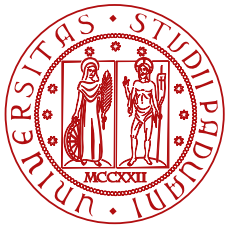

Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Master Thesis in ICT For Internet and Multimedia

Development of an Ultrasonic Platform for Privacy-Preserving Autonomous Sound Uroflowmetry at Home

Master Candidate

Mohammad Mohammadi

Student ID 2041467

Supervisor

Prof. Sergio Canazza Targon

University of Padova

Co-supervisor

Prof. Laura Arjona Aguilera

University of Deusto

Academic Year
2023/2024

To my parents
and friends

Abstract

This thesis focuses on developing an ultrasonic platform for privacy-preserving autonomous sound uroflowmetry at home. The research work is structured into three main stages, including generating a sound database, dataset preparation, and developing artificial intelligence (AI) models for flow rate extraction. My involvement in these tasks has led to acquiring knowledge in audio signal processing, flow rate analysis, water acoustics, and applying machine learning models to audio signals. This research aims to address the challenge of enabling individuals to conduct urinary flow measurements autonomously within the privacy of their homes.

Sommario

Contents

List of Figures

List of Tables

List of Algorithms

List of Code Snippets

List of Acronyms

CSV Comma Separated Values

0.1 LIST OF ACRONYM

Comma Separated Values (CSV)

DSP Digital Signal Processing

FFT Fast Fourier Transform

FR Frequency Response

SVM Support Vector Machine.

UF Uroflowmetry

SFU A single-use flowmeter

ANN Artificial Neural Network

MFCC Mel-Frequency Cepstral Coefficients

DCT Discrete Cosine Transform

HMMs Hidden Markov Models

NP NumPy

pd Panda

DFT the discrete Fourier transform

f_vector Frequency vector

sklearn scikit-learn

1

Introduction

In transitioning from conventional healthcare practices to more advanced and proactive models, it becomes crucial to implement innovative strategies. The focus lies in empowering patients to take charge of their health through self-monitoring, thereby moving away from relying solely on in-person and reactive healthcare systems. A pivotal aspect of this evolution involves shifting the use of medical-grade sensing technologies from the confines of hospitals to the comfort and convenience of patients' homes. The specific focus of this research is on addressing the challenges associated with urinary tract-related conditions, with an emphasis on proactive monitoring and early intervention. Traditional methods, such as Uroflowmetry tests, fall short when it comes to continuous monitoring outside clinical settings. These tests, often distressing and burdensome, induce voiding on demand, leading to variations in results due to situational stress factors. To overcome these limitations, recent advancements in the use of mobile devices like smartphones and smartwatches have shown promise in characterizing urinary flow patterns through a method known as audio Uroflowmetry. However, existing devices come with their own set of limitations, including the need for active user interaction, frequent battery recharging, and potential privacy concerns. In response to these challenges, our research introduces a novel approach that explores sounds beyond the range of human hearing for automatic, sound-based Uroflowmetry at home. The development includes a proof-of-concept acoustic platform based on Raspberry Pi (RPi), a small, affordable computer. This platform integrates a state-of-the-art machine learning algorithm capable of automatically detecting and recording voiding events at

home without requiring any active involvement from the user. An important aspect of this approach is the removal of human audible frequencies from the audio signals, ensuring utmost user privacy. The successful implementation of automatic detection and recording of voiding events at home, while safeguarding user privacy, is crucial for the practical application of audio Uroflowmetry and represents a significant stride towards more proactive and user-friendly healthcare solutions.[2]

1.1 Goals

This work addresses challenges in in-clinic Uroflowmetry tests, where patients are unnaturally void on demand. It also targets privacy concerns in experimental sound-based Uroflowmetry platforms using human-audible microphones. The core contribution is a Raspberry Pi-based acoustic platform with an ML classification model, automating voiding event detection with user privacy in focus. [3]

