Hochschule Offenburg
offenburg.university

MASTER THESIS

Machine Learning Based Microphone Fingerprinting

Submitted by:

Victor Azzam

Enterprise and IT Security (ENITS)

Media Department

1st Supervisor – Prof. Dr. phil. M.Sc. Andreas Schaad

2nd Supervisor – Prof. Dr.-Ing. Janis Keuper

Project code repository

https://github.com/victorazzam/mic

Licensed under (GPLv2)

April 13th, 2023

ABSTRACT

Much of the research in the field of audio-based machine learning has focused on recreating human speech via feature extraction and imitation, known as deepfakes.[1][2] The current state of affairs has prompted a look into other areas, such as the recognition of recording devices, and potentially speakers, by only analysing sound files. Segregation and feature extraction are at the core of this approach.

This research focuses on determining whether a recorded sound can reveal the recording device with which it was captured. Each specific microphone manufacturer and model, among other characteristics and imperfections, can have subtle but compounding effects on the results, whether it be differences in noise, or the recording tempo and sensitivity of the microphone while recording. By studying these slight perturbations, it was found to be possible to distinguish between microphones based on the sounds they recorded.[3]

After the recording, pre-processing, and feature extraction phases we completed, the prepared data was fed into several different machine learning algorithms, with results ranging from 70% to 100% accuracy, showing Multi-Layer Perceptron and Logistic Regression to be the most effective for this type of task.

This was further extended to be able to tell the difference between two microphones of the same make and model. Achieving the identification of identical models of a microphone suggests that the small deviations in their manufacturing process are enough of a factor to uniquely distinguish them and potentially target individuals using them. This however does not take into account any form of compression applied to the sound files, as that may alter or degrade some or most of the distinguishing features that are necessary for this experiment.

Building on top of prior research in the area, such as by Das et al. in [4] in which different acoustic features were explored and assessed on their ability to be used to uniquely fingerprint smartphones, more concrete results along with the methodology by which they were achieved are published in this project's publicly accessible code repository.[5]

CCS Concepts

Computing methodologies → Machine learning → Machine learning algorithms → Feature selection

Additional keywords: Machine learning, microphone, sound, audio, fingerprinting, forensics, classification.

Citation Format

Victor Azzam, 2023. Machine Learning Based Microphone Fingerprinting. Master Thesis. Offenburg University of Applied Sciences, GitHub code repository: https://github.com/victorazzam/mic

TABLE OF CONTENTS

TABLE OF FIGURES

GLOSSARY

AAC	An audio format by Bell Labs and others
AI	Artificial Intelligence
AUC-ROC	Area Under the Curve of the Receiver Operating Characteristic
CNN	Convolutional Neural Network
DNN	Deep Neural Network
HNR	Harmonic-to-Noise Ratio
JSON	A standard text-based format for representing structured data
MFCC	Mel-frequency cepstral coefficient
ML	Machine Learning
MLP	Multi-Layer Perceptron
MP3	An audio format by the Fraunhofer Society
NMF	Negative Matrix Factorisation
OGG	An audio format by the Xiph.Org Foundation
PCA	Principal Component Analysis
RF	Random Forest
RNN	Recurrent Neural Network
SciPy	Science Python
SD card	Secure Digital memory card
SVM	Support Vector Machine
WAV	An audio format by IBM and Microsoft
ZCR	Zero-crossing rate

1 INTRODUCTION

Machine learning has rapidly grown in popularity in scientific research communities and in computer science to either optimise existing research methodologies or to create novel ones. This paper focuses on existing but scarce research in microphone attribution, and is primarily based on manually recorded audio training data that is fed into several ML classifiers for supervised learning. The resulting models attempt to determine the source of each distinct piece of audio, or more specifically the device via which a particular sound segment was captured.

1.1 Motivation

One way in which this experiment could be helpful is to be able to tell, for example, whether a particular audio segment from a news report was recorded in a studio in conditions other than those suggested by the news clip. Among many other domain-specific uses, it would also be possible to tell if a highly influential public figure indeed said what was suggested by people in a viral video, or if it was a deepfake and not recorded with the microphone as shown in the clip.

1.2 Problem Statements

Although the gist of the research lies upon the question of audio attribution, this goal could be further divided into research questions, some of which include:

- Is it possible to identify a microphone based on its audio recordings?
- How to identify discrepancies in sound captured by two recording devices?
- What could limit or hinder the ability to distinguish between them?
- How does the presence of human speech, noise, or other types of sound affect the process?
- What minimum length of audio recording is required for successful training?

Some of the questions that can arise during the course of the experiment that can aid in further research include:

- Can existing methods, such as those implemented by other researchers and companies in the past, be adapted to this experiment?
- Do the methods outlined and tested in this research lend themselves to being exploited by adversaries?
    - To follow up, which mitigation strategies can be used to circumvent detection?
    - Can this be replicated to mimic other recording devices?

To ensure that the results are more definitive, a choreographed series of steps must be followed, starting with the data collection phase, pre-processing and feature extraction, machine learning analysis, evaluation, and ending with some results and conclusions. This completes the pipeline of this project, where any of the steps can be implemented as a separate building block and automated in some way.

1.3 Technologies used

This part describes the outline of a plan for carrying out the experiment. Starting with 5 different microphones and an audio interface that can capture sound from all of them at the same time, the first step is to ensure that everything is recorded in synchronicity. A stable high-quality loudspeaker is necessary to produce all audible frequencies in a consistent manner across all microphones. Once the recordings have been saved, they must be transferred from the audio interface to a computer for the remainder of the experiment.

Furthermore, the audio format that the recordings are stored in should also be carefully considered. This is due to the subtle differences in compression among formats such as MP3, OGG, and WAV, to name a few. Each level of compression degrades the audio quality to enough of a degree that can be detrimental to the research. Therefore, the format currently settled on by many practitioners in this area, as is evidenced by its widespread programming support, is the WAV file format. Due to its lossless nature, it is capable of retaining much of the information caused by the subtle features and characteristics of the recording equipment, more so than MP3 or AAC compressed files.[6]

Moreover, a sufficiently long recording time must be observed to ensure that there is enough training data to extract features from and subsequently feed into the machine learning models for training in later stages. Although not all of the recorded sounds need to be used, it is wise to have ample training data that can be trimmed as needed, especially in cases where some of the segments might be contaminated with static noise or complete silence.

A multitude of software technologies must be considered following the management and storage of the recordings. For instance, the way in which the audio files should be processed depends on their size and format, as well as their propensity to be streamed. If they can be manipulated in the computer's main memory, versus having to read the entire file before being able to load a particular audio segment, it would help in achieving faster speeds in later stages due to a lesser dependence on processing capabilities.

The exact features that would be ideal for extraction from the recorded sounds should be reviewed independently, whether it is by studying scientific literature that claims to have achieved useful results, or by conducting manual tests. In actuality, the more features there are to extract and compare, the greater the processing capabilities must be. Therefore, to keep this experiment within the realms of possibility in terms of computational processing power, a more research-oriented approach was taken for the feature selection and the subsequent feature extraction stages, based on the literature review in the following section.

Lastly, after features from all audio files are extracted and saved, they should be prepared appropriately for entering the training process. One of the more accessible resources for this task is using the Python programming language, which offers the necessary coding libraries for scientific work, namely SciPy, and for machine learning, such as the scikit-learn suite of utilities. A library for plotting results in a way that is easy to understand and compare must also be considered, such as matplotlib. Python has emerged as a versatile programming language for machine learning applications, owing to its extensive ecosystem of libraries and modules that provide researchers and developers with a wide range of functionalities, and therefore it has proved to be a suitable tool for the task.[7]

To help with the orchestration of the post-recording stages, it can be helpful to establish a configuration document from which the written code could gather metadata about what that stage of the program should do. As such, a JSON object file is a plausible candidate for fulfilling these requirements, as it is designed to be easy to understand for both humans and computers.

Overall, the mixing of different technologies in various stages of a complex and layered experiment is a balancing act that is error-prone, but one in which mistakes can be avoided by taking the right precautionary steps, and with enough computing resources.

2 LITERATURE REVIEW

The study of microphones and audio forensics has spanned more than half a century since the Federal Bureau of Investigation (FBI) in the US put it to use in the 1960s.[8] In more recent years, large commercial conglomerates have amassed sizeable digital sound libraries – audiobooks, songs, snippets from videos, and speech, to name a few – and have used them to great effect to help in audio recognition, transcription, subtitling, and in the addition of other useful features to their respective platforms.

This portion of the paper discusses other researchers in the field and their contributions to audio forensics that can lend support to this experiment.

2.1 Uses of AI and ML in Audio Analysis

Over the past decade, artificial intelligence and machine learning have had a transformative impact on various industries, and the field of audio engineering and analysis is no exception. The application of AI and ML techniques in audio forensics and fingerprinting has led to significant advancements, improving the accuracy and efficiency of tasks like music recognition, noise reduction, and speech enhancement. The following sections will discuss specific examples of AI and ML applications in audio engineering, referencing publicly available scientific papers and highlighting companies that have played a crucial role in shaping the current era of research in this field.

2.1.1 Music Recognition Software

Artificial Intelligence can play a paramount role in sound interpretation. Shazam is an example of a company that uses sound samples to source their origins and creators.[24] Meanwhile, Google has furthered that capability by recognising humming, whistling, and singing. This suggests that Google has already researched and developed the technology for granular audio dissection and fingerprinting, made possible by incorporating advanced machine learning models that can analyse the melodic contour, rhythm, and other musical features of the user's input and match them against a vast database of songs.[9]

While their solutions are closed source and cannot necessarily be effectively studied up close, it is worth observing their development curve with respect to known research to potentially trace the techniques used and whether or not they align with any recent discoveries.

2.1.2 Audio Forensics

Audio forensics is a field that involves the analysis and processing of audio recordings for attribution, usually to provide evidence in legal and investigative contexts, but also in many other domains. The application of AI and ML has been critical in improving the accuracy and efficiency of various tasks in audio forensics, such as speaker identification, voice biometrics, and background noise analysis. One noteworthy example is the research conducted by Li et al. which demonstrated the effectiveness of DNNs in enhancing speaker identification accuracy.[10]

AI and ML have also been instrumental in the field of audio fingerprinting. Audio fingerprinting involves extracting unique features from an audio recording to create a digital "fingerprint" for identification and comparison purposes. Companies like Shazam and SoundHound have revolutionised music recognition through the application of AI and ML algorithms, enabling users to identify songs in a few seconds.

