Wij} \parallel_F^2 + \frac{\rho}{2} \sum_{(i,j)\in\Lambda} \parallel X_i - Z_{Hj} + U_{Hij} \parallel_F^2 \\ - \frac{\rho}{2} \sum_{(i,j)\in\Lambda} \parallel U_{Hij} \parallel_F^2 \end{aligned} \tag{3.7}$$

where W_i, H_i are the primal variables for i-th node, U_{Wij}, U_{Hij} are the scaled form for i-th node and j-th group. ρ is the learning rate of dual variables. Let $\Phi(i) = j|(i,j) \in \Phi$, which means that node i only needs to pull the relevant groups z_j for $j \in \Phi(i)$. Let $\Phi(j) = i|(i,j) \in \Phi$, which means that group j concern about the nodes x_i for $i \in \Phi(j)$. The iterative optimization process is shown in Algorithm 3. n_j is the number of local variable entries corresponding to the global variable entry z_j.

Algorithm 3 General Form consensus ADMM NMF routine

Input X$\in \mathbb{R}_+^{n\times m}, \rho, o, M$
Output W$\in \mathbb{R}_+^{n\times o}$, $H \in \mathbb{R}_+^{o\times m}$

procedure NMF
  W_i^0,H_i^0,Z_W^0,$Z_H^0 \leftarrow \mathcal{U}(0,1)$, $\quad U_{Wi}^0$,$U_{Hi}^0 \leftarrow \mathbf{0}$
  while $k < M$ **do**
    $W_i^{(k+1)} = [X_i H_i^{kT} + \rho \sum_{j\in\Phi(i)} (Z_{Wj}^k - U_{Wij}^k)][H_i^k H_i^{kT} + \rho I]^{-1}$
    $H_i^{(k+1)} = [W_i^{(k+1)T} W_i^{(k+1)} + \rho I]^{-1}[W_i^{(k+1)T} X_i + \rho \sum_{j\in\Phi(i)} (Z_{Hj}^k - U_{Hij}^k)]$
    ▷ Primal update
    $Z_{Wj}^{(k+1)} = max(\frac{1}{n_j} \sum_{i\in\Phi(j)} W_i^{(k+1)} + \frac{1}{n_j} \sum_{i\in\Phi(j)} U_{Wij}^k, 0)$
    $Z_{Hj}^{(k+1)} = max(H_i^{(k+1)} + U_{Hi}^k, 0)$
    ▷ Global update
    $U_{Wij}^{(k+1)} = U_{Wij}^k + (W_i^{(k+1)} - Z_{Wj}^{(k+1)})$
    $U_{Hij}^{(k+1)} = U_{Hij}^k + (H_i^{(k+1)} - Z_{Hj}^{(k+1)})$
    ▷ Dual update
  end while
  return $\max(W^M, 0), \max(H^M, 0)$
end procedure

3.4 The proposed Unsupervised Anomaly Detection System

Unsupervised anomaly detection is the task of modeling and classifying data in a dataset by unsupervised algorithms to detect anomalies. Depending on the training data in the dataset, unsupervised anomaly detection can be categorized broadly into two types. When both normal

and anomalous data are available in the dataset but the labels are unknown, and when only normal data are available in the dataset. In real life, there is a wide variety of anomalous sounds, it is hard to collect all of them for training. Therefore, we design an unsupervised anomaly sound detection system based on the non-negative matrix factorization using consensus ADMM algorithm proposed in Section 3.2. It aims to extract a codebook from the acoustic features of normal sounds and training with suitable classifiers, then perform anomaly sound detection on the trained model.

The framework of the unsupervised anomaly sound detection system proposed in this thesis is shown in Figure 3.3. The system consists of three parts, feature extraction, feature representation using the NMF algorithm, and anomaly detection with a classifier.

In the training pipeline, the spectral features of normal audio from the training set are firstly extracted, typical acoustic features such as STFT spectrogram and Mel filterbank features are chosen. Then the codebook of the spectral features are extracted by the NMF algorithm, which serves as the input of classifiers. In the prediction pipeline, similar with in the training pipeline, the spectral features of all audio files (normal and anomalous sounds) in the test set are firstly extracted. These features are fed into the NMF algorithm and we can get their codebooks. Then the codesbooks are fed into the trained classifier, the anomaly scores will be calculated for all audio files. ROC curves and AUC are used for evaluating the classification performance under different settings.

Figure 3.4 shows us the details of the i-th node in the designed system. The features X_i of the input signal Y_i is extracted with the methods of STFT or Mel filter banks, then the features are input into the NMF algorithm to get the codebook W_i. In the meantime, the node exchanges information with its left and right neighbors for global consistency, i.e., pass its codebook W_i to others and receive the codebooks W_{i-1} and W_{i+1}, which are generated by its neighbors. Then the codebooks are fed into the trained model, after the prediction we can obtain the anomaly score of the input signal Y_i. The prediction occurs on the edge device.

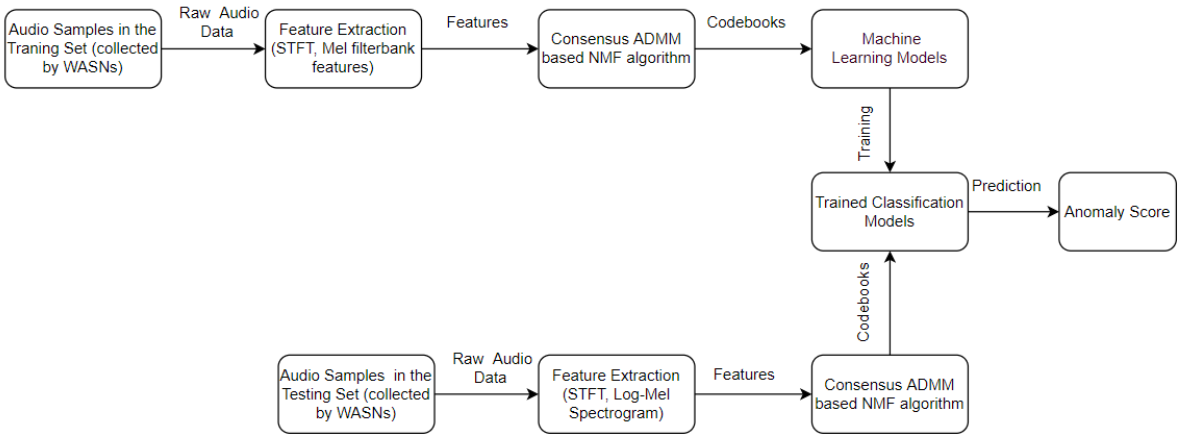


Figure 3.3: Framework of anomaly sound detection system.

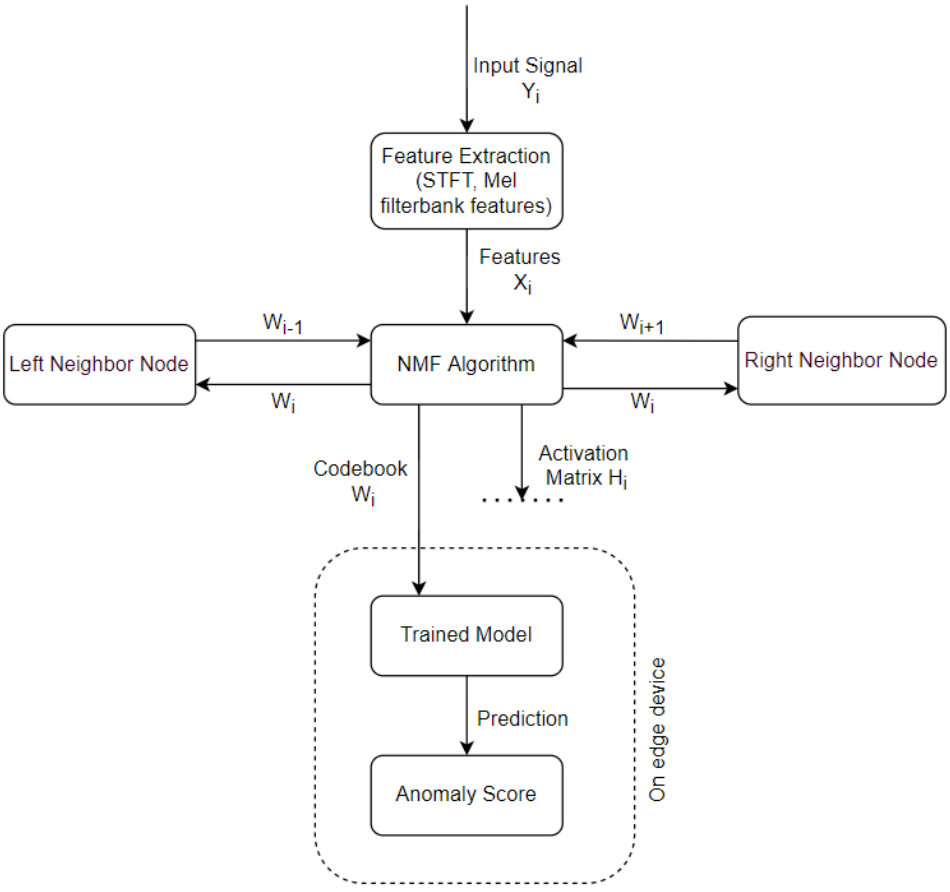


Figure 3.4: Flow of anomaly sound detection on one node.

Chapter 4

Experiments

In this chapter, the overall experimental scheme and setup are introduced, including the data sets used in the experiments and their pre-processing, as well as the experimental parameter settings in the evaluation. Then the results of part of the experiments are shown and analyzed in detail. Finally, we discuss the overall evaluation and the problems in experiments.

4.1 Experimental Design and Setup

The proposed algorithm is evaluated in a Python-based environment using the PyTorch and SciPy libraries. The dataset used for experiments is the SINS database [13]. It was preprocessed before experiments started. The noise used in experiments is from the noise database DEMAND [30]. The machine learning models used in experiments include SGDOneClassSVM and Isolation Forest, both of which are from the scikit-klearn library [23]. In the experiments, we changed the sample rate of original audio samples and used Short Time Fourier Transform and Mel filterbank to extract features from them, and use them as input to the proposed algorithm. The audio is augmented by mixing additional background noise and changing the sample rate. Further number of nodes is varied, as well as the network topology.

4.1.1 SINS Database

SINS is a database of real-life audio, recorded in a home environment over a period of a single week. The home environment used for data collection consisted of five different rooms, a combined living room and kitchen, bathroom, toilet, bedroom. The device used for recording is an acoustic sensor network containing thirteen sensor nodes, each node has four low-cost microphones. Theses nodes distributed uniformly in five rooms as shown in Figure 4.1. It focuses on daily activities

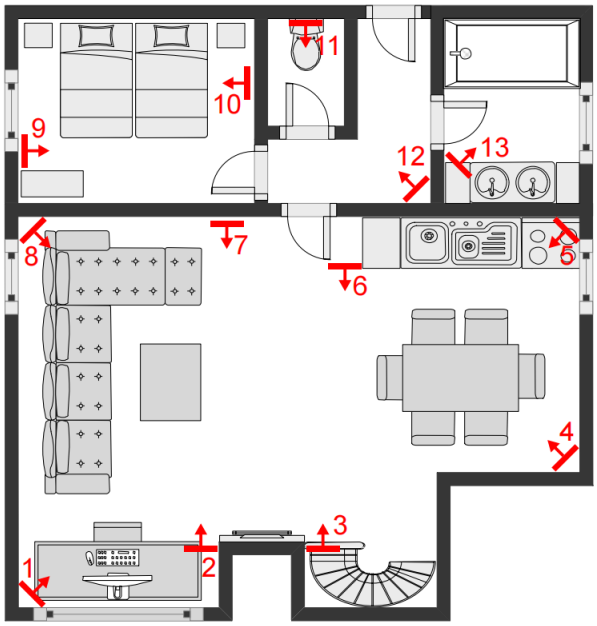


Figure 4.1: Floor map of the recording environment [13].

and contains activities being performed in a spontaneous manner, all audios are recorded as a continuous stream. All audio channels were sampled at a rate of 16 kHz. The acquired data is sent to a Raspberry Pi 3 for data storage [13]. As shown in Figure 4.2, there are 16 annotated activities for one person living in this home. It also shows the quantity of different activities and the examples, as well as the mean and standard deviation of the duration of all examples. However, the number of these activities is highly unbalanced, which reflects the imbalance of different activities in real life.
Since the original audios in the database are not strictly synchronized, the original audios need to be pre-processed first based on the annotations provided by the paper [13]. The sample rate of the processed audio is 16 kHz, the maximum length is about 60 seconds, and each audio sample contains 12 channels representing 12 nodes, because the fifth node in the original database is missing.

The rest of the samples are divided into a training dataset and a prediction dataset at random. To ensure alignment, all samples are constructed to a length of 480 k samples (30 s) before feature extraction. In our experiments, 672 samples in the training set are randomly selected for training the machine learning model. All training samples are considered as positive, namely inliers. The dataset used for prediction consisted of 48 negative samples, namely outliers, they belong to vacuum cleaning activities and 34 randomly selected inliers from the prediction dataset. This meets the criterion that the ratio of the training to the testing data should be 8:2.