Thesis outline

Thesis outline Chapter 1: This chapter outlines the problem domain the thesis aims to address, providing a clear understanding of the issue. Chapter 2: Literature Review This chapter provides a comprehensive summary of existing works relevant to various components of the thesis solution, presenting the current state-of-the-art in the field. Chapter 3: Problem Statement and Proposal This chapter presents the tools employed and thoroughly describes the author's work. The primary problem of the thesis is defined, and research questions are proposed for resolution. Chapter 4: Evaluation In this chapter, the implemented solution undergoes testing and evaluation, offering insights into its processes, reflections, and limitations. The results of these evaluations are also presented. Chapter 5: Conclusion and Future Work This chapter discusses the achieved goals of the thesis, along with potential avenues for improvement in future endeavors. The conclusions drawn from the study are summarized, and recommendations for further research are provided.

Example of quote

> *Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.*

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

2

State of the Art

Various tools are employed in managing patients experiencing voiding dysfunction, with the voiding diary recognized as a crucial diagnostic modality for such conditions[1, 10]. The voiding diary serves as a method for physicians to objectively monitor the subjective symptoms of patients with voiding dysfunction. Given that the diagnosis and treatment decisions are based on an objective assessment of patient symptoms, the voiding diary is the starting point for studies on voiding dysfunction, making it a key diagnostic method. Patient symptoms, which may vary based on individual characteristics and environmental factors, are often subjectively expressed, leading to the well-known saying that the patient's symptoms can be an unreliable witness. Therefore, achieving an objective diagnosis is paramount, and the voiding diary stands out as a crucial diagnostic tool. Despite its significance, obtaining precise data from voiding diaries presents challenges. Recording every event and detail accurately is demanding, even for diligent and well-educated patients with voiding diary knowledge[2]. Despite its significance, obtaining precise data from voiding diaries presents challenges.[10]. This challenge is particularly pronounced for children and elderly individuals, who may only be able to provide urination times. Given the multitude of variables that can be monitored in voiding diaries, . The concept of patients carrying their voiding diaries in the form of a wearable device, such as a watch or beeper, capable of automatically a novel urination-related data emerges as a potential solution. This approach could significantly aid in studying symptoms and mechanisms that remain uncharacterized in many patients. In line with these considerations, specifically acceleration and tilt angle data,

using smart bands worn by patients this study aims to develop a technology for recognizing urination by collecting and analyzing sensed movement information, This technological advancement is anticipated to pave the way for the implementation of a urination management monitoring system accessible to end-users, contributing to the field of voiding dysfunction research[17].

2.1 Related Works

Comparison between uroflowmetry and sono uroflowmetry in the recording of urinary flow in healthy men

Voiding dysfunction is a prevalent issue that significantly affects the quality of life for many men.[11] Uroflowmetry (UF) is a commonly used non-invasive test to assess bladder emptying, typically conducted on an outpatient basis in designated procedure areas. During UF, individuals are required to urinate into a uroflowmeter, often at specific times. This process, however, is unnatural and may result in voiding with either low or very high bladder filling, leading to substantial test-to-test variability. Recommendations for repeated uroflowmetry tests add to the challenge, necessitating time-consuming and costly clinic visits. In response to these limitations, there has been a growing demand for smaller, more practical devices, leading to the development of portable uroflowmeters. Despite their potential advantages, these devices face high costs and operational difficulties, particularly for elderly patients. A novel approach to recording urinary flow patterns and measuring flow parameters is represented by Sound Uroflowmetry (SUF). SUF captures the sound generated when the urine stream hits the water level in the toilet bowl. (figure 2.2) Using a web-based algorithm, the sound file is securely stored on a website and subsequently analyzed. This approach adopts a physical principle from technologies used to estimate rainfall intensity on large bodies of water or the flow through turbines in hydroelectric plants. While there is limited literature data on the analysis of sound associated with urine flow in urology, with the last article dating back to 1991, the present pilot study aims to compare urinary flow parameters obtained using SUF with those recorded by standard uroflowmetry in controlled settings with healthy male volunteers.