Another example can be found in the work done by Hennequin et al. in 2020, which explored the use of machine learning techniques in the detection of manipulated or synthesised audio recordings, known as deepfakes.[11] By developing algorithms that analyse the spectral and temporal features of audio, researchers were able to distinguish between genuine and falsified recordings with remarkable accuracy.

A study published in 2016 by Luo et al., titled "Deep Clustering and Conventional Networks for Music Separation: Strong Together", explored the use of deep learning techniques to separate individual sources from an audio mixture effectively.[12] It laid the foundation for better noise reduction and signal enhancement in various applications.

2.1.3 AI-Enhanced Noise Reduction

One of the most prominent applications of AI and ML in audio engineering is noise reduction, which has benefited greatly from the implementation of deep learning algorithms. In 2019, Nicolson et al. published a paper that introduced Deep Xi, an end-to-end deep learning (E2E-DL) model for speech enhancement.[13] This model successfully reduced noise in various environments, improving the intelligibility and quality of speech signals.

Similarly, Google's DeepMind has developed WaveNet, a generative model for raw audio that uses convolutional neural networks (CNNs) to synthesise and enhance speech signals. WaveNet has shown promising results in speech enhancement tasks, significantly improving the clarity of speech in noisy environments.[14] This signals that there is a possibility of recorded sounds being altered by microphones or specialised software, which could play a role in affecting the ability of algorithms to distinguish between different microphones in this experiment.

2.1.4 Speech Enhancement and Voice Synthesis

Advancements in AI and ML have also facilitated significant improvements in speech enhancement and voice synthesis. One remarkable example is the Tacotron 2 model developed by researchers at Google. The system uses a sequence-to-sequence model to generate human-like speech from text inputs, achieving impressive results in terms of naturalness and intelligibility.

Another breakthrough in voice synthesis comes from NVIDIA's text-to-speech system – WaveGlow[1]. It combines insights from WaveNet and Glow to synthesise high-quality speech with less computational complexity. This system has demonstrated the potential to revolutionise voice-based applications, from virtual assistants to audiobooks.

2.1.5 Speech Recognition

AI and ML have also improved speech recognition capabilities, enabling more accurate transcription and analysis of spoken language. A prime example is the research paper "Deep Speech: Scaling up end-to-end speech recognition" by Hannun et al., which introduced the Deep Speech system. This system leveraged deep learning techniques to achieve ground-breaking performance in speech recognition tasks.[15]

2.1.6 Automatic Mixing

Automatic mixing is another area where some significant impact has been propelled through the use of machine learning. By analysing the input from various instruments, AI-based systems can automatically adjust levels, equalisation, and other settings to create a well-balanced mix. A paper titled "A Semantic Approach To Autonomous Mixing" written by De Man and Reiss in 2013 demonstrated the potential of using ML techniques to achieve optimal mixes with minimal human intervention.[16]

2.2 Literature on Audio Based Fingerprinting

The following papers cover examples of research that is closely related to the subject at hand, with some even delving into the same topic, which inspired the creation of this project.

Grossberg et al. proposed ARTSTREAM in 2004, a neural network model that focuses on auditory scene analysis and source segregation.[17] The model simulates the perceptual organisation of complex auditory scenes into separate sound sources and utilises adaptive resonance theory (ART) to achieve effective segregation. ARTSTREAM addresses the problems of object formation, object selection, and the grouping of spatially separated sounds into coherent auditory streams.

In a paper in 2011, Kraetzer et al. presented a context model for microphone forensics, focusing on the identification of recording devices using microphone-specific features.[18] The model incorporates both recording conditions and post-processing into the evaluation, improving the reliability and accuracy of microphone forensics. The authors showcase the application of their context model in a variety of scenarios, demonstrating its effectiveness in differentiating between devices and processing chains. They also mention an increase in speed by a factor of 30 when reducing the number of features extracted to the top 20 most useful ones.

In 2014, Das et al. explored the idea of fingerprinting smart devices by analysing embedded acoustic components, specifically focusing on microphones and speakers.[4] The authors propose a novel method for device identification by utilising the unique hardware-based characteristics of the acoustic components. They demonstrate the effectiveness of

[1] Ryan Prenger, Rafael Valle, Bryan Catanzaro, NVIDIA Corporation. arXiv:1811.00002

their method in various real-world scenarios, achieving high accuracy in identifying devices and proving its potential as a security mechanism.

In 2018, Ferrara et al. investigated the fingerprinting of smart devices using microphone data extracted from video recordings.[19] Their study highlights the presence of unique device-specific characteristics in the recorded audio, which can be used for accurate device identification. The authors apply their method to a dataset of videos recorded using multiple smartphones, achieving promising results in terms of fingerprinting accuracy and demonstrating the potential of their approach in forensic applications.

In 2020, Chang et al. proposed a neural audio fingerprinting system for high-specific audio retrieval using contrastive learning.[20] The authors employ a deep neural network to extract discriminative features from audio signals, which are then used for content-based retrieval tasks. Their approach demonstrates improved performance compared to traditional methods, achieving better retrieval accuracy and specificity.

Later that same year, Qamhan and others explored the application of deep learning techniques in digital audio forensics, specifically focusing on the classification of microphones and environmental characteristics. The authors employ deep learning algorithms to extract features from audio signals, which are then used to identify the microphone model and the recording environment. The study demonstrates the potential of deep learning in enhancing the accuracy and efficiency of audio forensic investigations.[3]

In their research, the authors highlight the performance improvements achieved by their deep learning-based approach compared to traditional methods. The authors emphasise the importance of understanding the specific characteristics of different microphone models and environments to improve the overall reliability of their analysis. As a result, their work contributes significantly to the ongoing development and optimisation of audio forensic techniques in both academic and practical contexts.

The year 2021 also saw a research paper by Wang et al. where the authors introduced a speaker attribution method using voice profiles and graph-based semi-supervised learning. The authors create voice profiles by extracting speaker-specific features from audio recordings and then employ a graph-based semi-supervised learning algorithm for speaker identification.[21] This approach demonstrates promising results in terms of attribution accuracy, outperforming other traditional speaker recognition techniques.

Li et al. later investigated the sound source separation capabilities of various deep learning networks.[22] By comparing different network architectures, the authors identify the strengths and weaknesses of each model in separating sound sources from complex audio mixtures. This research provides valuable insights into the underlying mechanisms of deep learning networks for sound source separation, potentially guiding future model development.

In the year 2022, there was also a published research paper by Cobos et al. presenting a comprehensive overview of machine learning techniques employed in spatial audio capture, processing, and reproduction.[23] The authors discuss various applications of ML techniques, such as sound localisation, source separation, and spatial audio rendering. The review provides an in-depth analysis of the current state-of-the-art methods and highlights promising research directions for future advancements in spatial audio processing.

2.3 Influential Companies in Audio Forensics and Fingerprinting

Several more companies have played a pivotal role in the development and implementation of AI and ML, along with other similar techniques, in audio engineering and analysis.

2.3.1 Shazam

Shazam is one of the most popular music recognition software that has revolutionised the way people discover music. Founded in 1999, the company leverages ML algorithms to analyse audio fingerprints and match them to a vast database of songs.[24] Its flagship product can recognise songs and tunes within seconds, even in noisy environments, making it a pioneer in the audio fingerprinting domain. Its solutions in this field have been studied up close and were used as a basis for future research.

2.3.2 SoundHound

SoundHound is another popular music recognition application that leverages ML techniques to identify songs and provide related information, such as song lyrics and details of the performers. The company's Houndify platform employs natural language processing (NLP) and deep learning algorithms to create voice-enabled applications and virtual assistants. In a paper by Ghias and others back in 1995, the authors discussed a large-scale music search system based on audio fingerprinting and the use of parallel processing to achieve efficient and accurate song identification.[25]

2.3.3 Audible Magic

Audible Magic is a company that specialises in content identification and copyright compliance solutions. Its patented technology uses digital fingerprinting techniques to identify copyrighted material in various media types.[26] Their robust solutions have been widely adopted by social media platforms, streaming services, and content creators, playing a significant role in shaping the current era of audio forensics.

2.3.4 SONY CSL (Computer Science Laboratories)

SONY CSL has been a pioneer in the development of AI-driven audio engineering tools. In 2003, their research team published a paper titled “Popular Music Access: The Sony Music Browser” that discussed audio fingerprinting techniques and data mining techniques that can be used to achieve an audio discovery experience. In a paper by Briot et al. in 2017, partly in collaboration with SONY CSL, the authors present a comprehensive review of deep learning techniques for music generation, including an overview of SONY CSL's research in this area.[27] The Flow Machines project[2], developed by SONY CSL, uses AI to compose music in various styles, demonstrating the potential of machine learning in creative audio applications. The continued investment in research and development in audio technologies by SONY has contributed to advancements in the fields of audio engineering, analysis, and forensics.

[2] Product page: https://www.flow-machines.com

2.3.5 iZotope

iZotope is a leading audio technology company that specialises in developing software and plug-ins for audio professionals. Their flagship product, RX, is an audio repair suite that employs AI and ML algorithms for noise reduction, dialogue isolation, and spectral repair. iZotope's advancements in audio restoration have been highlighted in scientific papers, such as a study by Saeki and others in 2022 that presents an algorithm for separating vocals from music using non-negative matrix factorisation (NMF).[28]

2.3.6 Dolby Laboratories

Dolby, a global leader in audio technology, has incorporated AI and ML in its noise reduction and audio enhancement solutions. For example, Dolby's Voice Call product includes "advanced spatial audio" noise suppression technology, which uses advanced audio processing technology to enhance speech clarity and reduce background noise in real time.[29]

2.4 Summary

The past decade has witnessed remarkable advancements in audio engineering and analysis, driven by the increasing application of various machine learning techniques. From music recognition applications like Shazam and SoundHound to ground-breaking research in noise reduction, speech enhancement, and voice synthesis, AI and ML have revolutionised the field of audio engineering and have affected adjacent ones as well, with a notable one being audio fingerprinting.[24]

Companies such as iZotope, Dolby Laboratories, and SONY CSL continue to shape the landscape of audio forensics and fingerprinting, pushing the boundaries of what is possible in this technological area. As research and development in AI and ML continue to progress, it can be expected that even further significant breakthroughs and applications in the realm of audio engineering and analysis take place.