Room	Activity	Nr. ex.	duration (min.)
Living room	Phone call	22	8.17±13.73
	Cooking	19	16.62±9.49
	Dishwashing	15	6.37±1.49
	Eating	19	7.78±4.27
	Visit	9	13.3±12.11
	Watching TV	13	155.38±93.28
	Working	49	31.24±39.33
	Vacuum cleaning	13	4.79±2.14
	Other	200	0.75±0.95
	Absence	72	66.37±130.30
Bathroom	Drying with towel	10	1.67±0.28
	Shaving	13	1.91±1.46
	Showering	10	6.11±2.38
	Toothbrushing	19	1.41±0.25
	Vacuum cleaning	9	0.87±0.59
	Other	75	0.42±0.4
	Absence	35	248.56±263.62
Hall	Vacuum cleaning	9	3.31±1.11
	Other	164	0.36±0.22
	Absence	175	50.17±102.52
Toilet	Toilet visit	21	4.74±3.24
	Vacuum cleaning	7	0.53±0.07
	Absence	31	282.75±263.19
Bedroom	Dressing	28	1.53±1.10
	Sleeping	7	348.43±130.73
	Vacuum cleaning	7	1.04±0.27
	Other	22	0.27±0.23
	Absence	22	122.28±157.43

Figure 4.2: Recorded activities for each room [13].

4.1.2 DEMAND Database

To evaluate the performance of our algorithm in the presence of background noise, we need to mix noise with the audio samples. Therefore, we choose DEMAND Database as the library to provide the background noise. The DEMAND (Diverse Environments Multichannel Acoustic Noise Database) is a database of 16-channel environmental noise recordings. It provides a set of recordings that allow evaluation of the algorithm using real-world noise in a variety of settings. It contains 15 recordings, all recordings are generated with a 16-channel microphone array and are available as 16 single-channel wav files in one directory. Both 48 kHz and 16 kHz sampling rates are available [30]. In our experiments, we use the recordings with a sampling rate of 16 kHz. The database includes six categories, four of which are in internal environments and two of which are in the open-air. The internal environments are divided into Domestic, Office, Public,

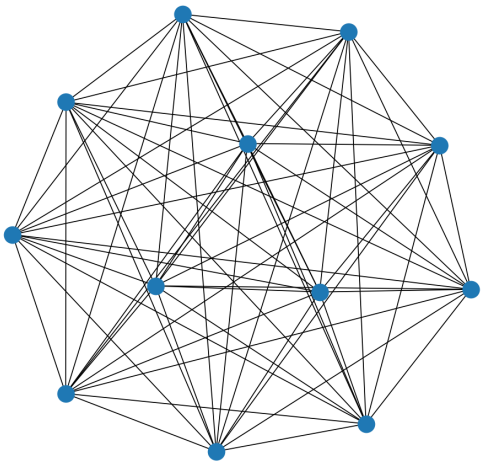


Figure 4.3: Fully connected network with 12 nodes.

and Transportation. The open-air environments are Street and Nature. A detailed description can be found in [30].
A recording of the DEMAND dataset is chosen at random and its first channel is added to all channels of the original audio. This adds uncorrelated noise with a fixed signal-to-noise ratio of 20dB.

4.1.3 Network Topology and Distribution of Acoustic Nodes

In wireless sensor networks, the bidirectional information exchange between nodes is generally done through direct connection. The different ways of connection between nodes are called topologies. In our research we choose two different topologies. One is the fully-connected topology with a degree of N -1, N is the number of nodes, which means each node can communicate with all of the other nodes. The other one is the ring topology with a degree of 2, which means it only communicates with its left and right neighbors. The structure of fully-connected topology is shown in Figure 4.3, each circle represents a node, and each node can communicate directly with other nodes, which means that the whole network is interconnected and does not need any repeater for relaying information. The structure of the ring topology with a degree of 2 is shown in Figure 4.4. It is a partially connected wireless sensor network, each node can communicate directly with its left and right neighboring nodes only.

4.1.4 Experimental Parameters Setting

To evaluate the performance of the anomaly detection system based on the proposed algorithm, we performed multiple experiments with different parameters. Using different prediction models namely OC-SVM and Isolation Forest, changing the sampling rate of original audio, using different

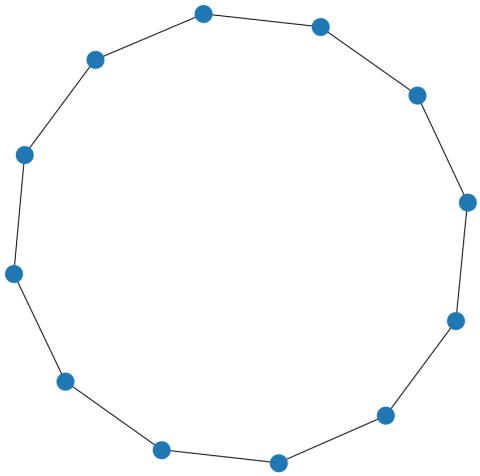


Figure 4.4: Ring network with 12 nodes.

methods of feature extraction, mixing the original audio and random background noise in the time domain as new inputs and compare the performance with the case without background noise, using different network topologies, changing the number of nodes in the network.
The experimental parameters are shown in following. (i) Pre-processing of the data. Since the input audio samples has different length, they are firstly cropped to the same length of 30 seconds with the original sampling rate of 16 kHz. In addition to the original audio samples , the raw audio is resampled at a sampling rate of 8 kHz. Besides, the original input audio and the randomly selected noise from DEMAND database are mixed in the time domain with a fixed SNR of 20 dB.
(ii) Feature extraction. We choose the spectrogram obtained with STFT and Mel filerbank features as the acoustic features. In STFT, the length of FFT was set to 512, the Hamming window was selected and the normalization of the data was performed. For the Mel filterbank, we use 20 Mel filters, the length of FFT is also set to 512, Hamming window is selected and the data is also normalized so that we can compare the results with those of using STFT.
(iii) Network structure. The fully connected topology and the ring topology with a degree of 2 are chosen in our experiments. In addition to selecting the audio recorded by all 12 nodes in the home environment as input, the audio recorded only by 6 nodes distributed in the living room and kitchen in Figure 4.1 are also selected, so that we can compare the impact of network structure and distribution of nodes on the algorithm performance.
To make the evaluation of the experiments more convincing, the same training dataset and test dataset are used in all experiments. We regard the NMF algorithm to reach convergence at a number of iterations of 100, with the learning rate $\rho = 1.0$ during the iterations, the number of columns of codebooks is set to 40. The codebooks with 40 features are used as input for training the model and prediction. To speed up the computation, the batch size is set as 21. The ADMM iteration process runs on GeForce 1060, the model training and prediction are performed on the

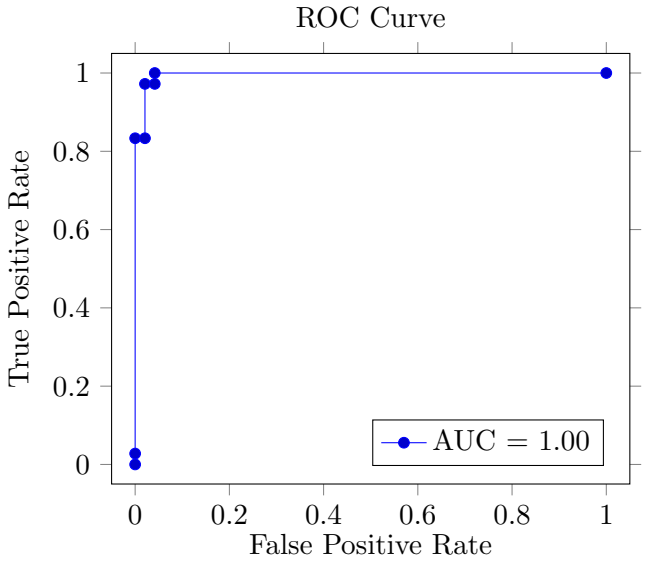


Figure 4.5: The ROC curve of baseline using OC-SVM.

CPU.

4.2 Experimental Results

The AUC metric is used to compare different experiments. We evaluate the performance of proposed algorithms under parameters introduced in section 4.1. We have two pipelines, the one is combining the NMF with OC-SVM, the other is combining the NMF with Isolation Forest. The anomaly detection runs on all nodes, after the prediction we integrate the results of all nodes to generate the ROC curves.

4.2.1 Evaluation of the Anomaly Sound Detection System using One-Class SVM

First, we establish a baseline as the reference. We use the spectrogram obtained by performing STFT on the original audio as the initial extracted features, as well as a network structure with a fully connected topology which considers all 12 nodes, and choose OC-SVM as the training model. The obtained results are shown in Figure 4.5. The AUC is about 1.00, which indicates that all of the prediction are correct, our classifier is close to perfect.

Table 4.1: Experiment settings with different network structures.

Experiments	Topology	Number of Nodes
A	Fully-Connected	12
B	Fully-Connected	6
C	Ring	12
D	Ring	6

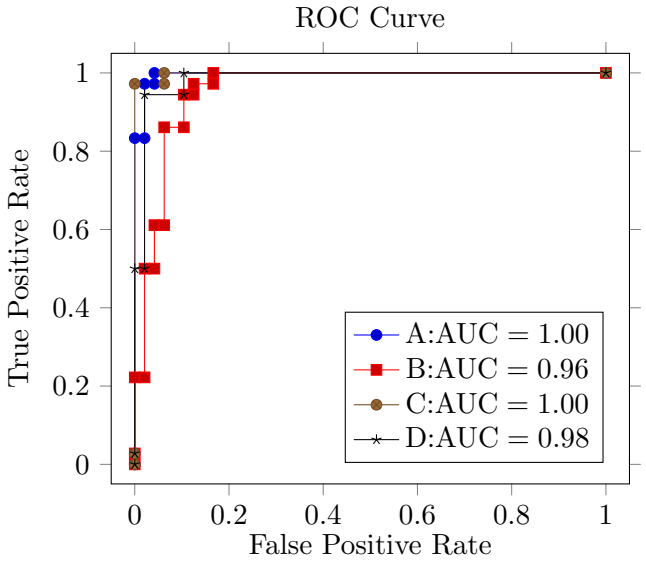


Figure 4.6: The ROC curve of different network structures. The parameter settings are shown in Table 4.1. Experiment A is the baseline.

4.2.1.1 The Effect of Different Network Structures

To investigate the effect of variations in the network structure such as different topologies and changing the number of nodes on the performance of the classifier, we conduct experiments using four different parameter settings under the premise of using OC-SVM as the detection model. The input signals are all sampled at 16 kHz without adding background noise, and STFT is performed to obtain preliminary features.

Comparing the results of Experiment A and Experiment C, it is clear that when all other parameters remained unchanged and only the topology was switched from fully-connected to ring, the AUC is kept at 1.00 and the classifier performance does not degrade. This shows that the NMF algorithm, we developed, is independent of the network topology. It can extract global features well even if each node only communicates with the left and right neighbors. Although it works for both topologies, it may take longer to converge for ring topology. Experiments A and C contain all 12 virtual nodes in the experimental environment, Experiments B and D choose 6 nodes distributed in the living room and kitchen, while the sound sources always located in the

Table 4.2: Experiment settings with different acoustic features

Experiments	Topology	Acoustic Features
A	Fully-Connected	STFT Spectrogram
B	Fully-Connected	Mel Filterbank Features
C	Ring	STFT Spectrogram
D	Ring	Mel Filterbank Features

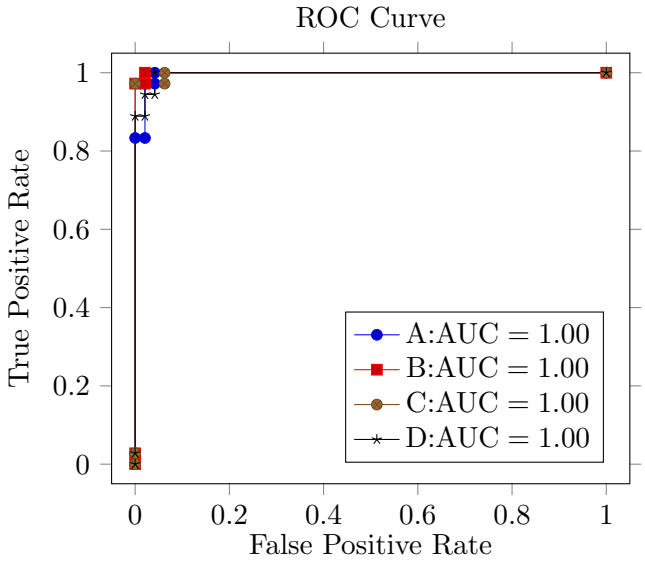


Figure 4.7: The ROC curve of different acoustic features. The parameter settings are shown in Table 4.2.

same position. Comparing the results of Experiment A and B with those of Experiment C and D, it can be seen that there is a slight decrease in the performance of the classifier when the number of nodes in the network is halved. The possible reason could be that the missing nodes also contain some feature information. But this does not have a significant impact on the overall prediction, the performance of the classifier is still excellent.