A Smartwatch-Based Platform to Perform Non-Intrusive Sound-Based Uroflowmetry

The widespread adoption of smartphones has positioned them as the preferred platform for developing applications to assist individuals[12]. Despite their communication and computing capabilities, along with various sensors, the inherent limitations of smartphones as true wearables hinder continuous and seamless usage in diverse physical and social situations[13]. In contrast, smartwatches exemplify the evolution of light, compact, versatile, mobile, wearable, and intelligent devices. Sharing a similar internal architecture with smartphones, smartwatches differ in terms of speed, size, sensor count, and storage capacity, accompanied by slightly modified operating systems and software stacks. Notably, smartwatches enjoy high user acceptance and growing popularity, evidenced by the steady growth of the smartwatch market from 2018 to 2021, with projections for continued expansion in the coming years [15]. Unlike smartphones, owing to their familiarity as traditional wristwatches, are acceptable in situations where smartphones may not be, contributing to a potential reduction in the high abandonment rates associated with assistive technology[13]. This inherent advantage has spurred significant interest in utilizing smartwatches as a platform for developing innovative applications to support individuals. The ability to wear such devices continuously, coupled with their advancing communication, computing, and sensor capabilities, enables the real-time capture of behavioral data in natural environments through multiple repeated measurements a concept known as ecological momentary assessment. This method minimizes response and recall bias, distinguishing it from retrospective methods[3]. Currently, smartwatches host a diverse range of support applications, encompassing traditional ones for monitoring activity or detecting falls in elderly individuals to more novel applications such as stress detection, real-time eating activity detection, or understanding a wearer's emotional state. While some applications utilize smartwatches' microphones for detecting and classifying environmental sounds to assist the deaf or hard-of-hearing, or for cough detection, to the best of our knowledge, smartwatches have not yet been employed to implement a sound-based uroflowmeter. Such a device would be ideal, given its constant presence on the user, non-intrusiveness during urination, and social discretion[3].

2.2 Process Mining

Recommender Systems in Process Mining

There are numerous methodologies available for data processing and mining, which I will outline in the following discussion. The initial stage involves gathering databases, followed by feature extraction as the subsequent step. The subsequent phase entails constructing models based on machine learning algorithms, and ultimately, we evaluate and train the model[9]. The Artificial Neural Network (ANN) is a machine learning technique designed to model the correlation between input tuples and output units. During the learning phases, the algorithm estimates weights for each connection in the neural network, refining its performance. Key characteristics of ANN include the activation function, network topology, and training algorithms. Notably, this algorithm exhibits a high tolerance for noisy data and can classify input attributes into patterns and classes, even without prior inclusion in the training data. Furthermore, ANN can process continuous data inputs and produce continuous outputs. A typical structure of ANN is a multilayer feed-forward neural network, comprising an input layer, an output layer, and hidden layers, with interconnected nodes in each layer. The backpropagation algorithm is widely used for training ANN, iteratively learning weights for predicting the class label of samples. In this process, the algorithm compares the estimated output points with the true target values of the training set, propagating backward through each hidden layer. For classification ANN, target values may represent recognized class labels, while for numeric prediction ANN, targets could be continuous values[9]. MFCC captures elements of human vocalization and perception. It represents the logarithmic perception of intensity and tone. The process for evaluating MFCC features is sequential. Initially, an estimate of the periodogram of the frequency band is created for each setup. Subsequently, the Discrete Cosine Transform (DCT) of the filter bank strengths is computed. DCT coefficients ranging from 1 to 13 are considered for further analysis, representing the crucial information of MFCC. Each frame retains a feature vector to store the corresponding features, generating a total of 13 essential MFCC features for every audio frame. Regarding classification methods, the Random Forest technique is employed. This method is widely used for various machine learning tasks such as regression and classification. Random Forest constructs a large number of decision trees during