3 METHODOLOGY

The purpose of this project is to find out whether it is possible, in small laboratory conditions, to fingerprint microphones based on their recordings. As such, there are hardware constraints involved that will limit the accuracy of the results, but not to an extent that nullifies the experiment. The methods of carrying out the recording and audio processing aspects are robust and reproducible, and they will therefore be listed in this section as neutrally as possible to allow others in the open-source community to follow them in the same way.

While experiments involving audio dissection and analysis can be aided by machine learning, it is equally vital to involve fundamental mathematical principles to avoid premature conclusions about which is the most effective strategy. To this end, this section will explore the possibility of simplifying the processing of audio data to render the experiment more effective with limited hardware capabilities. This involves the use of automation by implementing certain functions in code.

3.1 Overview

The steps of the implementation of the experiment, comprising the overall pipeline of the project, can be generalised as follows:

1. Recording: The first step is to decide on a hardware setup, choose a set of microphones and a loudspeaker, pick out a set of audio samples, record them using the prepared hardware, and store the audio data on a storage medium using an appropriate file format.

2. Data Pre-processing: The next step is to pre-process the acquired data by normalising the features, bringing them to an easy-to-work-with range (0 to 1), and splitting the resulting dataset into training and testing batches. The value of cross-validation is paramount, as it is essential in ensuring that the model does not overfit the training data, which can result in a disproportionately high number of false positives and false negatives.

3. Feature Extraction: In this step, the goal is to extract the relevant features from each audio recording in the same way, in particular the more common features that are used in audio analysis, including Mel Frequency Cepstral Coefficients (MFCCs), Spectral features, and Time-domain features. Python's Librosa library can help extract these key features.[35]

4. Machine Learning Algorithm Selection: A plethora of machine learning algorithms can be used for this task, such as Support Vector Machines (SVM), Random Forests, and Convolutional Neural Networks (CNN). Beginning with a simple algorithm like SVM can help to benchmark the overall performance and to solidify the experiment pipeline, making switching to another algorithm a much simpler task. The algorithms can later be compared, and the best one can be determined based on the best prediction accuracy and model performance.

5. Model Training: The training data is fed into the model, using a library such as Python's scikit-learn as the core implementation of the chosen machine learning algorithm. A more power-efficient and performant computer would greatly benefit this stage, as it is arguably the most arduous phase of this project. After the model is trained, it must be saved to preserve the progress.

6. Model Evaluation: To evaluate the model's performance on the test data, metrics like accuracy, precision, and recall are included with the built-in machine learning libraries. Those can be contrasted with later predictions of unknown data to determine whether there is a bias in the training methods.

7. Predictions: Finally, the trained model can be put into practice to predict the microphone used for any new recordings, with the results being displayed in a confusion matrix, pitting each of the model's predictions against the correct answers.

Following the definition of the project pipeline, the rest of the experiment can be largely automated in the context of code and programmable solutions. For instance, extracting features and testing different ML classifiers can be scripted to remain consistent between iterations. The aforementioned stages of the project will be further expanded upon in the following sections.

3.2 Recording

This part covers the first phase of the project pipeline – choosing the hardware and audio, recording with the help of the prepared hardware, and storing it in the proper format.

3.2.1 Microphone sensitivity

The human auditory system is capable of hearing in the spectrum of approximately 20 Hz to 20 kHz.[30] Microphones on the other hand do not have a fixed maximum sample rate, as they are analog devices.[31] As each microphone model differs in the manufacturing stage, coupled with possible micro-defects during production, the sensitivity will also differ during regular usage. As a result, experiments such as this can help determine the efficacy of detecting anomalies to a reasonable degree of accuracy, and enough to conduct microphone fingerprinting.

Manufacturers of microphones tend to adhere to a standard when it comes to measuring the sensitivity of their products. A 1 kilohertz (kHz) sine wave tone is usually used to determine the hearing range. The amplitude or magnitude of the output from the microphone is recorded, which means that any audible frequency is detectable, as well as those outside of the human hearing range. A sine sweep, where a continuous tone starts playing at a low frequency and gradually increases it to a certain maximum, is another example of the same technique.

Although this experiment is limited to dedicated microphones, more questions arise when considering the widespread use of smartphones as recording devices. If it were to be conducted on smartphone microphones instead, the miniature size would be a key factor and would largely affect the ability to pick up certain frequencies. However, there would also exist discrepancies between each smartphone microphone due to manufacturing imperfections. In addition to hardware factors, the presence of sound processing software in modern mobile operating systems lays claim to affecting the recording results, which could sway the outcome of the experiment in either direction.

3.2.2 Recording specifics

The length of time chosen to conduct the recordings is 6 hours, from which the audio will be segmented into smaller clips, splitting based on silence. A longer recording can be beneficial when faced with differing setups on separate occasions, keeping the environment and microphone settings consistent with each other.

The audio recordings should be sourced from license-free and copyright-free sources in the public domain, and they should ideally include human speech. Many suitable examples exist on the open internet, such as audiobooks and church songs, among others.

3.2.3 Summary

In summary, the recording phase of this research experiment can be described as follows:

1. Record from all microphones at the same time to capture the same reverberations, so that time is not a differential factor.
2. Keep the microphones at the same distance from the source of the sound. Position them in a circle to ensure equal exposure.

3. Use naturally sounding human speech, music with a mixture of bass and treble, and a sine sweep to produce as many distinct frequencies and combinations of sounds as possible to allow for more in-depth feature extraction.
4. Initially prepare a 5-minute recording segment for later phases, then keep doubling. It may be wise to trim a longer recording into appropriately sized pieces in advance or write some code to do the trimming in the computer's main memory without sacrificing storage space.

3.3 Data Pre-processing

This part covers the phase where the recorded audio files are prepared for use in the feature selection, feature extraction, and training stages.

3.3.1 Training and Testing dataset sizes

The selection of appropriate proportions of recordings for training and testing is crucial for the performance and generalisability of a machine learning classifier. Typically, the dataset is divided into two or three non-overlapping subsets, with the majority of the data allocated for training, and the remaining portion reserved for validation and testing.[32] A common approach is to follow the 70-30 or 80-20 rule, wherein 70% or 80% of the dataset is used for training and the remaining 30% or 20% for testing.[33] Another strategy is to use cross-validation, particularly k-fold cross-validation, where the dataset is partitioned into k equally sized subsets, and the model is trained and validated k times, using a different subset for validation each time.[34] This ensures that each data point is used for validation exactly once, thus providing a more robust evaluation of the model's performance.

The choice of the proportion of data for training and testing depends on factors such as the size and diversity of the dataset, the complexity of the model, and the desired level of generalisation. In the context of microphone fingerprinting, ensuring that the training set includes a wide variety of microphones, recording conditions, and sound sources can lead to better generalisation and improve the model's ability to handle unseen data.[19] Ultimately, the goal is to strike a balance between maximising the available data for training and having a sufficient amount of data for testing to ensure a reliable evaluation of the classifier's performance. For that reason, this work aims to adhere to a common standard, such as the 80-20 rule, with room for tuning on occasions when it is needed to make adjustments to the sizes of the data sets.

3.3.2 Trimming of recordings

Recordings of several hours are likely to contain excessive amounts of data that would only contribute to the artificial inflation of positive results. Picking out certain fixed-length segments from the recordings is an approach that could help limit the information surplus, which would narrow the focus on the essential parts of the audio, and hence enhance the model's performance and accuracy.

Such functionality could be implemented in code and integrated within the rest of the pipeline. A pseudocode version could look as follows:

```
Load recording file
Set current_minute and end_minute
While current_minute < end_minute
    Retrieve current_minute until current_minute + 1 of recording
    Save to disk
    Increment current_minute
```

Figure 3A – Pseudocode for trimming an audio file

More specific details, from the audio bitrate to the way the files are stored, are expected to be different, considering the option of keeping the data in the main memory to avoid costly writes onto the hard disk.

3.3.3 Splitting of recordings

One crucial step is the splitting of audio recordings into smaller segments. There are several reasons for segmenting the audio, including working with limited computational capacity, reducing the impact of background noise or distortion, and ensuring a more manageable and efficient analysis process.

Limited computational capacity is a significant concern in audio analysis, as processing long audio files can be time-consuming and resource intensive. By splitting the audio into smaller segments, the computational requirements are reduced, making it feasible to perform the experiment on machines that have limited resources, and greatly improving the overall analysis efficiency.[35]

Another advantage of splitting audio recordings is to minimise the impact of background noise or distortion that may be present in the original signal. By focusing on shorter segments, we can isolate and analyse portions of the signal that exhibit distinct characteristics of the microphone and other recording equipment, thereby improving the accuracy of the fingerprinting process.

Splitting the audio based on silence is an effective method for dividing the recordings into meaningful segments. By identifying periods of silence within the audio, we can ensure that the resulting segments correspond to distinct events or sounds, reducing the likelihood of introducing artefacts or irrelevant features into the analysis.[36]

The following is a generic pseudo-code for splitting audio recordings based on silence:

```
Load recording file
Set silence_threshold and min_length
Initialise segment_start and segment_end
For sample in recording file
    If sample < silence_threshold
        Update segment_end
        If segment_end – segment_start >= min_length
            Save between segment_start and segment_end
    Else
        Update segment_end
```

Figure 3B – Pseudocode for segmenting audio files by silence

By applying this splitting method, we can generate smaller audio segments that retain meaningful information, while mitigating the issues of computational complexity and background noise, and reducing the discontinuity of speech.[37]

3.4 Feature Extraction

The methodology of our microphone fingerprinting experiment relies on feature extraction to represent and describe the distinctive characteristics of each microphone and other recording equipment that affects the resulting sound capture. To get accurate and trustworthy fingerprinting results, it is vital to choose the most relevant and appropriate elements.

A select number of properties, including spectral, cepstral, and time-domain traits, can be taken into account. The following characteristics are among the most important to consider for this experiment.

3.4.1 Mel-frequency cepstral coefficients (MFCCs)

MFCCs are in widespread use among speech and audio processing tasks, as they can capture the spectral envelope of the signal while remaining less sensitive to noise pollution and other variations.[38] MFCCs have previously been shown to be effective in capturing microphone-specific characteristics.[39]

3.4.2 Spectral centroid

The spectral centroid is a measure of the "centre of mass" of the frequency components of a signal and can indicate the perceived brightness or spectral balance of the sound.[40] This feature may be useful in distinguishing between different microphones with varying frequency responses.

3.4.3 Harmonic-to-noise ratio (HNR)

The HNR is a measure of the periodicity of the signal, quantifying the ratio of harmonic components to noise components in the signal.[41] This feature may help to identify the differences in the harmonic structure and noise characteristics of recordings made with different microphones.