4.2.1.2 The Effect of Different Acoustic Features

In this section, we use OC-SVM as the detection model with different acoustic features as the input to the NMF algorithm, to investigate the effect of different audio pre-processing methods on the classifier performance. The original input signals are sampled at 16 kHz, no background noise is added, all 12 microphone nodes in the home environment are considered. All results are shown in Figure 4.7.

The results show that all four settings yielded an AUC of 1.00. This indicates that, without

Table 4.3: Experiment settings with different sample rate

Experiments	Input Features	Sample Rate
A	STFT Spectrogram	16kHz
B	Mel filterbank features	16kHz
C	STFT Spectrogram	8kHz
D	Mel filterbank features	8kHz

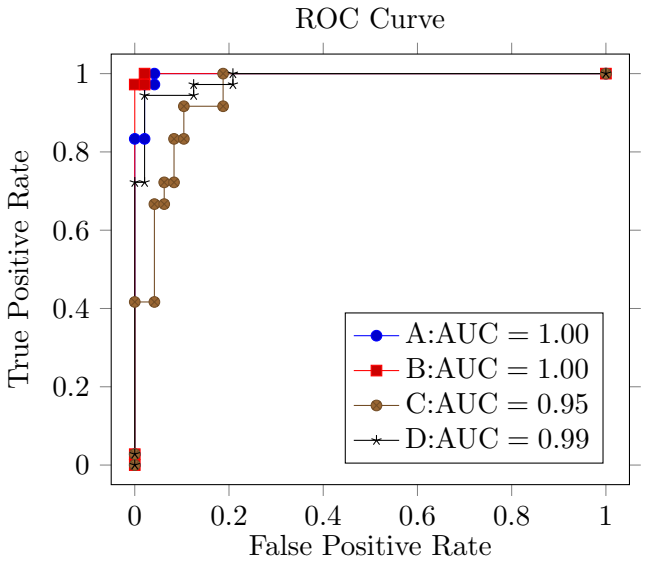


Figure 4.8: The ROC curve of different sample rate. The parameter settings are shown in Table 4.3.

changing the sampling rate, using the spectrogram obtained from STFT as the input feature or using the Mel filterbank features as the input feature does not affect the performance of the classifier. Also, the change in topology does not affect the performance when using Mel filterbank features as input.

4.2.1.3 The Effect of Different Sample Rate

The audio sample rate measures the number of samples captured per second from a continuous digital signal, measured in kHz. It determines the range of frequencies captured in digital audio. The sample rate of the original audio samples is 16 kHz. To investigate the effect of reducing the sampling rate on the classifier, we resample the input audio to 8 kHz and compare the performance of the classifier at different settings. All experiments in this section are performed using OC-SVM as the detection model and fully-connected topology. All results are shown in Figure 4.8.

Table 4.4: Experiment settings with different sample rate (6 nodes)

Experiments	Input Features	Sample Rate
E	STFT Spectrogram	16kHz
F	Mel filterbank features	16kHz
G	STFT Spectrogram	8kHz
H	Mel filterbank features	8kHz

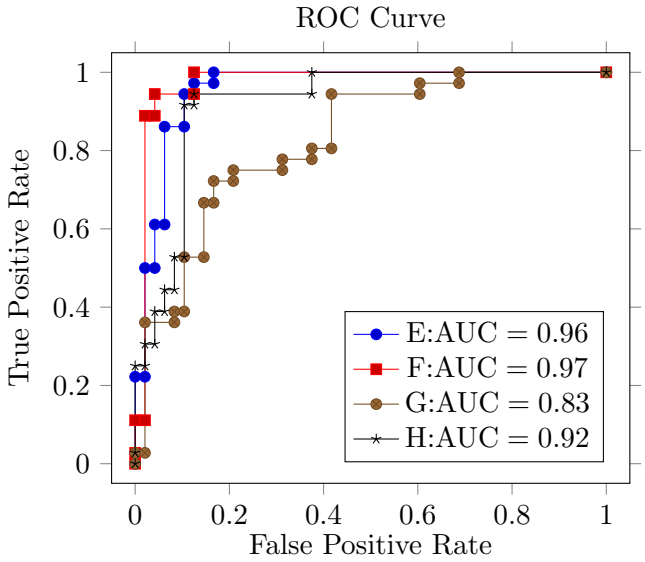


Figure 4.9: The ROC curve of different sample rate when using 6 nodes. The parameter settings are shown in Table 4.4.

By comparing the results of Experiment A with those of Experiment C, it is clear that reducing the sampling rate from 16 kHz to 8 kHz degrades the performance of the classifier when STFT is used (from AUC = 1.00 to AUC = 0.95). From the results of Experiment B and Experiment D, we can see that the effect of halving the sampling rate is less pronounced, when using Mel filterbank features.

To confirm that this conclusion is not isolated, we also perform Experiments E, F, G, H, which only consider the six microphone nodes in the living room and kitchen. All results are shown in Figure 4.9.

Comparing the results of Experiment E and Experiment G, it can be seen that the AUC decreases from 0.96 to 0.83 when the sampling rate is halved. The possible reason is that due to both the number of nodes and the sampling rate being halved, fewer effective features can be extracted from the audio, causing a decrease in the performance of the classifier. The results of Experiment F and Experiment H show that halving the sampling rate has less influence on the classifier performance when using the Mel filterbank features as the input feature.

Table 4.5: Experiments with and without noise

Experiments	Input Features	Noise
A	STFT Spectrogram	Without Noise
B	Mel filterbank features	Without Noise
C	STFT Spectrogram	With Noise (SNR = 20dB)
D	Mel filterbank features	With Noise (SNR = 20dB)

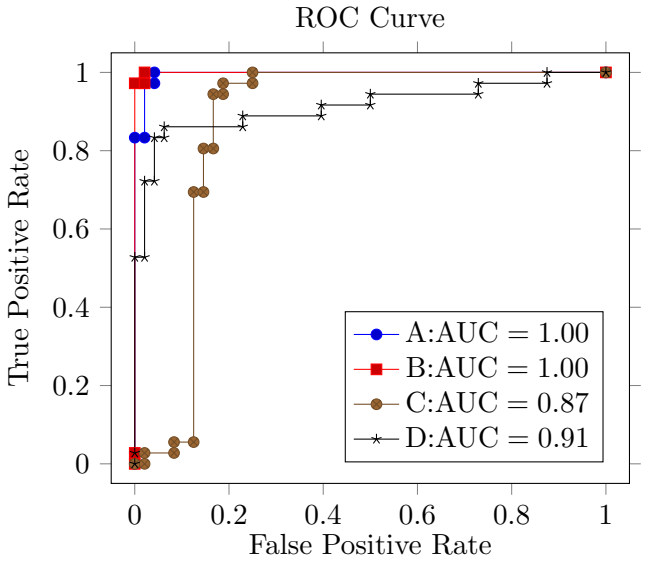


Figure 4.10: The ROC curve of experiments with and without additional noise. The parameter settings are shown in Table 4.5.

4.2.1.4 The Effect of Additional Background Noise

To investigate the effect of noise, we mix noise with the SNR of 20 dB and the original audio, then perform experiments to compare with the experiment results using the original audio samples. The noise used in the experiments is obtained from the DEMAND noise dataset in Section 4.1.2. All experiments in this section are performed using OC-SVM as the detection model and fully-connected topology with 12 nodes. The sample rate of all audio samples is 16 kHz. All results are shown in Figure 4.10.

Results in Figure 4.10 clearly show that when using OC-SVM as the classification model and STFT as the method to obtain features, noise has a significant impact on the classifier performance, with the AUC dropping from 1.00 to 0.87. One possible reason for this is that SVM is sensitive to noise and susceptible to noise whose features differs significantly from the features in the original dataset, as described in Chapter 2.4.2.1.

Table 4.6: Experiment settings with different network structures using Isolation Forest.

Experiments	Topology	Number of Nodes
A	Fully-Connected	12
B	Fully-Connected	6
C	Ring	12
D	Ring	6

4.2.2 Evaluation of the anomaly sound detection system using Isolation Forest

SVM algorithms have a time complexity of either $O(m^3)$ or $O(m^2)$ [31], where m is the training set size. OC-SVM is more suitable for small and medium-sized dataset than large dataset. As an efficient unsupervised anomaly detection algorithm, Isolation Forest has linear time complexity with excellent accuracy and can run efficiently on large-scale datasets. Since each tree is generated independently, it can be deployed on large-scale distributed systems to speed up operations. As a result, in addition to OC-SVM, we also use Isolation Forest as the training model for experiments in this section.

4.2.2.1 The effect of different network structure with Isolation Forest

Similar to the OC-SVM evaluation, we first explore the effect of changes in topology and the number of nodes within the wireless network on the performance of the Isolation Forest model. All experiments in this section are performed using Isolation Forest as the classification model, with STFT spectrogram as acoustic features, and the sample rate is 16 kHz, no additional background noise. All results are shown in Figure 4.11.

The results comparing Experiment A and Experiment C show that whether using the fully-connected topology or ring topology, the AUC is 1.00, which meets the definition of a perfect classifier. This indicates that even if the detection model changes, the changes in topology still do not affect the performance of our algorithm. By comparing Experiment A and C or Experiment B and D, we can see when the number of nodes is reduced from 12 to 6, there is a slight decrease in the detection accuracy. This is in line with our conclusion in Section 4.2.1 that the missing nodes also contain some acoustic features, but don't have a significant impact on the overall detection.

4.2.3 The Effect of Different Acoustic Features

In this section, we use different topologies and different acoustic features as inputs to the NMF algorithm.

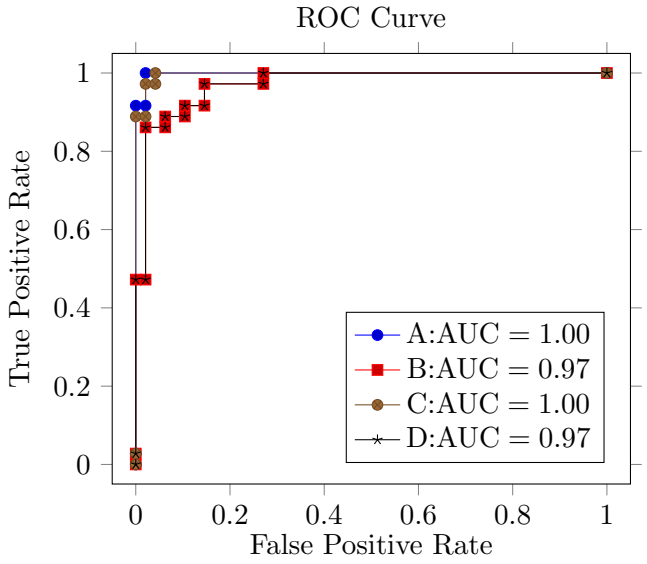


Figure 4.11: The ROC curve of different network structures with Isolation Forest. The parameter settings are shown in Table 4.6.

Table 4.7: Experiment settings with different features using Isolation Forest.

Experiments	Topology	Input Features
A	Fully-Connected	STFT Spectrogram
B	Fully-Connected	Mel filterbank features
C	Ring	STFT Spectrogram
D	Ring	Mel filterbank features

The aim is to investigate how the performance of the Isolation Forest model is affected by different audio preprocessing methods. The original input signal is sampled at 16 kHz without adding background noise, and all 12 microphone nodes in the home environment are considered. The results are shown in Figure 4.12. Unlike when using OC-SVM, where the AUC does not change with the input features. When using the Isolation Forest model, the classification performance using the Mel filterbank features is slightly degraded under both topologies. But all AUC are still more than 0.95, which means the performance is still good.

4.2.4 The Effect of different sample rate

To explore the effect of reducing the sample rate on the Isolation Forest model, we resample the input audio at 8 kHz and compare it to the original audio sample at 16 kHz. All experiments in this section are performed with fully connected topology and no background noise is added. All results are shown in Figure 4.13.

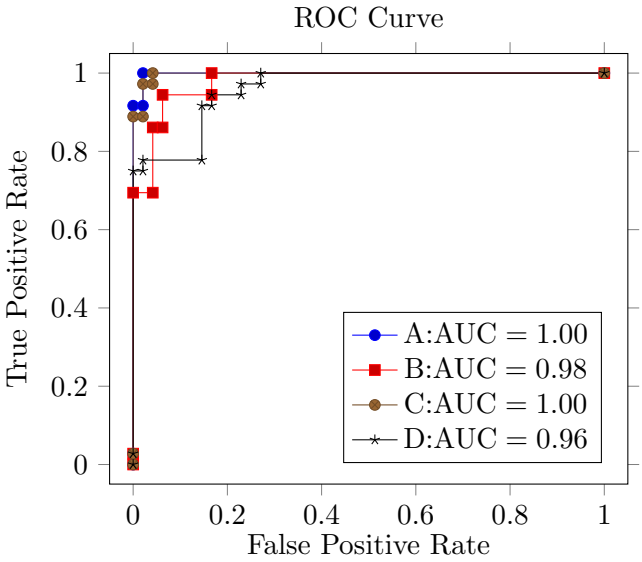


Figure 4.12: The ROC curve of different acoustic features with Isolation Forest. The parameter settings are shown in Table 4.7.