training, preventing overfitting to the training data. The method creates decision trees focused on approximately. The node splits based on the randomly selected feature, reaching an optimal split of the node. The tree grows to its highest point without trimming[16]. The regression forest models are employed to achieve high prediction accuracy in diagnosing abnormal voiding from uroflowmeter data. Several ensemble-based models are selected for this purpose. Three classification models are trained from classical machine learning, which includes an ensemble learning model and two random forest models[14]. Labeling systems for sound focus on predicting the content of an audio segment. Traditional machine learning methods utilize features such as Fast Fourier Transforms (FFTs) and Mel Frequency Cepstral Coefficients (MFCCs), employing conventional supervised learning algorithms for classification. For instance, Foggia et al. [7] employ a bag-of-words model with Support Vector Machines (SVMs) as a classifier. Hidden Markov Models (HMMs) have been used for context recognition, as demonstrated by Eronen et al [6]. In recent developments, deep learning has demonstrated remarkable performance in sound labeling tasks, leveraging convolutional neural networks (CNNs) for classification and eliminating the need for handcrafted features. Examples include AudioSet [8], which can label all events in a sound clip, and NELS [5], a system continuously learning sound-language relations from the web, utilizing a CNN for event detection. Sound has also been applied to scene and object recognition, exemplified by SoundNet [4]. Consequently, these models may not easily extend to utilize inaudible frequencies.

2.3 Problem Statement and Proposal

There are numerous methodologies available for data processing and mining, which I will outline in the following discussion. The initial stage involves gathering databases, followed by feature extraction as the subsequent step. The subsequent phase entails constructing models based on machine learning algorithms, and ultimately, we evaluate and train the model. The following proposal is based on the findings and implications of a study that explored the use of microphone audio data in urology telemedicine for classifying volume signals. The proposal is outlined as follows: 1. Implementation of Machine Learning Models: The potential of using binary classical machine learning tech-

niques to classify signals as normal will be explored. The proposal involves the implementation of trained classification models that will allow for real-time analysis of volume signals collected through a microphone. This would enable remote and proactive systems for continuous and non-intrusive care in urology telemedicine. 2. Development of a Diagnostic Tool: The findings indicate that the use of microphone audio data can help identify underlying pathologies associated with the urinary tract. Therefore, a proposed tool would involve the development of a diagnostic system that leverages machine learning algorithms to distinguish between healthy and unhealthy volume signals. This tool could be integrated into urology telemedicine platforms to assist healthcare providers in prompt diagnosis and monitoring of volume dysfunctions. 3. Collaboration with Healthcare Providers: The proposal emphasizes collaboration with healthcare providers to integrate the diagnostic tool into their telemedicine practices. This would involve demonstrating the accuracy and clinical relevance of the tool in identifying volume dysfunctions, thereby enhancing the toolkit available for urology telemedicine. 4. Ethical Considerations: Given the sensitive nature of urological health data, the proposal includes a focus on ethical considerations, such as patient privacy, informed consent, and data security. Collaboration with regulatory bodies and adherence to ethical guidelines would be integral to the successful implementation of the diagnostic tool.

Research questions

1. How can machine learning techniques be optimized to improve the accuracy of classifying volume signals as normal or abnormal using microphone audio data? 2. What are the key acoustic features in Microphone audio data that are indicative of volume signals, and how can they be effectively utilized for classification? 3. What are the challenges and limitations associated with using Microphone audio data for classifying volume signals, and how can these be addressed to enhance the reliability of the classification process?

2.4 Recommender Systems

We collected simulation urination recordings 58 audio files in our system using a specific microphone and a laptop. To expand the dataset, (figure 2.6). We recreated various states of urination using a syringe while maintaining the same positional conditions as the original recordings. Further details about this process will be explained in the upcoming chapter.

In this *Comparison between uroflowmetry and sono uroflowmetry*, the authors investigate the differences between traditional uroflowmetry and sono uroflowmetry techniques for recording urinary flow in men. Traditional uroflowmetry involves measuring the rate of urine flow using a flowmeter, typically in a clinical setting. On the other hand, sono uroflowmetry utilizes ultrasound technology to visualize the flow of urine in real time. The study compares the accuracy, reliability, and patient experience of both techniques, considering factors such as ease of use, discomfort, and diagnostic yield. Overall, the findings suggest that sono uroflowmetry may offer several advantages over traditional uroflowmetry, including improved visualization of urinary flow patterns and reduced patient discomfort. However, further research is needed to validate these findings and determine the optimal use of sono uroflowmetry in clinical practice., in the recording of urinary flow in healthy men method is the invasive method that exposes people to radiation[11]. Although my method is non-invasive and healthy, "The UroSound: A Smartwatch-Based Platform to Perform Non-Intrusive Sound-Based has limitations, such as depending on the smartwatch itself[3]. This could lead to issues like insufficient battery life and the need for specialized knowledge, especially in cases involving older individuals. This reliance on technology can be particularly challenging in rural or less developed areas where access to urology specialists is difficult[3]. In contrast, our method, which relies on a specific microphone and a laptop, offers a more accessible and user-friendly solution.