3.4.4 Zero-crossing rate (ZCR)

The ZCR measures the number of times a signal crosses the zero-amplitude level within a given period. This time-domain feature can provide insights into the temporal properties and frequency content of the signal, which the unique characteristics of a microphone can influence.

The following **Figure 3C** illustrates the idea behind the ZCR concept:

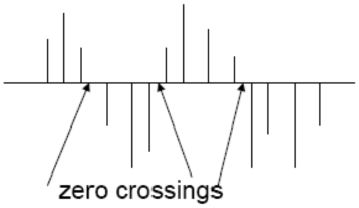


Figure 3C – Points at which the signal passes zero[3]

3.4.5 Summary

It is possible to extract these features through the use of various signal-processing techniques and programming libraries, such as Python’s Librosa module.[35] By combining these features into a large and comprehensive feature vector, it becomes simpler to capture the unique properties of each microphone and use this information to train and evaluate the fingerprinting model.

3.5 Training

This part of the experiment is one that requires the most heuristics, as it is the determinant of how the model will turn out performance-wise. Training is a process that involves feeding correctly prepared data into a classification algorithm that can use its mathematical and probabilistic formulae to reach certain conclusions. In this subsection, we discuss the choice between supervised and unsupervised approaches to machine learning, and the importance of hyperparameter tuning to achieve optimal performance.

3.5.1 Supervised learning

This machine learning paradigm gives the algorithm input-output pairs of labelled training data. The learning process aims to find a relationship between the inputs and the outputs so that the model can generalise to new data.[42] In classification tasks like microphone fingerprinting, where each recording is connected to a particular microphone label, supervised learning is frequently used.

3.5.2 Unsupervised learning

On the other hand, unsupervised learning is an approach where the algorithm is not provided with any labels during the training phase. Instead, the model learns by itself, trying to garner extra information and identify certain patterns or structures in the input data, often in the form of clusters or representations.[43] Unsupervised learning can be useful in situations where labelled data is not available or when the goal is to discover unknown relationships in the data.

[3] Image sourced from: Bachu R.G., Kopparthi S., Adapa B., & Barkana D.B. 2009. Voiced/Unvoiced Decision for Speech Signals Based on Zero-Crossing Rate and Energy. https://dx.doi.org/10.1007/978-90-481-3660-5_47

3.5.3 Optimal learning approach

For this experiment, supervised learning is more appropriate due to the nature of the task. Since the goal is to classify audio recordings based on the microphones they came from, having labelled data with the corresponding microphone information is essential to train an accurate and dependable model. Furthermore, supervised learning algorithms, such as Support Vector Machines (SVM), Random Forest (RF), and Convolutional Neural Networks (CNN), have been shown to perform well in similar classification tasks.[3]

In addition to selecting an appropriate learning approach, the optimisation of hyperparameters is crucial to achieving the best possible performance. Hyperparameters are adjustable settings that control the behaviour of the learning algorithm.[44] Examples of hyperparameters include the learning rate, regularisation parameters, and the number of hidden layers in a neural network. Tuning these hyperparameters can strongly influence the model's ability to generalise to new data, and it is customary practice to use techniques such as grid search or Bayesian optimisation to find the optimal settings.[45]

In conclusion, this project benefits most from a supervised learning approach, given the nature of the classification task and the availability of labelled data. Additionally, optimising the hyperparameters of the classifier can play a key role in ensuring the model's performance is as accurate as possible.

3.6 Testing

Once the model has been trained and saved, the remainder of the experiment amounts to little more than benchmarking its performance and generalisation capabilities. It starts by presenting to the model a set of previously unseen audio recordings. These recordings have been held out from the training dataset to assess the model's ability to generalise to new data. It is crucial to avoid reusing the same data in both the training and testing phases, to ensure that the evaluation provides an unbiased estimate of the model's performance.

Any number of metrics can be used to monitor and compare different iterations of testing the model. There exist multiple metrics that are useful for this experiment. They include, but are not limited to, the following:

- **Accuracy** – This metric simply measures the proportion of correctly classified recordings out of the total number of recordings in the test dataset. It is intuitive and widely used in the evaluation of classification tasks, although it may not be suitable for imbalanced datasets where there is one dominant class.[46]
- **Precision** – Also known as the positive predictive value, it measures the fraction of true positive predictions out of all true positive and false positive predictions.[47]
- **Recall** – Also known as sensitivity, it measures the fraction of true positives out of all actual positive instances.[47]
- **F1-score** – This is the harmonic mean of precision and recall and provides a balanced measure of both metrics, especially useful when dealing with imbalanced datasets.[47]

- **Confusion Matrix** – This metric is a tabular representation of the true labels against the predicted ones, providing a simple but comprehensive view of the classifier's performance. The confusion matrix can help to identify the specific classes with which the model had difficulty distinguishing from the others.[48]
- **Area Under the Receiver Operating Characteristic Curve** – This metric measures the trade-off between the true positive rate and the false positive rate at various thresholds. A value near 1 means excellent performance, while a value close to 0.5 suggests that the model is no better than random guessing.[49]

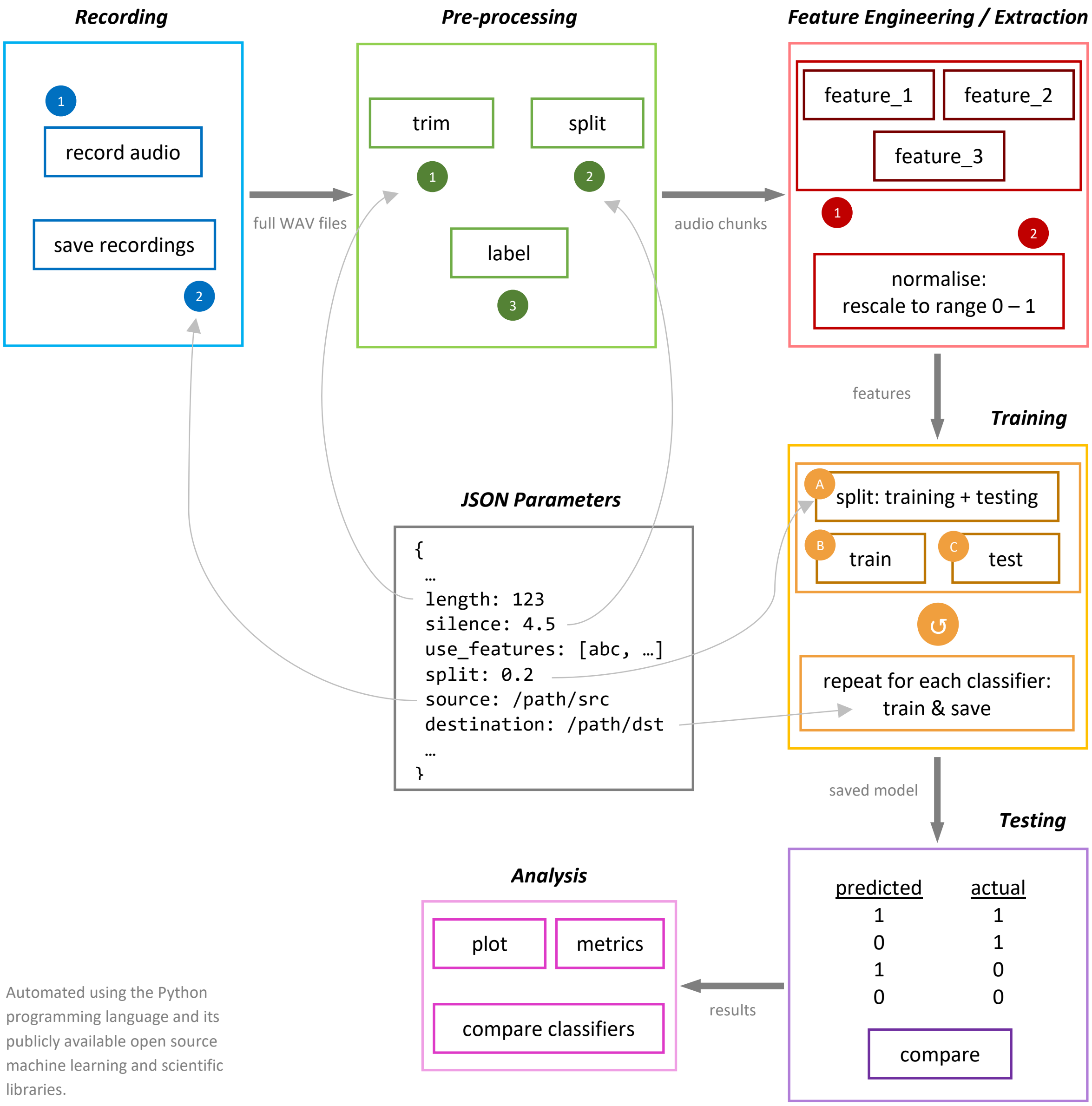
3.7 Pipeline automation

The main gripe a future researcher may have with reproducing the steps conducted in this work is the need to tweak individual parameters inside the code. If the experiment proves ineffective against X minutes of recordings, then it should be simple to change X to another Y without touching the code. To avoid getting into the weeds of programming specifics, a more future-proof approach should be taken.

To this end, the recommended way of keeping track of the experiment's parameters is by parameterisation through the use of a configuration file. In this way, the main program reads from this file, sets up the training data based on the settings specified inside the file, and carries out the complete process without the need to prompt the user to input anything, amounting to almost complete automation.

3.8 Summary

After the recording and storage of the audio, the rest of the experiment revolves around pre-processing it and sending it through a series of steps to be used for training and testing. This can be illustrated as a so-called pipeline, where some data starts on one end and, after being transformed by going through various stages that fundamentally change it, ends up on the other side. The diagram shown in **Figure 3D** demonstrates a generalised version of the project pipeline, from start to finish.



```
{
 …
 length: 123
 silence: 4.5
 use_features: [abc, …]
 split: 0.2
 source: /path/src
 destination: /path/dst
 …
}
```

Automated using the Python programming language and its publicly available open source machine learning and scientific libraries.

Figure 3D – Project pipeline

In step 1, each microphone maintains its own sound capture stored in separate files. To speed up input and output operations to and from the hard disk, step 2 will start by splitting each piece of audio evenly, based on the quiet parts of the recording where there is usually less speech or a long pause, labelling, and finally storing in the same folder as the original recording. With each microphone having its own folder of sound recordings, it is easier to pick training back up from this phase without expending a surplus of time and computational power on re-segmenting the audio, as the data is already pre-processed.