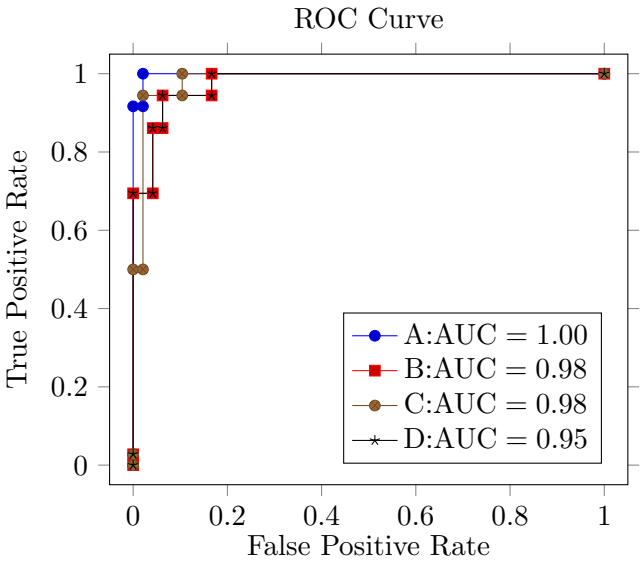


Figure 4.13: The ROC curve of different sample rate with Isolation Forest. The parameter settings are shown in Table 4.8.

Table 4.8: Experiment settings with different sample rate using Isolation Forest

Experiments	Input Features	Sample Rate
A	STFT Spectrogram	16kHz
B	Mel filterbank features	16kHz
C	STFT Spectrogram	8kHz
D	Mel filterbank features	8kHz

Table 4.9: Experiments with and without noise using Isolation Forest

Experiments	Input Features	Noise
A	STFT Spectrogram	Without Noise
B	Mel filterbank features	Without Noise
C	STFT Spectrogram	With Noise (SNR = 20dB)
D	Mel filterbank features	With Noise (SNR = 20dB)

It can be seen that halving the sample rate slightly reduces the performance of the Isolation Forest. Similar to the results obtained in Section 4.2.1, halving the sample rate causes less impact when using Mel fliterbank features.

4.2.5 The Effect of Additional Background Noise with Isolation Forest

Similarly, we also explore the effect of noise when using the Isolation Forest model. We mix noise with SNR of 20 dB and the original audio samples and compare the results using the original audio samples. All experiments in this section use the full-connected topology with 12 nodes, all audio samples are sampled at 16 kHz.

Figure 4.14 shows the results. From the results of Experiment A and Experiment C, it can be seen that the addition of background noise only decreases the AUC by 0.01 both when using STFT spectrogram or Mel filter bank features. While in Section 4.2.1.4, the additional background noise decreases the AUC of the OC-SVM model by more than 0.1. This indicates that Isolation Forest model is more insensitive to noise and more suitable for anomaly detection in noisy environments. In order to compare the sensitivity of Isolation Forest and OC-SVM to noise more intuitively. We compare the performance of both models under the same experimental parameters. The identical experimental parameters are, audio samples at 16 kHz, using 12 microphone nodes, using fully connected topology, using STFT spectrogram as acoustic features.

Figure 4.15 shows that the additional background noise decreases the AUC of OC-SVM from 1.00 to 0.87, while the AUC of Isolation Forest only reduces by 0.01. Therefore, it can be concluded that the additional background noise causes more degradation in the performance of OC-SVM under the same experimental parameters. In other words, Isolation Forest is more robust to

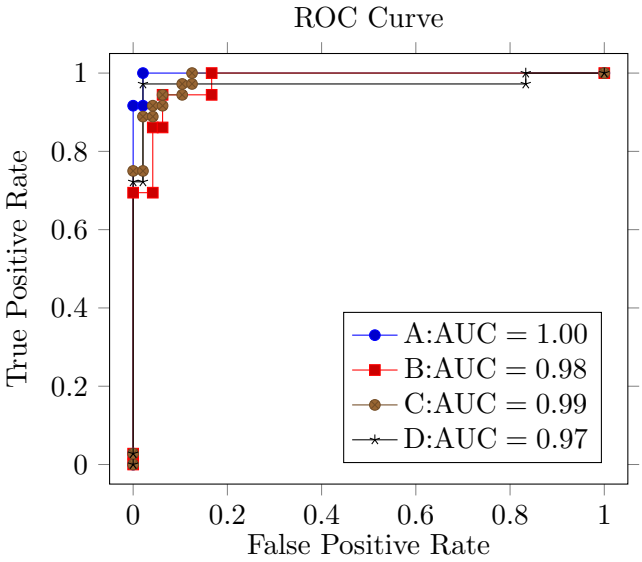


Figure 4.14: The ROC curve of experiments with and without additional noise using Isolation Forest. The parameter settings are shown in Table 4.9.

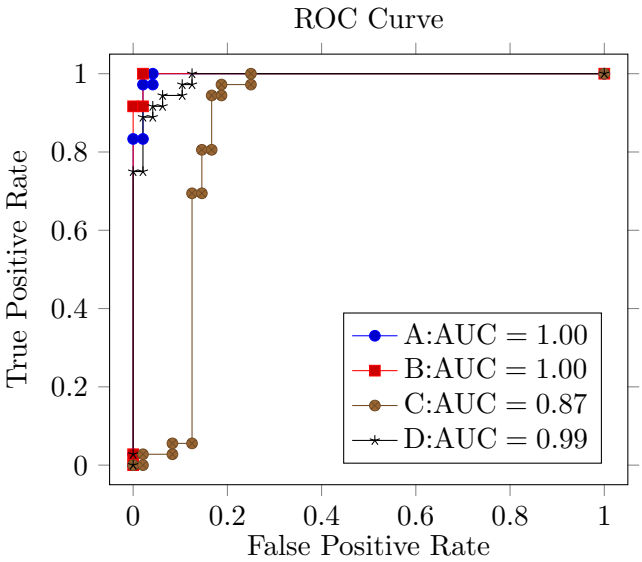


Figure 4.15: Comparison of the robustness to noise between OC-SVM and Isolation Forest. The parameter settings are shown in Table 4.10.

Table 4.10: Experiment settings for comparison of the robustness to noise between OC-SVM and Isolation Forest.

Experiments	Classification Model	Noise
A	OC-SVM	Without Noise
B	Isolation Forest	Without Noise
C	OC-SVM	With Noise (SNR = 20dB)
D	Isolation Forest	With Noise (SNR = 20dB)

noise.

4.3 Discussion of the Current Research

To evaluate the performance of the proposed NMF algorithm, we have performed 64 experiments with different parameters in total. The parameter settings are, audio samples at 16 kHz and 8 kHz, using 12 and 6 microphone nodes, using fully connected and ring topology, using STFT spectrogram and Mel filterbank features as acoustic features, using original audio samples or mixing them with additional background noise, and using OC-SVM and Isolation Forest as classification models. The complete experimental results are shown in the Appendix A.

From the analysis of the experimental results, we can draw the following conclusions. If we want to get better performance in anomaly sound detection, we should use higher sampling rate for audio samples and use Isolation Forest as the classification model because of its robustness to noise. The performance of the proposed algorithm is not affected by the changes in topology when all other settings are the same. Meanwhile, the performance of the classification model does not deteriorate severely even when the number of microphone nodes is halved, it still shows an excellent anomaly detection capability. This satisfies our requirement for flexibility, i.e. nodes in WASNs can (i) join and leave at any time (ii) communicate with only a subset of their neighbors. However, the more nodes in WASNs, the better performance, which indicates the superiority of the proposed ADMM-based NMF algorithm for unsupervised anomaly sound detection. Resampling the signal at a lower sample rate causes degradation in detection performance, especially when using OC-SVM and STFT. When adding background noise that differs significantly from the original audio samples, OC-SVM exhibits higher sensitivity to noise, while the Isolation Forest model is less influenced. Therefore, in the real world, the Isolation Forest model is more suitable for scenarios with high background noise.

In addition to the STFT spectrogram and Mel filterbank features, we also tried to use the Log-mel spectrum as the acoustic feature by logarithmic operation of the Mel filterbank features and fed it into the NMF algorithm. However, the results were not satisfactory, with the highest detection accuracy is about 75% and only about 55% when adding background noise and halving

the sampling rate, which is similar to the random guess. According to our speculation, the possible reason for this is that the same operation as for the Mel filterbank features needs to be performed in the initialization of the NMF algorithm.

Chapter 5

Conclusions and Further Work

5.1 Conclusions

Wireless acoustic sensor networks (WASNs) are widely used in smart cities, home automation, etc. Common applications include hearing aids, acoustical monitoring, and ambient intelligence. WASNs have simple and flexible network structures. Depending on the communication protocols, they have different topologies. Flexibility is often a key challenge in these topologies. Nodes should be able to join and leave the network at any time without affecting the performance of acoustic processing tasks. Meanwhile, they should be allowed to communicate with only a subset of their neighbors. Therefore, we propose two novel Non-Negative Matrix Factorization (NMF) algorithms to solve such distributed problems. The proposed algorithms do not depend on the changes of network topology and the number of nodes within the topology, which meets the demand for flexibility. Besides, we apply the proposed NMF algorithm to anomaly sound detection, design an anomaly sound detection system and evaluate its performance by conducting experiments with different parameters. The content of this research is as follows.

First, considering the flexibility of topologies in WASNs, we apply the consensus Alternating Direction Method of Multipliers (ADMM) to NMF. By splitting variables into local and global variants, alternating local optimization and global optimization, we can synchronize locally solved problems in a global context. The first algorithm we propose is NMF using consensus ADMM for fully connected topology, which requires that each node in the topology must communicate with all other nodes. Based on this algorithm, we propose the NMF using consensus ADMM for all topologies, which has higher flexibility and allows the node only communicate with a subset of nodes. Experiments demonstrate that our proposed algorithms have a fast convergence so that they can be used to solve large-scale distributed problems.

Then we design an anomaly sound detection system combining the NMF algorithm with two classification models, One-Class support machines and Isolation Forest. The NMF represents the

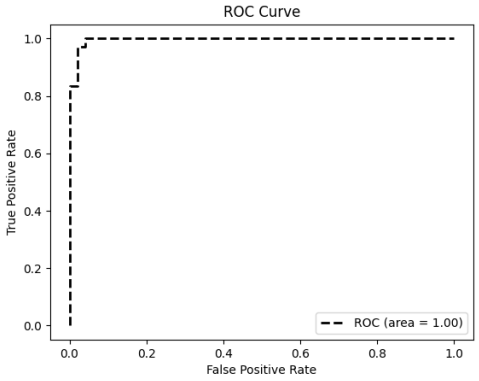
acoustic features with a codebook, which is fed into the model for training and prediction. The detection occurs in every node, after integrating the results we get an anomaly score for each audio to determine the abnormality of sound.
Finally, by conducting multiple experiments with different parameters, we evaluate the performance of our anomaly detection system and seek the best practices. The experiment results show that if we want to obtain better performance, we need to sample the original audio at the highest possible sampling rate and use the Isolation Forest as a detection model because of its robustness to noise. In addition, increasing the number of nodes within the topology can improve the performance of the system. Through these methods, we can better utilize the proposed ADMM-based NMF algorithms.

5.2 Further Work

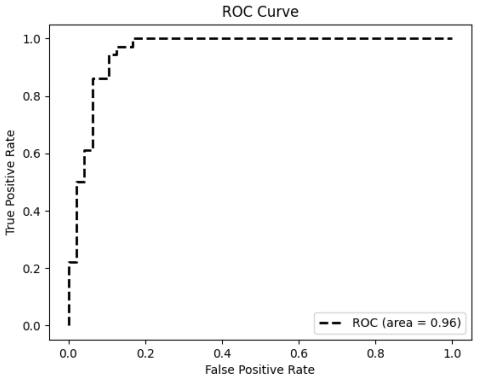
There are various directions for future work on this research. For example, we now focus only on anomaly sound detection in the home environment. By using different datasets, we can find new application scenarios, such as detecting anomaly sounds of machines in factories. In our research we use two acoustic feature extraction methods, Short-time Fourier transform and Mel filter bank. We also try to use Log-Mel spectrum as features but the result is not satisfying. In the future we can adopt other acoustic features such as Log-Mel spectrum, Mel-frequency cepstral coefficients, and revise our NMF algorithm to make it applicable to the case where other acoustic features are used as input. Another point can be the complexity of the network. In our experiments, only the fully connected topology and the ring topology where nodes communicate only with their left and right neighbors are considered, whose structures are relatively simple. In future research, we can use more complex topologies to evaluate the performance of the anomaly sound detection system. Another point is to compare the performance of our algorithm with other algorithms, such as auto-encoder, which is popular in anomaly sound detection. We can do some research on it and seek the possibility of using neural network based algorithms in our detection system. Moreover, we should also notice the difference between real-world estimation and dataset, and investigate the ways how our detection system can be applied in the real life.