The rest of the pipeline shows the feature engineering and extraction phases to help identify useful acoustic peculiarities that can contribute to identifying the correct microphone, with the last step being the machine learning aspects related to training and testing, and saving the model to disk for quicker use in the future.

In summary, all of the steps are prone to parameterisation due to the heuristics required in this experiment. These parameters can be stored as key-value pair settings, which can be used as inputs to functions and as determinants of whether to use one training algorithm over another. The basis of this idea is to be able to automatically tweak the parameters based on a desirable end state. If a 5-minute recording ended up doing a poor job of figuring out which microphones recorded which audio clip, it should be trivial to use the program to be able to adjust the source audio clip to a 10-minute segment, and so on, with the hope that, given the limited resources available, 6 hours will be a sufficiently large limit to be relied upon if the shorter recordings fail to yield any results. As such, the whole pipeline becomes efficient at dealing with changes as they arise, making the experiment more accessible and reproducible by researchers and hobbyists alike.

4 IMPLEMENTATION

In this section, we describe a step-by-step approach to the aforementioned pipeline and give insights into the practical limitations when carrying it out. It delves into the implementation details of the experiment, outlining the various steps undertaken to achieve the goal of microphone fingerprinting using machine learning algorithms. The implementation process can be broadly divided into the following steps: data collection and pre-processing, feature extraction, model selection and training, and evaluation of the results.

4.1 Recording

The goal of this part is to outline a plan that can guide the recording aspects of the project. This is the first phase in the project pipeline, followed by pre-processing, feature extraction and tweaking, training, testing, and finally analysing the results.

4.1.1 Equipment

- **Recording** – 5 microphones, 4 of which are different
- **Accessories** – mounting gear and cables for all microphones
- **Audio interface** – TASCAM Model 12 (with an SD card for saving the data)
- **Sound** – speakers and audio file (recording of a copyright-free audiobook)
- **Storage** – laptop running audio recording software (in case of an audio interface mishap)
- **Environment** – sound-proof chamber (to a reasonable degree)
- **Time** – 2 days

4.1.2 Procedure

1. Arrange the microphones by mounting them in a semi-circular formation.
2. Place the source of sound (speakers) at an arm's length from the microphones, facing them.
3. Connect the speakers to the laptop or another source of audio.
4. Connect the audio interface to the microphones. In case the SD card presents problems:
    a. Connect the audio interface to the laptop as well.
    b. Use recording software on the laptop to process the audio and save it to disk.
5. Turn on all of the prepared equipment and verify that the microphones are recording.
6. Erase the SD card and start anew, recording only the sound coming from the speakers.
7. Leave the room undisturbed for 6 hours.
8. Stop the recording.
9. Make a digital copy of the SD card data.
10. Verify the audio file integrity by playing the recordings at various timestamps.

4.1.3 Limitations

- The TASCAM Model 12 comes with an SD card and claims it is capable of storing up to 24 hours of audio, though no tests were conducted to verify this. Nevertheless, the required 6 hours of recording per microphone, as established in the methodology section, were not able to fill the storage space to the fullest extent.
- Arranging and moving the microphones ever so slightly may lead to noticeable disturbances, distortions, and perturbations, and hence can produce an inconsistent recording. The environment for all the microphones can only be kept the same up to a certain point.
- The recording room may also have equipment that can alter the way the sound bounces around and reaches the microphones. All of these factors must be taken into consideration.
- The bar for recording was set to 6 hours per microphone before it was known whether the feature extraction and machine learning algorithms needed more data to work with.

The hardware arrangement of the experiment can be seen in **Figure 4A** below, consisting of 2 audio interfaces, a loudspeaker, and 5 microphones. One audio interface is used for adjusting and equalising the audio passed to the speaker, while the other is for recording the microphones.

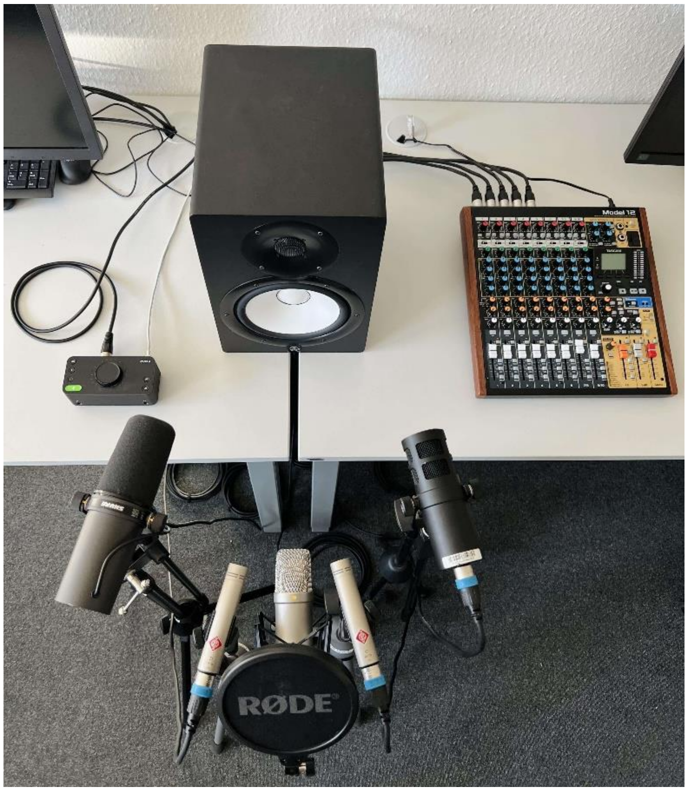


Figure 4A – Recording equipment setup

The following microphones are shown in **Figure 4A**:

- Shure SM7B – leftmost
- The t.bone MB 7 Beta – rightmost
- Neumann KM 184 – 2: centre-left and centre-right
- RØDE NT1-A – centre

The inclusion of two identical models of a microphone is intentional. As previously mentioned, it is important to test if microphones of the same make and type have enough of a difference in their sound profiles that an ML algorithm could figure out that difference.

4.2 Pre-processing

The experiment continues with a review of the recorded audio files. This stage prepares them to undergo feature extraction.

4.2.1 Sanity check

The so-called sanity check assures that the sound files are correctly stored and retrievable in their current state. A three-second snippet of one of the samples taken at random produces the power spectrogram shown in **Figure 4B**.

Spectrograms are a visual representation of the energy (or power) distribution of a signal across different frequencies over time. In simple terms, it is a way to visualise how the frequency content of an audio signal, be it music or speech, changes over time. The colour at each point on the graph corresponds to the signal's intensity at that specific frequency and time.

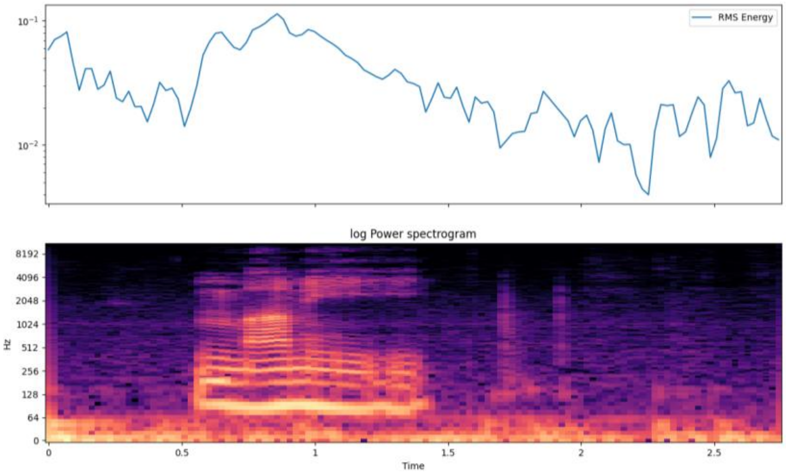


Figure 4B – Visualisation of spectral features

4.2.2 Trimming

Once the audio has been verified to function correctly, it needs to be trimmed to an appropriate length to make sure that it is feasible to extract features using limited resources. The chosen way to go about this is to find out the sample (or frame) rate of a given WAV sound file, multiply it by the number of seconds that need to be retrieved, and read that many frames. **Figure 4C** depicts a code snippet that achieves this in Python, with adjustments to the parameters to be able to specify a start and end minute.

```python
def trim_sound(input_file, output_file, start_minute, end_minute):
    with wave.open(input_file, 'rb') as wav_in:
        params = wav_in.getparams()
        start_frame = int(start_minute * 60 * wav_in.getframerate())
        end_frame = int(end_minute * 60 * wav_in.getframerate())
        frames_to_read = min(end_frame - start_frame, wav_in.getnframes() - start_frame)
        wav_in.setpos(start_frame)
        audio_data = wav_in.readframes(frames_to_read)
        with wave.open(output_file, 'wb') as wav_out:
            wav_out.setparams(params)
            wav_out.writeframes(audio_data)
```

Figure 4C – Python code to trim a WAV audio file

4.2.3 Splitting by silence

The next step is to prepare the audio files for feature extraction. As discussed in the methodology section, the audio needs to undergo a segmentation process, which will find silent parts and use them as a breaking point, dividing a long recording into shorter pieces.

The code for splitting the sound files is shown in **Figure 4D** below. The code relies on the pydub audio manipulation library. The minimum duration and other settings are set by parameters specified in a JSON file. This configuration file also hosts the settings for all other parts of the experiment, including the previously mentioned trimming step.

```python
def break_silence(input_file, output_folder, min_duration=1, silence_thresh=-70, keep_silence=100):
    audio = AudioSegment.from_wav(input_file)
    segments = split_on_silence(
        audio,
        min_silence_len = int(min_duration * 1000),
        silence_thresh = silence_thresh,
        keep_silence = keep_silence
    )
    for i, segment in enumerate(segments):
        output_file = os.path.join(output_folder, f'part-{i:03d}.wav')
        if not os.path.isdir(output_folder):
            os.makedirs(output_folder)
        segment.export(output_file, format='wav')
```

Figure 4D – Python code to split a WAV audio file by silence

4.3 Feature extraction

From the aforementioned features in the methodology section, the MFCCs were chosen to be extracted from the audio files. This is due to their compact nature, efficiently packing useful sound characteristics, akin to the human auditory system. The way in which the MFCCs were extracted was through the use of Librosa. The sum of all the MFCC values over the extracted number of frames (n) was taken, and was used as the final feature measurement:

$$\frac{\sum_{0}^{n} \mathrm{MFCC}(n)}{n}$$

Figure 4E – Formula used for calculating features over a given period

Although other features were initially extracted alongside, they were found to have encumbered the MFCCs from playing a role in the training process, causing the results to be more mixed than accurate. This could be due to the mixture of time-domain and spectral types of features, although more time spent on testing is needed to confirm this hypothesis.

4.4 Training

The last step before launching a series of tests and performance benchmarks is to train classifiers on the prepared audio features. The following classifiers were trained, and are subsequently reported on in the following section of this paper:

- sklearn.svm.SVC
- sklearn.neural_network.MLPClassifier
- sklearn.neighbors.KNeighborsClassifier
- sklearn.ensemble.RandomForestClassifier
- sklearn.ensemble.GradientBoostingClassifier
- sklearn.linear_model.LogisticRegression
- sklearn.naive_bayes.GaussianNB

The hyperparameters were kept at their respective default values, and the random seed values were all set to 42 to ensure a fair comparison between the classifiers. Though the classifiers all came from the same programming library, the code could be extended to make use of other frameworks, including PyTorch and Tensorflow, to name a couple.

4.5 Summary

The implementation covered the recording and data processing aspects of the experiment in great detail. Although not all of the used code was included, the full program and its dependencies are listed in the project's freely available GitHub repository.[5]

Once the prototyping of the code was completed, the JSON parameters were kept in a file of their own, as shown in **Figure 4F** below.

```json
{
  "paths": {
    "root_path":    "/path/to/sounds",
    "train_x_ft":   "train-x.features",
    "train_y_ft":   "train-y.features",
    "test_ft":      "test-x.features",
    "model_name":   "example.model"
  },
  "general": {
    "keep_silence":false,
    "min_silence": 1.4,
    "threshold":    -80,
    "test_size":    0.2
  },
  "training": {
    "train_src":    "/2023-02-09",
    "train_dst":    "/training",
    "train_start": "random.randint(0, 30)",
    "train_end":    "train_start + 10"
  },
  "testing": {
    "test_src":     "/2023-02-10",
    "test_dst":     "/testing",
    "test_start":  "train_end + random.randint(0, 30)",
    "test_end":     "test_start + 5"
  },
  "evaluate": [
    "training.train_start",
    "training.train_end",
    "testing.test_start",
    "testing.test_end"
  ]
}
```

Figure 4F – JSON parameters example

4.6 Dataset availability - Kaggle

The raw audio files that were used to conduct this experiment are available on the Kaggle data science platform via the following link: https://www.kaggle.com/datasets/victorazzam/microphone-fingerprinting

The compressed size of the data set is approximately 4.5 gigabytes, whereas the actual data after decompression amounts to approximately 13 gigabytes.

It is licensed with the GNU General Public License version 2 (GPLv2) in the same way as the GitHub code repository of this project.

5 ANALYSIS OF RESULTS

Testing several unique machine learning classifiers, there are notable differences that can be observed in the tests and their resulting confusion matrices that can signify which of the algorithms are more suited to this type of experiment involving audio analysis.

5.1 Classifier comparison

The following figures show the results produced by each machine learning classification algorithm, including the both the average F1-score and confusion matrix.