Appendix A

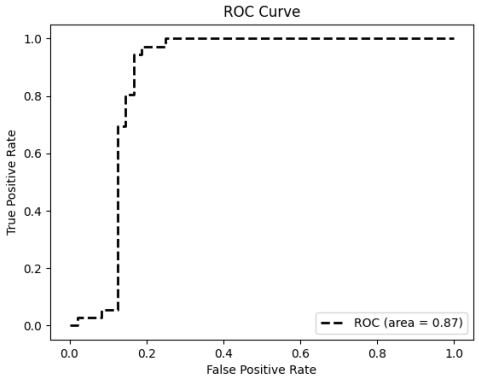
Experiment Results



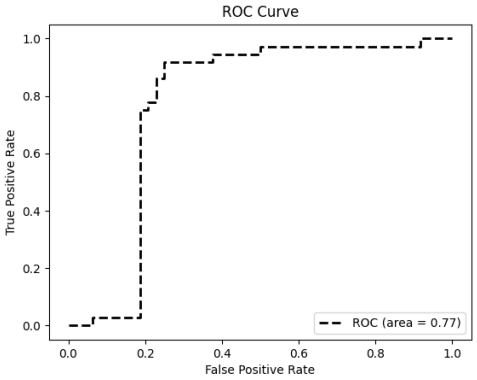
(a) Result 1



(b) Result 2

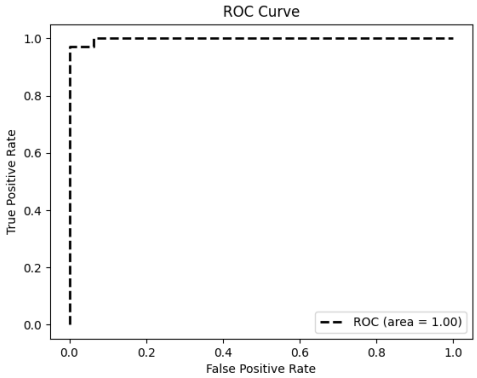


(c) Result 3

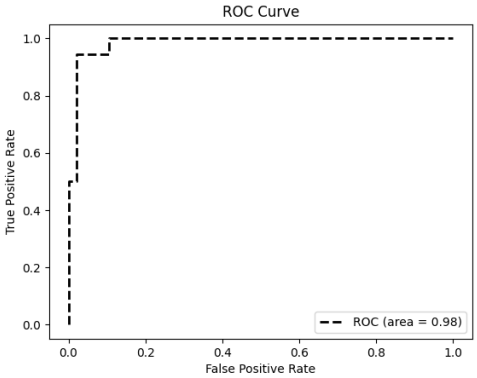


(d) Result 4

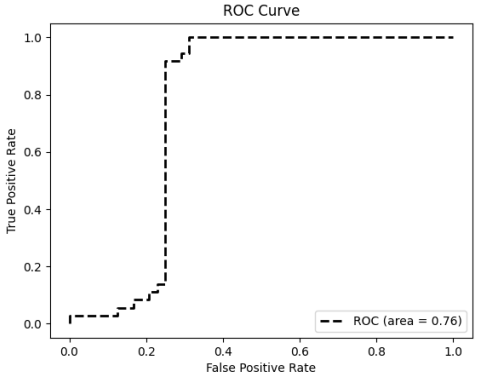
Figure A.1: Result 1-4



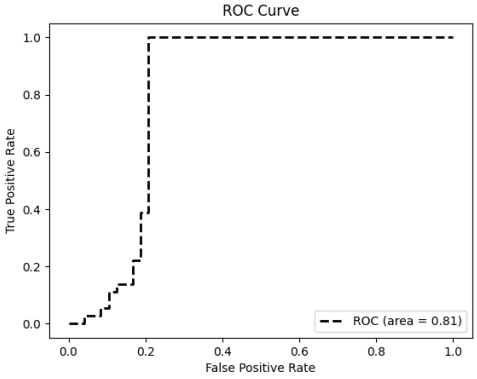
(a) Result 5



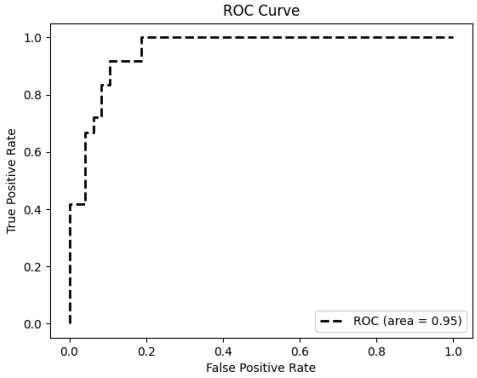
(b) Result 6



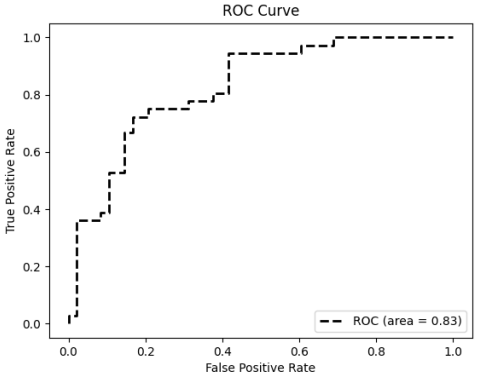
(c) Result 7



(d) Result 8

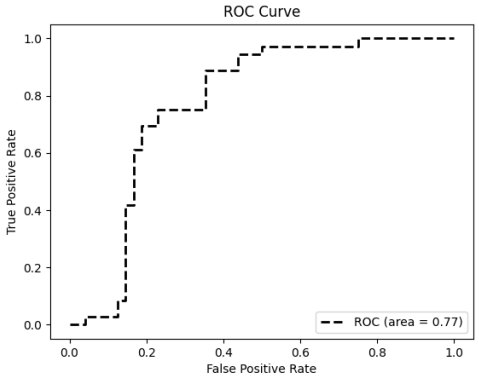


(e) Result 9

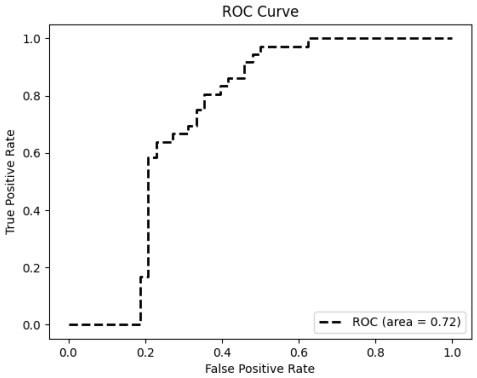


(f) Result 10

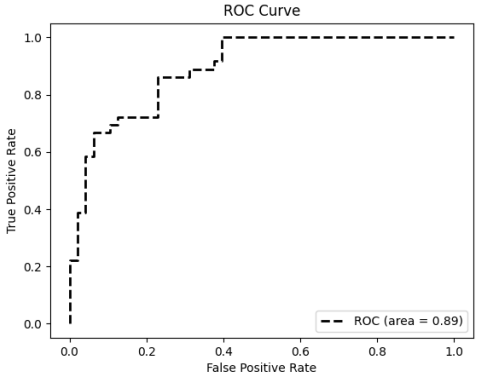
Figure A.2: Result 6-10



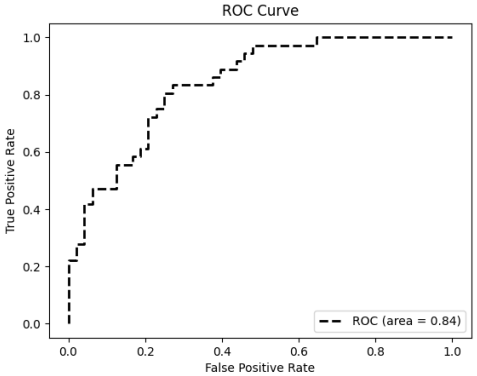
(a) Result 11



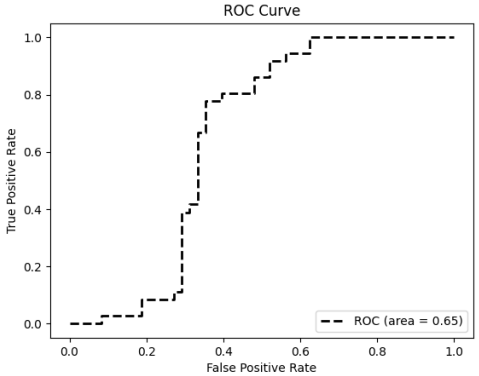
(b) Result 12



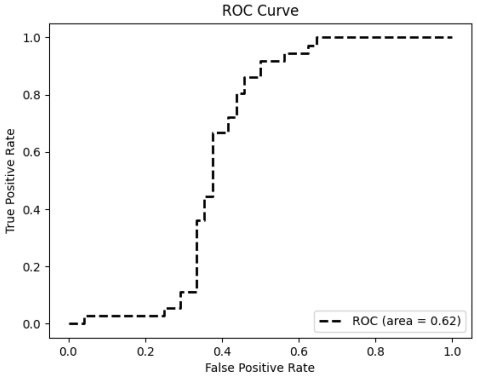
(c) Result 13



(d) Result 14

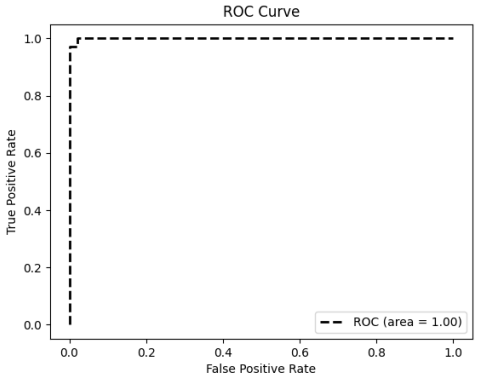


(e) Result 15

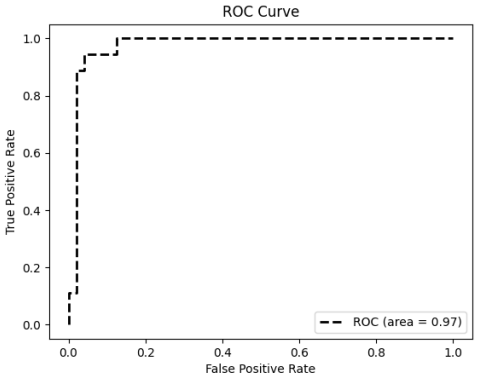


(f) Result 16

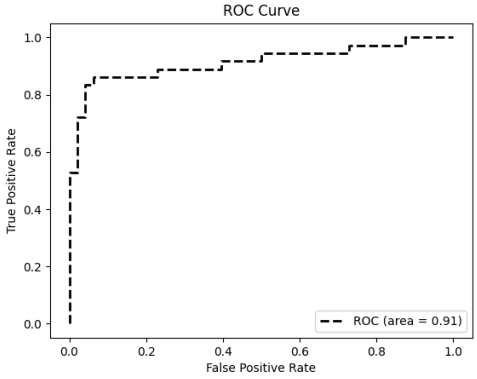
Figure A.3: Result 11-16



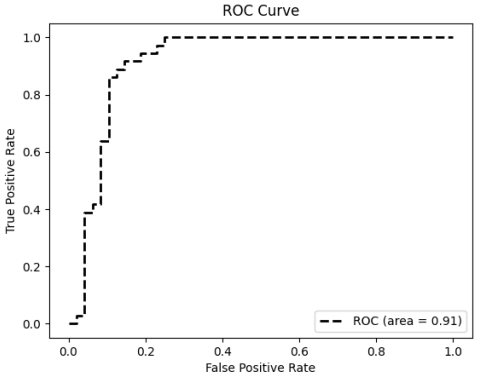
(a) Result 17



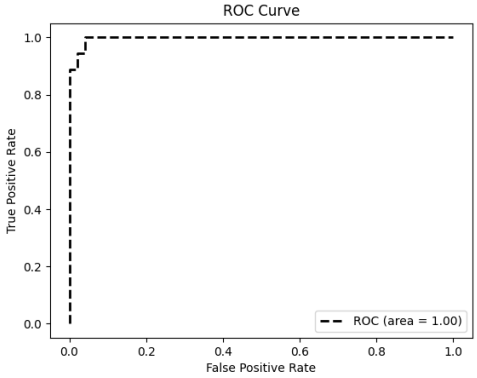
(b) Result 18



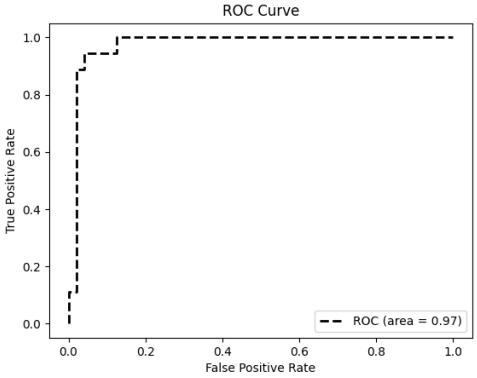
(c) Result 19



(d) Result 20

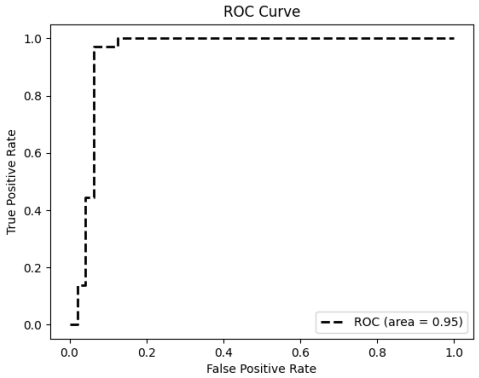


(e) Result 21