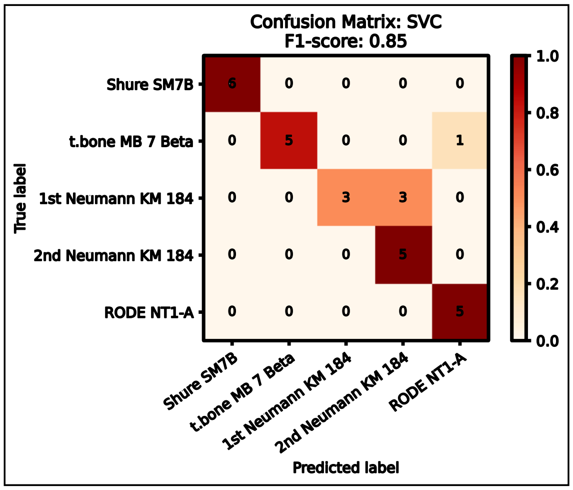


Figure 5A – Support Vector Machine

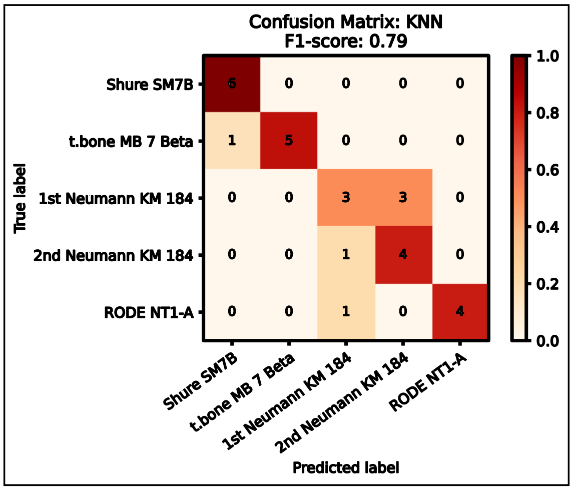


Figure 5B – K-Nearest Neighbour

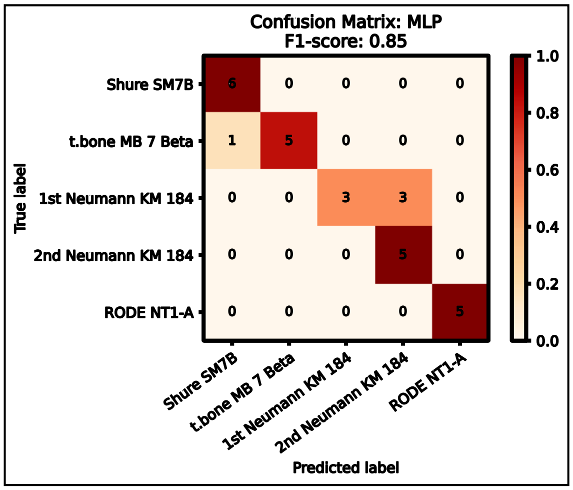


Figure 5C – Multi-Layer Perceptron

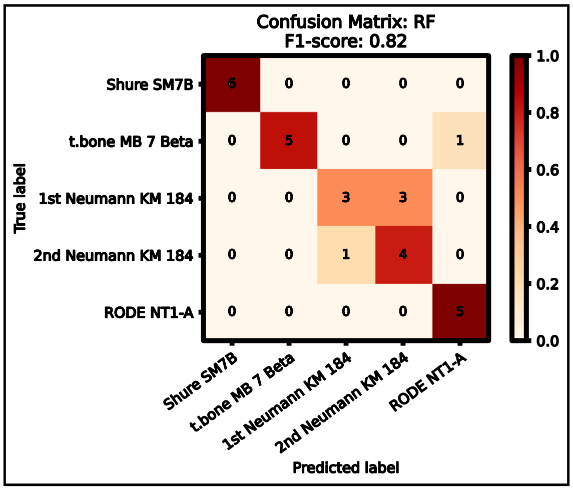


Figure 5D – Random Forest

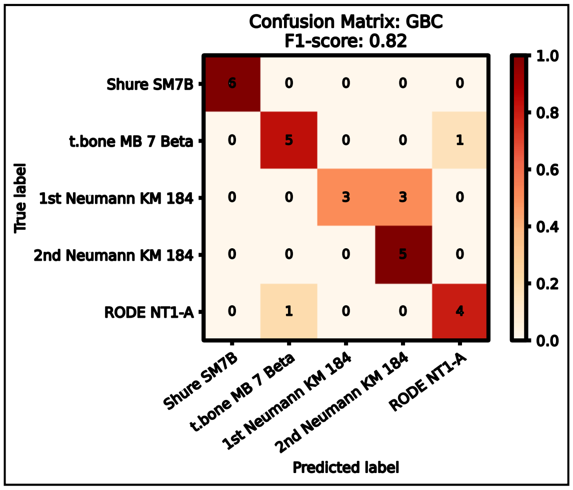


Figure 5E – Gradient Boosting

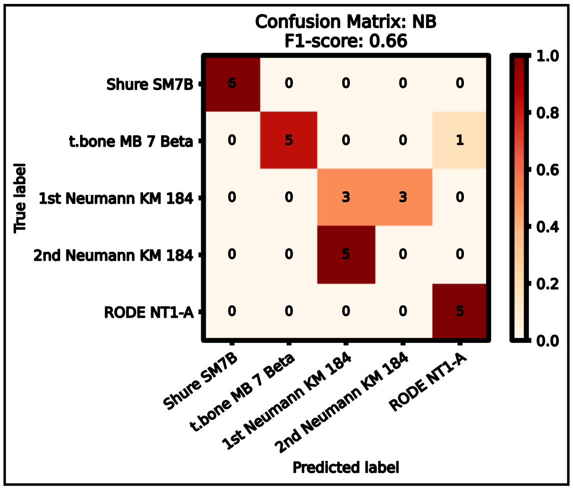


Figure 5F – Naïve Bayes

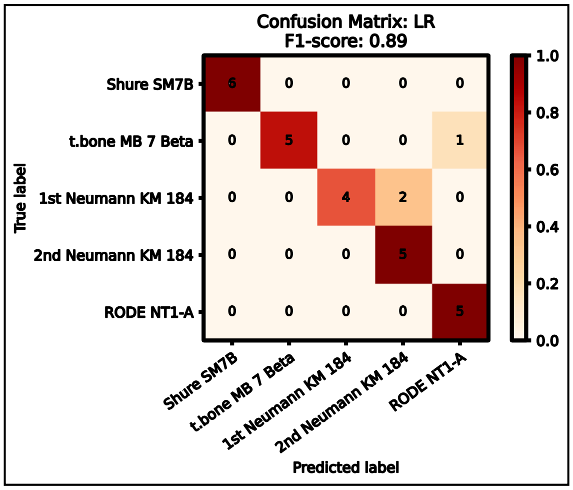


Figure 5G – Linear Regression

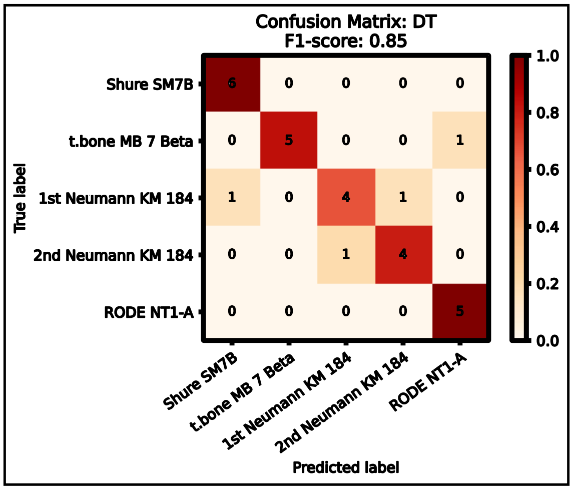


Figure 5H – Decision Tree

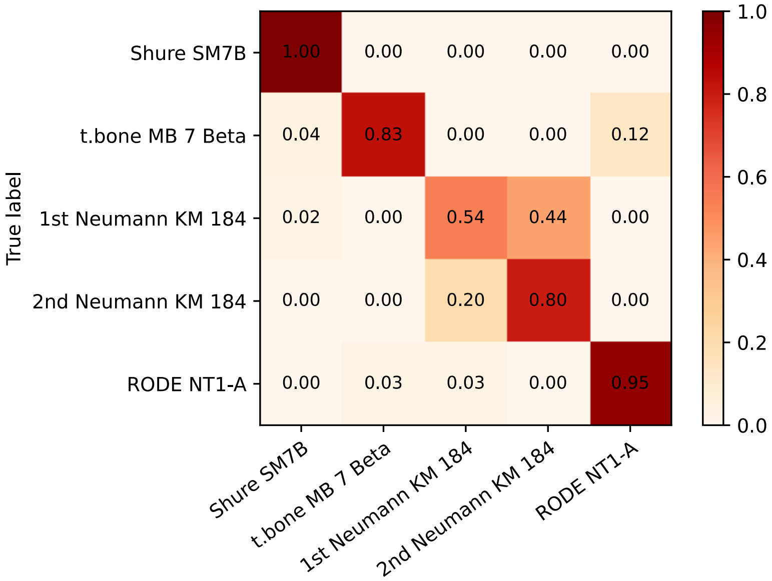


Figure 5I – Summary of all confusion matrices

Class	Precision	Recall	F1-score	Support
mic 1	0.96	1.00	0.98	6
mic 2	0.98	0.83	0.90	6
mic 3	0.77	0.50	0.59	6
mic 4	0.54	0.80	0.64	5
mic 5	0.87	0.95	0.90	5

accuracy	0.81	0.81	0.81	0.81
macro avg	0.82	0.82	0.80	28.00
weighted avg	0.83	0.81	0.81	28.00

Figure 5J – Summary of all classification reports

5.2 Preliminary conclusions

Observing the F1 score, it becomes apparent that the majority of classifiers objectively scored high in this classification task. As previously mentioned in the Methodology section, the F1 score is a harmonic mean of precision and recall. The formula for this is shown in **Figure 5X** below:

$$p = \frac{TP}{TP + FP}, \qquad r = \frac{TP}{TP + FN}, \qquad \mathrm{F1} = \frac{2\,p\,r}{p + r}$$

Figure 5K – Formula for calculating the F1 score

A high F1 score suggests that the model has low false positives (i.e., it rarely misidentifies the microphone) and low false negatives (i.e., it rarely fails to identify the correct microphone when it is present). This implies that the microphone fingerprinting model is effective and reliable in discerning the unique characteristics of different microphones from their recordings.

It is worth noting that, despite the highly accurate results produced by both LR and MLP classifiers, there are a few factors that could have spurred a bias during the training process.

For one, a chance of overfitting always poses a considerable risk to the accuracy and reliability of a trained machine learning model. In a machine learning context, overfitting is an undesirable behaviour that occurs when a trained model gives accurate predictions for training data but not for new data.[50] In theory, this would defeat the purpose of training altogether, as working with unseen data is necessary for a model to be functionally useful.

Another potential problem arises if both the training and testing data are conducted in the same environment. The noise and hardware setup from one recording period to the next may differ enough to render the recordings of even the same exact pieces of equipment completely different on those alternate occasions. A common way of counteracting such issues is to expose the model to a more diverse training dataset and to prolong the training period by a significant factor, thereby ensuring that the model is more flexible and adaptable to unseen data, and increasing the reliability of its predictions.

5.3 Method of improving the results

A dimensionality reduction technique such as PCA could help identify and capture the most important information from the high-dimensional feature space while discarding the least significant components. In the context of this experiment, the audio features extracted from microphone recordings might exhibit high dimensionality, which can increase computational complexity and the likelihood of overfitting. By applying PCA, essential features that best describe the variability in the data are preserved, while reducing the number of dimensions. The transformed features, represented by Principal Components (PC), are linear combinations of the original features and are orthogonal to each other.

Another method that could be applied in this experiment is t-distributed stochastic neighbour embedding (t-SNE), which is a non-linear dimensionality reduction technique well-suited for embedding high-dimensional data into a lower-dimensional space for visualisation purposes, typically two or three dimensions (van der Maaten and Hinton, 2008). The algorithm minimises the divergence between two probability distributions that represent pairwise similarities of the input objects.

In summary, using techniques that minimise the dimensional representation of the data set, while maintaining the majority of the original information, will make it easier to train and test the classifiers efficiently, resulting in a more performant and accurate model.

6 ROBUSTNESS OF EXPERIMENT

This section discusses the robustness of the microphone fingerprinting model employed in this work, focusing on the reliability of the machine learning-based strategies and their susceptibility to adversarial attacks. Robustness in the context of audio fingerprinting refers to the ability of the model to perform consistently and accurately across various conditions, including noisy or distorted audio signals, and under potential adversarial scenarios.

6.1 Resistance to noise

One aspect of robustness is a model's resilience to environmental noise and signal distortions. It is necessary to evaluate the model's performance in realistic conditions, such as those where recordings may be subject to an increased amount of background noise, or where they are subjected to distortion from compression or transmission artefacts.[19] Several techniques can be employed to improve the model's robustness in this regard, for instance by incorporating noise reduction algorithms, using robust features which are less susceptible to noise, or augmenting the training data with artificially noisy samples to enhance the model's ability to generalise samples.

6.2 External threats

Another aspect of robustness is the model's resistance to adversarial attacks, where a potential attacker may attempt to manipulate the audio signal to cause the algorithm to misidentify the source microphone. Adversarial attacks on audio-based machine learning models can be carried out by adding carefully crafted perturbations to the input signal, which may be imperceptible to humans, but can cause the model to make incorrect predictions.[51] Evaluating the model's performance in the presence of adversarial examples and exploring defence mechanisms, such as adversarial training or employing robust features, can help improve the overall robustness of the microphone fingerprinting system.[52]

Furthermore, it is essential to consider scenarios where an attacker might purposely mimic a different microphone to fool the algorithm. Such attacks may involve tampering with the audio signal to imitate the unique characteristics of another microphone or using signal processing techniques to alter the spectral or temporal properties of the recording. Assessing the model's ability to identify such attempts and exploring countermeasures, such as incorporating additional features or using multiple classifiers, can contribute to the robustness and reliability of the fingerprinting system.[53]

6.3 Summary

Ultimately, the model's robustness is an essential aspect that needs to be considered for practical applicability. Ensuring resilience to environmental noise, signal distortions, and adversarial attacks contributes to the overall reliability of the model and its effectiveness in real-world scenarios.

While this experiment has shown promising results thus far, further work could focus on addressing these challenges by exploring the aforementioned advanced techniques and strategies. In doing so, not only can the performance of the current model be enhanced, but it also becomes possible to lay the groundwork for the development of more secure and reliable microphone fingerprinting systems in the future.

7 MICROPHONE FINGERPRINTING AS A SERVICE (MFAAS)

This section explores the feasibility of developing an accessible platform around this project, addressing the potential effectiveness, data requirements, and use cases, with a focus on its application in journalism for detecting and debunking fake audio recordings.

7.1 Outlook

The prevalence of audio manipulation techniques and deepfakes has raised concerns about the authenticity of audio recordings. A reliable way of verifying the origin of audio recordings is of paramount importance to many people. MFaaS could address this need by providing an online platform for audio authentication based on microphone characteristics.

7.2 Effectiveness

The effectiveness of a MFaaS platform would largely depend on the accuracy of the underlying microphone fingerprinting algorithm. Recent studies have demonstrated promising results in identifying microphones based on unique frequency response patterns and other audio features.[3] However, further research is needed to ensure the robustness of these algorithms against adversarial attacks and varying recording conditions. This project could act as a starting point.

7.3 Data requirements

Building a reliable MFaaS platform requires a diverse and representative dataset comprising recordings from a wide variety of microphones. It would entail collecting data from large numbers of consumer and professional microphones, as well as accounting for different models, manufacturing variations, and their usage history. Additionally, training data would need to be collected under various environmental conditions and recording setups to ensure the generality of the fingerprinting model.

7.4 Use cases

The primary use case of MFaaS for journalists could be the detection of fake recordings. They could leverage the service to verify the authenticity of leaked or whistleblower voice messages, as well as audio clips from interviews and any other sources. Furthermore, the service could help identify manipulated or selectively edited recordings, thereby aiding the integrity of the journalistic process.

7.5 Summary

Developing a MFaaS platform presents an opportunity to enhance the verification of audio recordings, particularly in the field of journalism. The potential platform's effectiveness is contingent upon the robustness of both the underlying microphone fingerprinting algorithm, and the quality of the training data. With continued research and development, MFaaS could become a valuable tool in combating the spread of disinformation and ensuring the trustworthiness of audio content.

8 CONCLUSIONS AND FURTHER WORK

This study has successfully explored the application of various machine learning algorithms to the task of microphone fingerprinting, with the aim of identifying the correct microphone based on its audio recordings. The analysis of results has demonstrated that while some classifiers, such as MLP and Linear Regression, achieved a 100% success rate in identifying the correct microphone, there remains a possibility of false positive results in these cases. This suggests the need for further investigation and validation to ensure the robustness and reliability of the models used.

A critical aspect to consider in future work is the refinement of the feature extraction process. As audio recordings contain a wealth of information, identifying the most salient and discriminative features that uniquely represent each microphone's characteristics is vital for enhancing the model's performance. Exploring alternative or additional features, such as those derived from the time-frequency domain or the microphone's impulse response, could lead to more accurate and reliable classification.

Moreover, expanding the dataset used in the study will contribute to the generalisability and robustness of the models. By incorporating a more diverse range of microphones, recording environments, and sound sources, the models can be better trained to recognise various nuances in the audio signal. This would not only improve the accuracy of the classification task but also make the system more resistant to potential false positives.