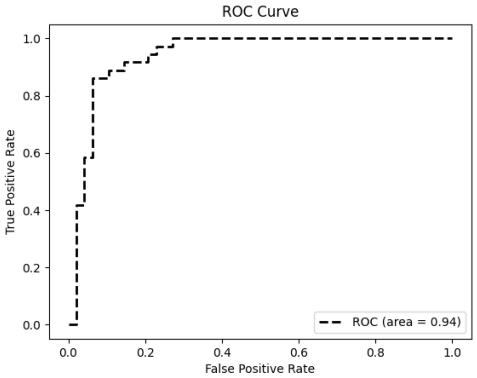


(f) Result 22

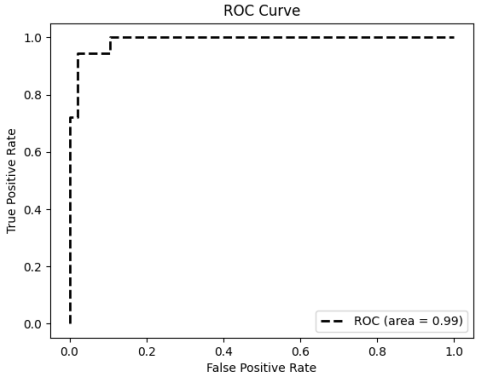
Figure A.4: Result 16-22



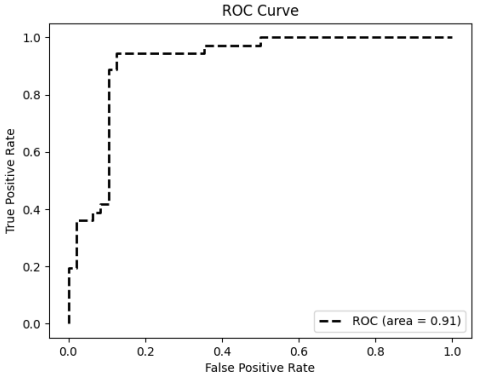
(a) Result 23



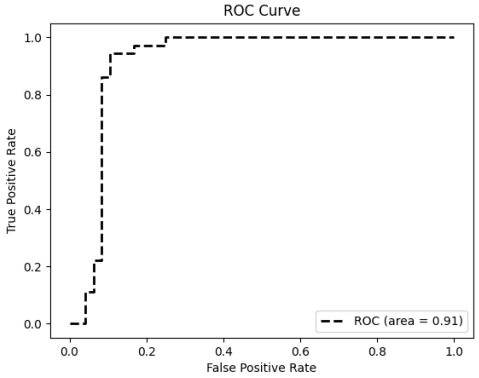
(b) Result 24



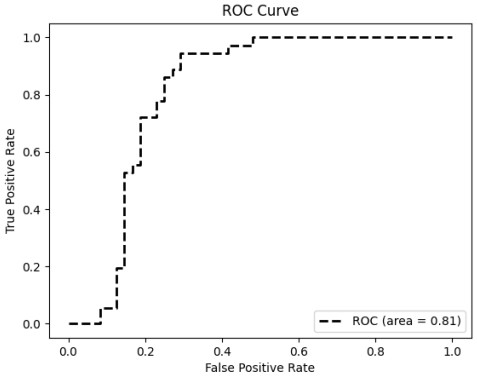
(c) Result 25



(d) Result 26

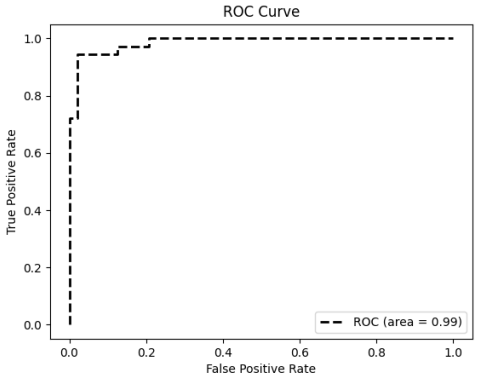


(e) Result 27

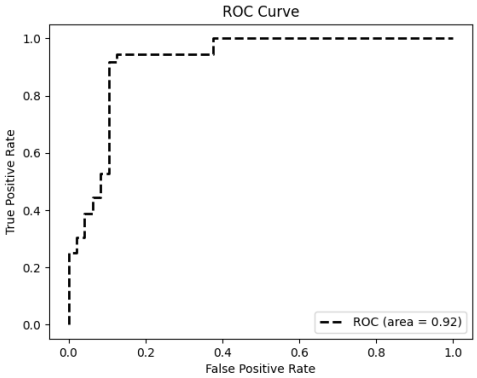


(f) Result 28

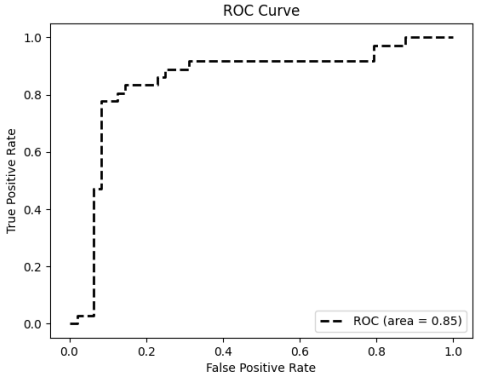
Figure A.5: Result 23-28



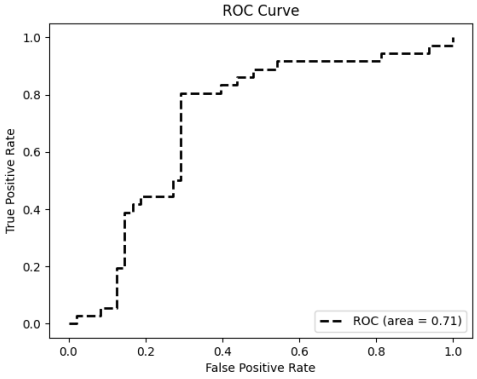
(a) Result 29



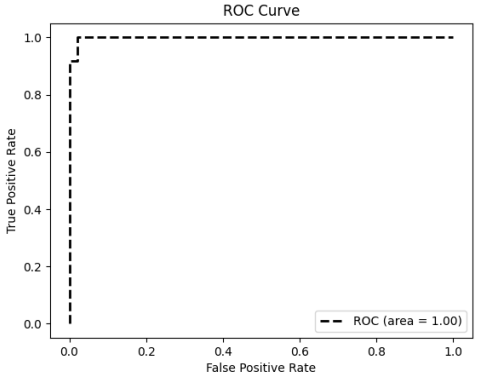
(b) Result 30



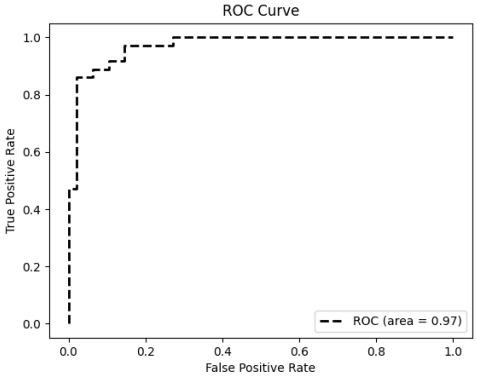
(c) Result 31



(d) Result 32

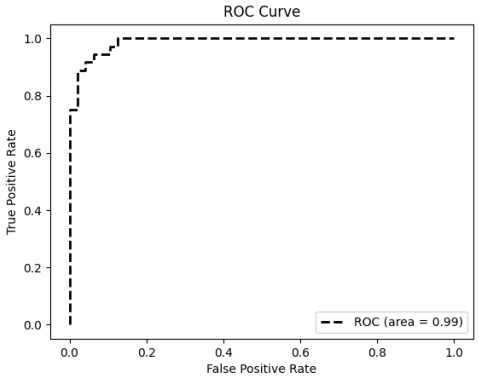


(e) Result 33

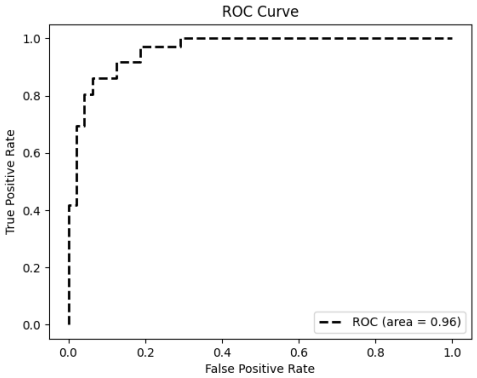


(f) Result 34

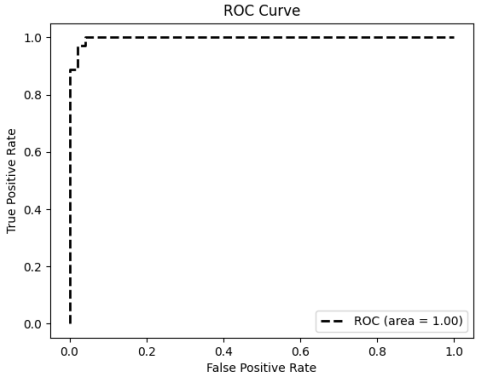
Figure A.6: Result 29-34



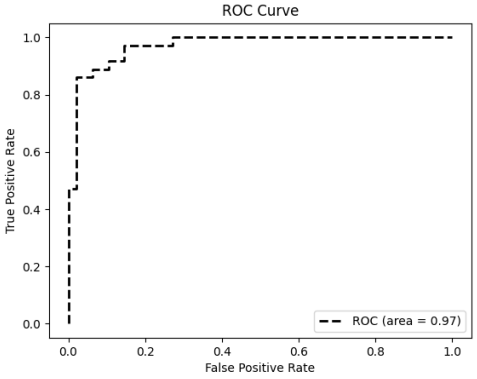
(a) Result 35



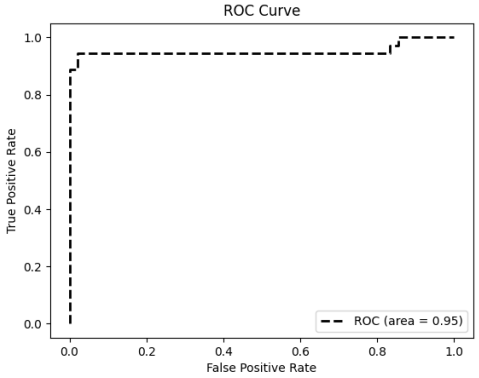
(b) Result 36



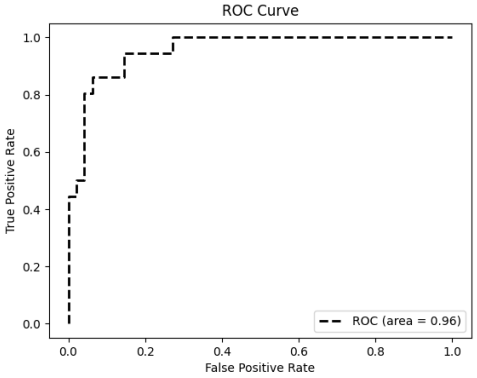
(c) Result 37



(d) Result 38

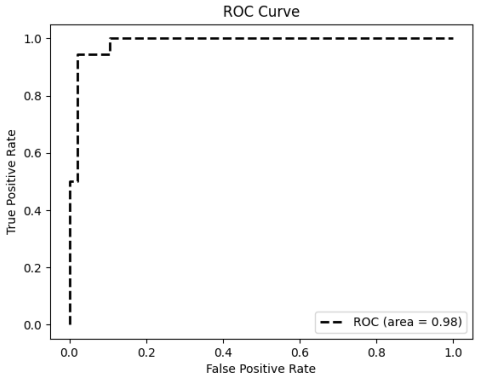


(e) Result 39

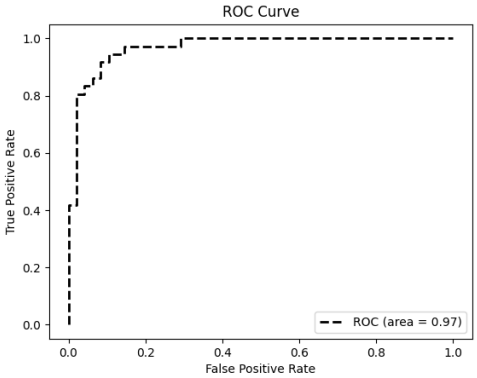


(f) Result 40

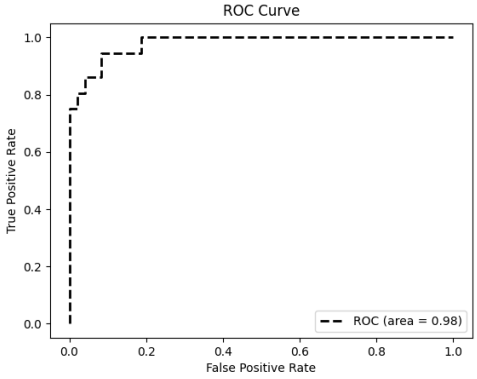
Figure A.7: Result 35-40



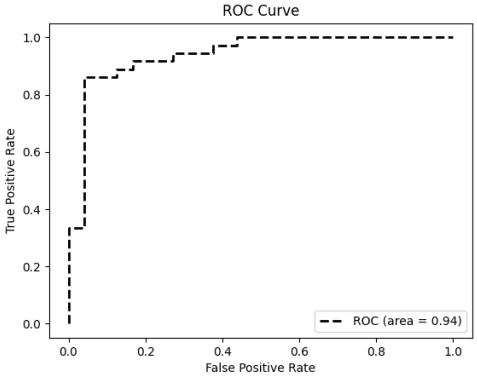
(a) Result 41



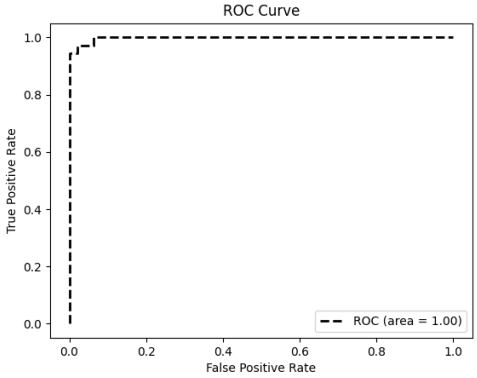
(b) Result 42



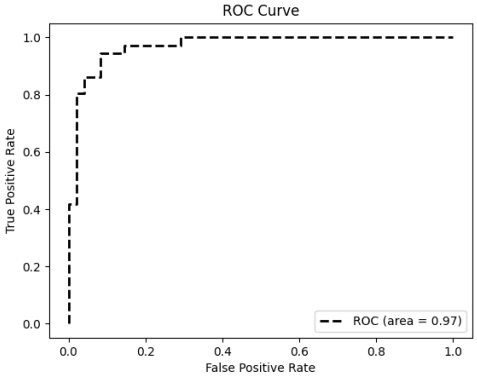
(c) Result 43



(d) Result 44

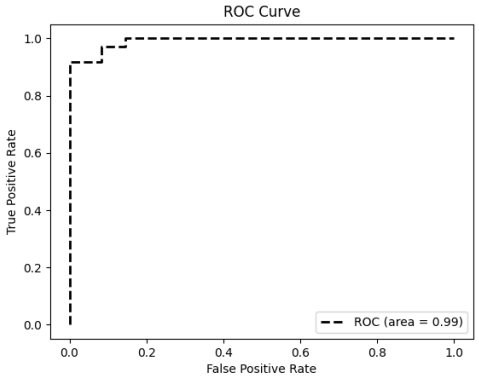


(e) Result 45

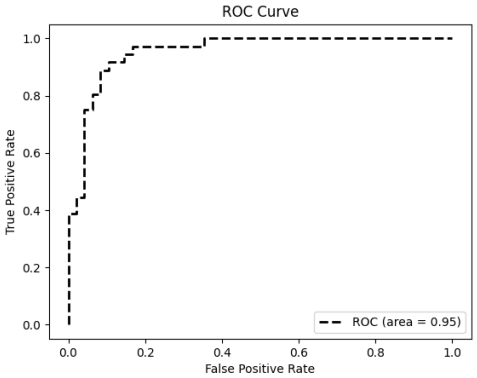


(f) Result 46

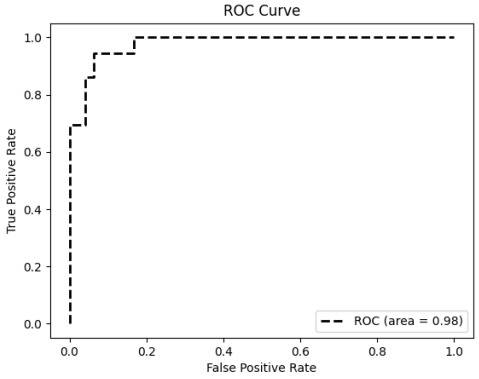
Figure A.8: Result 41-46



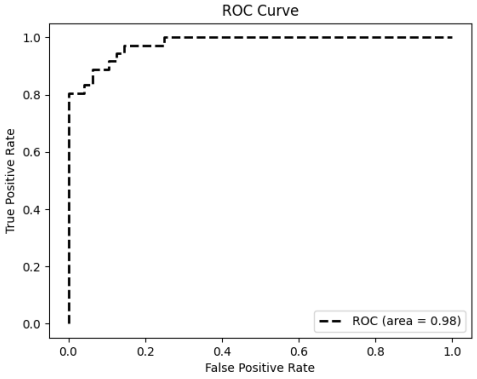
(a) Result 47



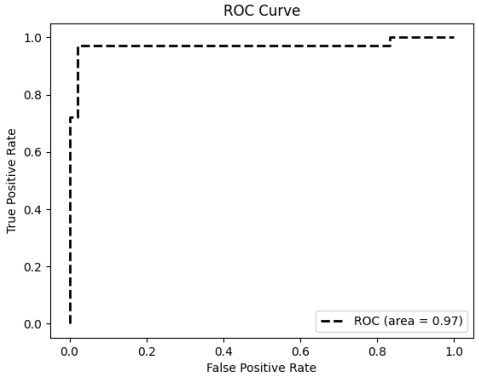
(b) Result 48



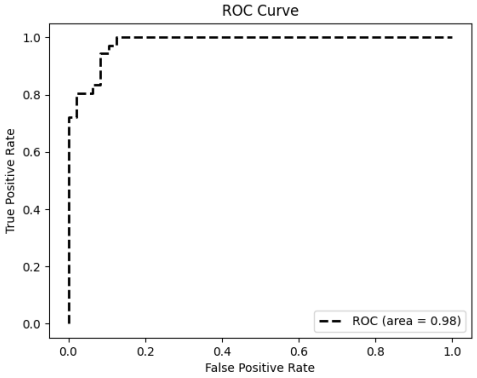
(c) Result 49



(d) Result 50

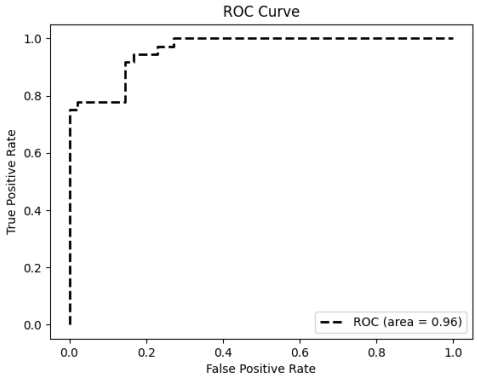


(e) Result 51

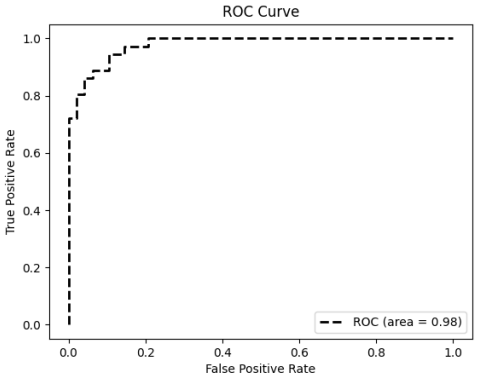


(f) Result 52

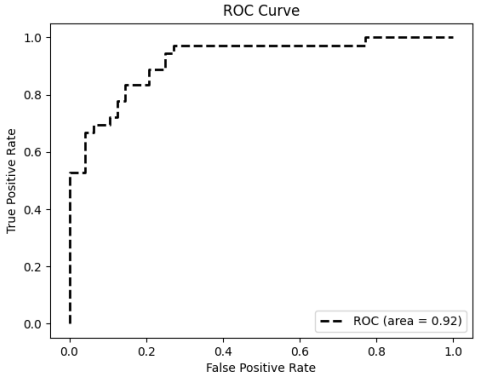
Figure A.9: Result 47-52



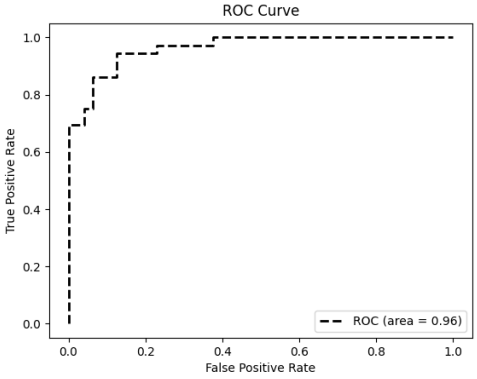
(a) Result 53



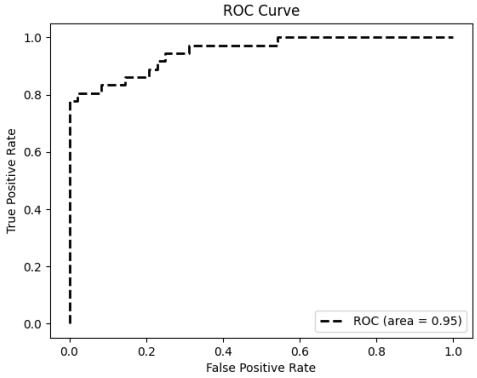
(b) Result 54



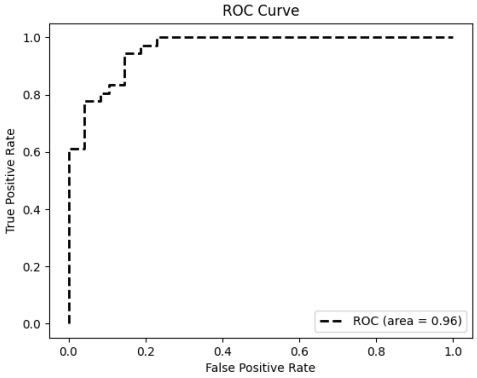
(c) Result 55



(d) Result 56

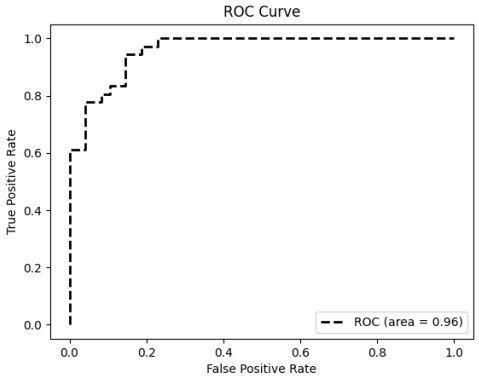


(e) Result 57

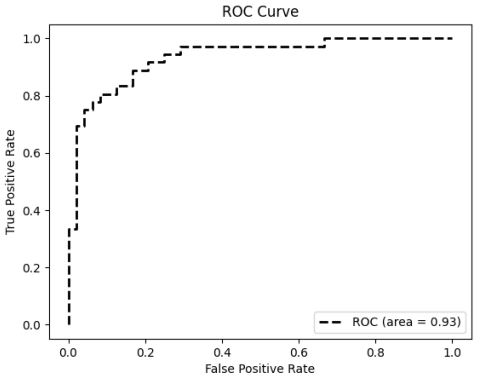


(f) Result 58

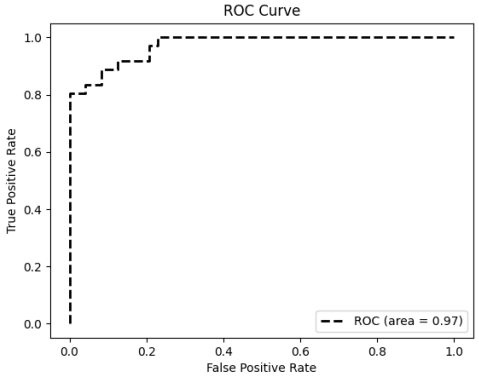
Figure A.10: Result 53-58



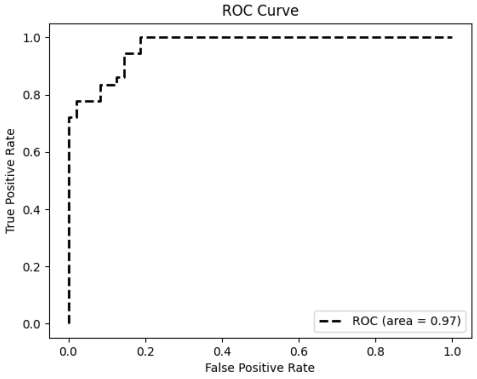
(a) Result 59



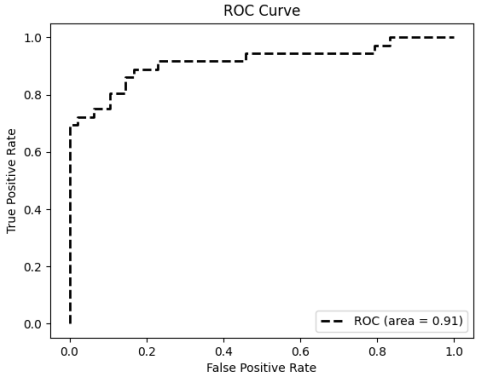
(b) Result 60



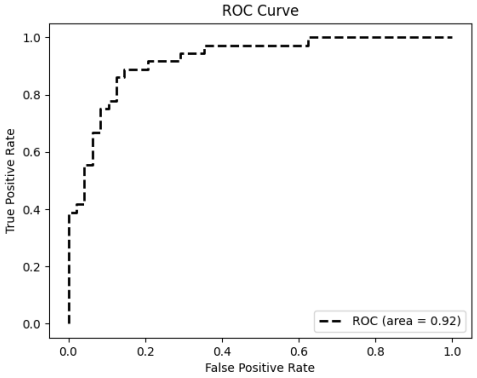
(c) Result 61



(d) Result 62



(e) Result 63



(f) Result 64

Figure A.11: Result 59-64

Table A.1: Experiment Results.