The potential false positive results from the MLP and Linear Regression classifiers warrant further investigation to determine the underlying causes of such high success rates. It is possible that these classifiers might have overfitted the training data, leading to artificially inflated performance. Techniques such as regularisation, cross-validation, and using dropout layers in the case of deep learning models can help mitigate the risks of overfitting and increase the model's performance and reliability.

Finally, exploring state-of-the-art deep learning models such as CNNs, RNNs, and transformer models may yield improved results in the context of microphone fingerprinting. These models have demonstrated remarkable success in various audio analysis tasks, including speech recognition, music source separation, and audio event detection.[3] They can potentially capture complex patterns and relationships within the data, leading to more accurate and reliable classification.

In conclusion, this study has demonstrated the feasibility and potential of machine learning-based approaches for microphone fingerprinting. However, further work is needed to address potential false positive results, enhance the feature extraction process, expand the dataset, and explore advanced deep learning models. Such efforts will contribute to the development of a robust and reliable system for microphone identification, with potential applications in audio forensics, security, and quality control.

9 REFERENCES

[1] Harwell D. F. 2019. Top AI researchers race to detect deepfake videos: We are outgunned. The Washington Post, Jun. 12.

[2] Lyu S. 2020. DeepFake Detection: Current Challenges and Next Steps. arXiv:2003.09234

[3] Qamhan M.A., Altaheri H., Meftah A. H., Muhammad G., and Alotaibi Y. A. 2021. Digital Audio Forensics: Microphone and Environment Classification Using Deep Learning. IEEE Access, vol. 9, pp. 62719-62733.

[4] Das A., Nikita Borisov N., and Caesar M. 2014. Fingerprinting Smart Devices Through Embedded Acoustic Components. arXiv:1403.3366

[5] Azzam V. 2023. Machine Learning Based Microphone Fingerprinting. Master Thesis. Offenburg University of Applied Sciences, GitHub code repository: https://github.com/victorazzam/mic

[6] Bosi M. and Goldberg R.E. 2003. Introduction to digital audio coding and standards. The Springer International Series in Engineering and Computer Science (SECS), vol. 721.

[7] Oliphant T. E. 2007. Python for Scientific Computing. Computing in Science & Engineering, vol. 9, no. 3, pp. 10-20.

[8] Maher R. C. 2018. Principles of forensic audio analysis. Springer. doi:10.1007/978-3-319-99453-6

[9] Joder C., Essid S., and Richard G. 2011. A Conditional Random Field Framework for Robust and Scalable Audio-to-Score Matching. IEEE Transactions on Audio, Speech, and Language Processing, vol. 19, no. 8, pp. 2385-2397, Nov.

[10] Li L., Chen Y., Shi Y., Tang Z., and Wang D. 2017. Deep Speaker Feature Learning for Text-Independent Speaker Verification. In Interspeech, pp. 1542-1546. arXiv:1705.03670

[11] Hennequin R., Khlif A., Voituret K., and Moussallam M. 2020. Spleeter: a fast and efficient music source separation tool with pre-trained models.

[12] Luo Y., Chen Z., Hershey J.R., Le Roux J., and Mesgarani N. 2016. Deep Clustering and Conventional Networks for Music Separation: Strong Together.

[13] Nicolson A. and Paliwal K. 2019. Deep learning for minimum mean-square error approaches to speech enhancement. Speech Communication, Volume 111, Pages 44-55, ISSN 0167-6393. doi: 10.1016/j.specom.2019.06.002

[14] Oord A.V.D., Dieleman S., Zen H., Simonyan K., Vinyals O., Graves A., and Kavukcuoglu K. 2016. Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499.

[15] Hannun A., Case C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A. and Ng, A.Y., 2014. DeepSpeech: Scaling up end-to-end speech recognition. arXiv preprint arXiv:1412.5567.

[16] De Man B., Reiss J.D. & Black D.A.A. 2013. A semantic approach to autonomous mixing. Journal of the Audio Engineering Society, 64(9), ISSN: 1754-9892, pp. 646-653.

[17] Grossberg S., Govindarajan K.K., Wyse L., and Cohen M.A. 2004. ARTSTREAM: a neural network model of auditory scene analysis and source segregation. Neural Networks, 17(4), 511-536. doi: 10.1016/j.neunet.2003.10.002

[18] Kraetzer C., Qian K., Schott M., Dittmann J. 2011. A context model for microphone forensics and its application in evaluations. doi: 10.1117/12.871929

[19] Ferrara P. and Beslay L. 2018. Microphone smart device fingerprinting from video recordings. Publications Office of the European Union, Luxembourg. doi: 10.2760/775442

[20] Chang S., Lee D., Park J., Lim H., Lee K., Ko K., Han Y. 2020. Neural Audio Fingerprint for High-specific Audio Retrieval based on Contrastive Learning. arXiv preprint arXiv:2010.11910

[21] Wang J., Xiao X., Wu J., Ramamurthy R., Rudzicz F., and Brudno M. 2020. Speaker Attribution with Voice Profiles by Graph-Based Semi-Supervised Learning. Proc. Interspeech 2020, 289-293, doi: 10.21437/Interspeech.2020-1950

[22] Li H., Chen K., Wang L., Liu J., Wan B., and Zhou B. 2022. Sound Source Separation Mechanisms of Different Deep Networks Explained from the Perspective of Auditory Perception. doi: 10.3390/app12020832

[23] Cobos M., Ahrens J., Kowalczyk K., and Politis A. 2022. An overview of machine learning and other data-based methods for spatial audio capture, processing, and reproduction. doi: 10.1186/s13636-022-00242-x

[24] Wang A. 2003. An Industrial-Strength Audio Search Algorithm. Proceedings of the 4th International Conference on Music Information Retrieval, ISMIR, Baltimore, USA. pp. 7-13.

[25] Ghias A., Logan J., Chamberlin D., and Smith B.C. 1995. Query by humming: musical information retrieval in an audio database. In Proceedings of the third ACM international conference on Multimedia (MULTIMEDIA '95). Association for Computing Machinery, New York, NY, USA, 231–236. doi: 10.1145/217279.215273

[26] Audible Magic Corporation, "Automatic Content Recognition (ACR) Solutions" [Online]. Available: https://www.audiblemagic.com/automatic-content-recognition-acr [Accessed: Mar. 31, 2023].

[27] Briot J.-P., G. Hadjeres G., and Pachet F. 2017. Deep Learning Techniques for Music Generation: A Survey. arXiv preprint arXiv: 1709.01620.

[28] Saeki T., Takamichi S., Nakamura T., Tanji N., and Saruwatari H. 2022. SelfRemaster: Self-Supervised Speech Restoration with Analysis-by-Synthesis Approach Using Channel Modeling. arXiv:2203.12937

[29] Dolby Laboratories. Voice Call API: Live Voice Chat Tools & Solutions | Dolby.io. [Online]. Available: https://dolby.io/products/voice-call [Accessed: Apr. 3, 2023]

[30] Moore B.C.J. 2012. An introduction to the psychology of hearing, 6th edn. Bingley: Emerald Group Publishing Ltd.

[31] Green A. 2018. "Phone microphone frequency range > 22kHz" [Online]. Available: https://stackoverflow.com/a/48565702 [Accessed: Apr. 8, 2023].

[32] Han J., Kamber M., and Pei J. 2011. Data Mining: Concepts and Techniques, 3rd ed, Morgan Kaufmann.

[33] Hastie T., Tibshirani R., and Friedman J. 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second edition. Springer.

[34] Kohavi R. 1995. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. Proceedings of the 14th International Joint Conference on Artificial Intelligence, vol. 2, pp. 1137-1143.

[35] McFee B., Raffel C., Liang D., Ellis D. P.W., McVicar M., Battenberg E., and Nieto O. 2015. librosa: Audio and Music Signal Analysis in Python. Proceedings of the 14th Python in Science Conference, pp. 18-24.

[36] Muda L., Begam M., Elamvazuthi I. 2010. Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques. arXiv:1003.4083.

[37] Yadav K. 2020. Review On Speech Signal Processing & Its Techniques. European Journal of Molecular & Clinical Medicine. ISSN 2515-8260 Volume 07, Issue 07.

[38] Davis S.B. and Mermelstein P. 1980. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Transactions on Acoustics, Speech, and Signal Processing, 28(4), pp.357-366.

[39] Ganchev T., Fakotakis N. & Kokkinakis G. 2005. Comparative evaluation of various MFCC implementations on the speaker verification task. In Proceedings of the SPECOM, 1(1), pp.191-194.

[40] Peeters, G. 2004. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. CUIDADO IST Project Report, 1(1), pp.1-25.

[41] Kraxberger F., Wurzinger A., and Schoder S. 2022. Machine-learning applied to classify flow-induced sound parameters from simulated human voice. arXiv:2207.09265

[42] Alpaydin E. 2020. Introduction to Machine Learning, 4th edn. Cambridge, MA: The MIT Press.

[43] Hastie T., Tibshirani R., & Friedman J. 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd edn. New York: Springer-Verlag.

[44] Bergstra J. and Bengio Y. 2012. Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(Feb), pp.281-305.

[45] Snoek J., Larochelle H., and Adams R.P. 2012. Practical Bayesian optimization of machine learning algorithms. In Advances in Neural Information Processing Systems, pp. 2951-2959.

[46] Powers D.M.W. 2011. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. Journal of Machine Learning Technologies, 2(1), pp. 37-63.

[47] Sokolova M. and Lapalme G. 2009. A systematic analysis of performance measures for classification tasks. Information Processing & Management, Volume 45, Issue 4, pp. 427-437. doi: 10.1016/j.ipm.2009.03.002

[48] Sammut C. and Webb G. I. 2017. Encyclopedia of Machine Learning and Data Mining. Springer.

[49] Fawcett T. 2006. An introduction to ROC analysis. Pattern Recognition Letters, 27(8), pp. 861-874.

[50] Amazon Web Services Incorporated, "Overfitting in Machine Learning Explained" [Online]. Available: https://aws.amazon.com/what-is/overfitting [Accessed: Apr. 9, 2023].

[51] Carlini N. and Wagner D. 2018. Audio Adversarial Examples: Targeted Attacks on Speech-to-Text. IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, pp. 1-7, doi: 10.1109/SPW.2018.00009.

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

[53] Lakomkin E., Zamani M. A., Weber C., Magg S., and Wermter S. 2018. On the Robustness of Speech Emotion Recognition for Human-Robot Interaction with Deep Neural Networks. IROS 2018: 854-860