Experiments	Classification Model	Acoustic Features	Topology	Noise	Number of Nodes	Sampling Rate	AUC
1	OC-SVM	STFT Spectrogram	Fully Connected	Without Noise	12	16 kHz	1.00
2	OC-SVM	STFT Spectrogram	Fully Connected	Without Noise	6	16 kHz	0.96
3	OC-SVM	STFT Spectrogram	Fully Connected	SNR = 20 dB	12	16 kHz	0.87
4	OC-SVM	STFT Spectrogram	Fully Connected	SNR = 20 dB	6	16 kHz	0.77
5	OC-SVM	STFT Spectrogram	Ring	Without Noise	12	16 kHz	1.00
6	OC-SVM	STFT Spectrogram	Ring	Without Noise	6	16 kHz	0.98
7	OC-SVM	STFT Spectrogram	Ring	SNR = 20 dB	12	16 kHz	0.76
8	OC-SVM	STFT Spectrogram	Ring	SNR = 20 dB	6	16 kHz	0.81
9	OC-SVM	STFT Spectrogram	Fully Connected	Without Noise	12	8 kHz	0.95
10	OC-SVM	STFT Spectrogram	Fully Connected	Without Noise	6	8 kHz	0.83
11	OC-SVM	STFT Spectrogram	Fully Connected	SNR = 20	12	8 kHz	0.77
12	OC-SVM	STFT Spectrogram	Fully Connected	SNR = 20	6	8 kHz	0.72
13	OC-SVM	STFT Spectrogram	Ring	Without Noise	12	8 kHz	0.89
14	OC-SVM	STFT Spectrogram	Ring	Without Noise	6	8 kHz	0.84
15	OC-SVM	STFT Spectrogram	Ring	SNR = 20	12	8 kHz	0.65
16	OC-SVM	STFT Spectrogram	Ring	SNR = 20	6	8 kHz	0.62
17	OC-SVM	Mel Filterbank	Fully Connected	Without Noise	12	16 kHz	1.00
18	OC-SVM	Mel Filterbank	Fully Connected	Without Noise	6	16 kHz	0.97
19	OC-SVM	Mel Filterbank	Fully Connected	SNR = 20	12	16 kHz	0.91
20	OC-SVM	Mel Filterbank	Fully Connected	SNR = 20	6	16 kHz	0.91
21	OC-SVM	Mel Filterbank	Ring	Without Noise	12	16 kHz	1.00
22	OC-SVM	Mel Filterbank	Ring	Without Noise	6	16 kHz	0.97
23	OC-SVM	Mel Filterbank	Ring	SNR = 20	12	16 kHz	0.95
24	OC-SVM	Mel Filterbank	Ring	SNR = 20	6	16 kHz	0.94
25	OC-SVM	Mel Filterbank	Ring	Without Noise	12	8 kHz	0.99
26	OC-SVM	Mel Filterbank	Ring	Without Noise	6	8 kHz	0.91
27	OC-SVM	Mel Filterbank	Ring	SNR = 20	12	8 kHz	0.91
28	OC-SVM	Mel Filterbank	Ring	SNR = 20	6	8 kHz	0.81
29	OC-SVM	Mel Filterbank	Fully Connected	Without Noise	12	8 kHz	0.99
30	OC-SVM	Mel Filterbank	Fully Connected	Without Noise	6	8 kHz	0.92
31	OC-SVM	Mel Filterbank	Fully Connected	SNR = 20	12	8 kHz	0.85
32	OC-SVM	Mel Filterbank	Fully Connected	SNR = 20	6	8 kHz	0.71
33	Isolation Forest	STFT Spectrogram	Fully Connected	Without Noise	12	16 kHz	1.00
34	Isolation Forest	STFT Spectrogram	Fully Connected	Without Noise	6	16 kHz	0.97
35	Isolation Forest	STFT Spectrogram	Fully Connected	SNR = 20 dB	12	16 kHz	0.99
36	Isolation Forest	STFT Spectrogram	Fully Connected	SNR = 20 dB	6	16 kHz	0.96
37	Isolation Forest	STFT Spectrogram	Ring	Without Noise	12	16 kHz	1.00
38	Isolation Forest	STFT Spectrogram	Ring	Without Noise	6	16 kHz	0.97
39	Isolation Forest	STFT Spectrogram	Ring	SNR = 20 dB	12	16 kHz	0.95
40	Isolation Forest	STFT Spectrogram	Ring	SNR = 20 dB	6	16 kHz	0.96
41	Isolation Forest	STFT Spectrogram	Fully Connected	Without Noise	12	8 kHz	0.98
42	Isolation Forest	STFT Spectrogram	Fully Connected	Without Noise	6	8 kHz	0.97
43	Isolation Forest	STFT Spectrogram	Fully Connected	SNR = 20 dB	12	8 kHz	0.98
44	Isolation Forest	STFT Spectrogram	Fully Connected	SNR = 20 dB	6	8 kHz	0.94
45	Isolation Forest	STFT Spectrogram	Ring	Without Noise	12	8 kHz	1.00
46	Isolation Forest	STFT Spectrogram	Ring	Without Noise	6	8 kHz	0.97
47	Isolation Forest	STFT Spectrogram	Ring	SNR = 20 dB	12	8 kHz	0.99
48	Isolation Forest	STFT Spectrogram	Ring	SNR = 20 dB	6	8 kHz	0.95
49	Isolation Forest	Mel Filterbank	Fully Connected	Without Noise	12	16 kHz	0.98
50	Isolation Forest	Mel Filterbank	Fully Connected	Without Noise	6	16 kHz	0.98
51	Isolation Forest	Mel Filterbank	Fully Connected	SNR = 20 dB	12	16 kHz	0.97
52	Isolation Forest	Mel Filterbank	Fully Connected	SNR = 20 dB	6	16 kHz	0.98
53	Isolation Forest	Mel Filterbank	Ring	Without Noise	12	16 kHz	0.96
54	Isolation Forest	Mel Filterbank	Ring	Without Noise	6	16 kHz	0.98
55	Isolation Forest	Mel Filterbank	Ring	SNR = 20 dB	12	16 kHz	0.92
56	Isolation Forest	Mel Filterbank	Ring	SNR = 20 dB	6	16 kHz	0.96
57	Isolation Forest	Mel Filterbank	Fully Connected	Without Noise	12	8 kHz	0.95
58	Isolation Forest	Mel Filterbank	Fully Connected	Without Noise	6	8 kHz	0.96
59	Isolation Forest	Mel Filterbank	Fully Connected	SNR = 20 dB	12	8 kHz	0.96
60	Isolation Forest	Mel Filterbank	Fully Connected	SNR = 20 dB	6	8 kHz	0.93
61	Isolation Forest	Mel Filterbank	Ring	Without Noise	12	8 kHz	0.97
62	Isolation Forest	Mel Filterbank	Ring	Without Noise	6	8 kHz	0.97
63	Isolation Forest	Mel Filterbank	Ring	SNR = 20 dB	12	8 kHz	0.91
64	Isolation Forest	Mel Filterbank	Ring	SNR = 20 dB	6	8 kHz	0.92

Bibliography

[1] *Classification: Roc curve and auc.* https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc?hl=en, Retrieved 12.05.2009.

[2] *The intuition understanding of nmf.* https://www.geeksforgeeks.org/non-negative-matrix-factorization/, Retrieved 12.05.2009.

[3] *Mel-frequency cepstral coefficients (mfccs).* https://wiki.aalto.fi/display/ITSP/Cepstrum+and+MFCC.

[4] *Neural networks and mlp.* https://www.dotnetlovers.com/article/243/neural-networks-and-mlp.

[5] H. Abdi and L. J. Williams, *Principal component analysis*, Wiley interdisciplinary reviews: computational statistics, 2 (2010), pp. 433–459.

[6] C. Ai, X. Sun, H. Zhao, R. Ma, and X. Dong, *Pipeline damage and leak sound recognition based on hmm*, in 2008 7th World Congress on Intelligent Control and Automation, IEEE, 2008, pp. 1940–1944.

[7] K. Baker, *Singular value decomposition tutorial*, The Ohio State University, 24 (2005).

[8] A. Bertrand, *Applications and trends in wireless acoustic sensor networks: A signal processing perspective*, in 2011 18th IEEE symposium on communications and vehicular technology in the Benelux (SCVT), IEEE, 2011, pp. 1–6.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Foundations and Trends® in Machine learning, 3 (2011), pp. 1–122.

[10] C. J. Burges, *A tutorial on support vector machines for pattern recognition*, Data mining and knowledge discovery, 2 (1998), pp. 121–167.

[11] E. Cakir and T. Virtanen, *Convolutional recurrent neural networks for rare sound event detection*, Deep Neural Networks for Sound Event Detection, 12 (2019).

[12] J. Cheng, B. Xie, C. Lin, and L. Ji, *A comparative study in birds: call-type-independent species and individual recognition using four machine-learning methods and two acoustic features*, Bioacoustics, 21 (2012), pp. 157–171.

[13] G. Dekkers, S. Lauwereins, B. Thoen, M. W. Adhana, H. Brouckxon, B. Van den Bergh, T. Van Waterschoot, B. Vanrumste, M. Verhelst, and P. Karsmakers, *The sins database for detection of daily activities in a home environment using an acoustic sensor network*, Detection and Classification of Acoustic Scenes and Events 2017, (2017), pp. 1–5.

[14] N. Gillis, *The why and how of nonnegative matrix factorization*, Connections, 12 (2014).

[15] D. Hu, F. Nie, and X. Li, *Deep multimodal clustering for unsupervised audiovisual learning*, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 9248–9257.

[16] X. Huang, A. Acero, H.-W. Hon, and R. Reddy, *Spoken language processing: A guide to theory, algorithm, and system development*, Prentice hall PTR, 2001.

[17] D.-S. Kim, S.-Y. Lee, and R. M. Kil, *Auditory processing of speech signals for robust speech recognition in real-world noisy environments*, IEEE Transactions on speech and audio processing, 7 (1999), pp. 55–69.

[18] D. Lee and H. S. Seung, *Algorithms for non-negative matrix factorization*, Advances in neural information processing systems, 13 (2000).

[19] F. T. Liu, K. M. Ting, and Z.-H. Zhou, *Isolation forest*, in 2008 eighth ieee international conference on data mining, IEEE, 2008, pp. 413–422.

[20] J. Ma and S. Perkins, *Time-series novelty detection using one-class support vector machines*, in Proceedings of the International Joint Conference on Neural Networks, 2003., vol. 3, IEEE, 2003, pp. 1741–1745.

[21] P. Malhotra, L. Vig, G. Shroff, P. Agarwal, et al., *Long short term memory networks for anomaly detection in time series*, in Proceedings, vol. 89, 2015, pp. 89–94.

[22] A. Ozerov, C. Févotte, and E. Vincent, *An introduction to multichannel nmf for audio source separation*, in Audio Source Separation, Springer, 2018, pp. 73–94.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine learning in Python*, Journal of Machine Learning Research, 12 (2011), pp. 2825–2830.

[24] D. A. Reynolds, *Gaussian mixture models.*, Encyclopedia of biometrics, 741 (2009).

[25] M. Sakurada and T. Yairi, *Anomaly detection using autoencoders with nonlinear dimensionality reduction*, in Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis, 2014, pp. 4–11.

[26] B. Schölkopf, R. C. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, *Support vector method for novelty detection*, Advances in neural information processing systems, 12 (1999).

[27] J. V. Stone, *Independent component analysis: a tutorial introduction*, (2004).

[28] T. H. Summers and J. Lygeros, *Distributed model predictive consensus via the alternating direction method of multipliers*, in 2012 50th annual Allerton conference on communication, control, and computing (Allerton), IEEE, 2012, pp. 79–84.

[29] D. L. Sun and C. Fevotte, *Alternating direction method of multipliers for non-negative matrix factorization with the beta-divergence*, in 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE, 2014, pp. 6201–6205.

[30] J. Thiemann, N. Ito, and E. Vincent, *Demand: a collection of multi-channel recordings of acoustic noise in diverse environments*, in Proc. Meetings Acoust, 2013, pp. 1–6.

[31] I. W. Tsang, J. T. Kwok, P.-M. Cheung, and N. Cristianini, *Core vector machines: Fast svm training on very large data sets.*, Journal of Machine Learning Research, 6 (2005).

[32] V. N. Vapnik, *An overview of statistical learning theory*, IEEE transactions on neural networks, 10 (1999), pp. 988–999.

[33] S. A. Vavasis, *On the complexity of nonnegative matrix factorization*, SIAM Journal on Optimization, 20 (2010), pp. 1364–1377.