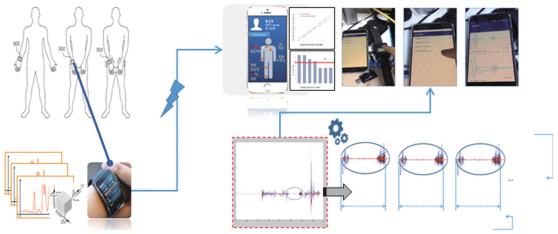

Figure 2.1: Conceptual diagram of urination recognition

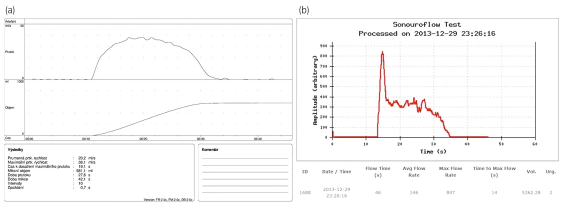

Figure 2.2: : Side-by-side comparison between the recordings of a single micturition. (a) Uroflowmetry trace. (b) The sound generated during micturition is recorded by a telephone, electronically analyzed, and transformed into a tracesonourogram.

Figure 2.3: Architecture of the platform: The UroSound App facilitates the wireless transmission of audio files to a distant web server through WiFi for signal processing and feature extraction. The securely stored audio files are managed in a MongoDB database.

Figure 2.4: User interface of the UroSound App for Wear OS: (a) Initiating the UroSound App and automatically commencing audio recording is achieved by pressing the lower-right button highlighted in green. (b) Upon tapping 'STOP RECORDING' on the screen, a dialog prompts the user to save or discard the recorded audio. (c) Commencing a new recording is facilitated by tapping 'START RECORDING' on the screen. (d) The concealed control menu, not visible to the users, is utilized for patient registration and the transmission of audio files to a remote database.

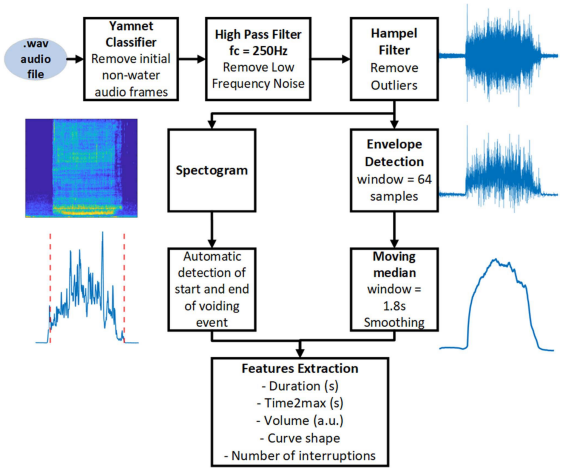

Figure 2.5: UroSound's process for estimating uroflowmetry parameters involves transforming the raw recorded audio into extracted voiding parameters and features.

Figure 2.6: Simulation of urination

3

Techniques and Tools

3.1 Tools

To record audio, I used a specific microphone which is shown in Figure 3.1. It has a USB port that connects to a laptop, and it works on a Linux system. I installed Linux on my laptop and used the Audacity software for editing because it provides several options. I selected a sampling rate frequency for audio files. The microphone has four channels, which increases the quality of the recording. After the recording from the Audacity software, I merged the audio into one file. Another tool used in this project is a syringe for simulating urination, as shown in Figure 3.4. Before beginning the experiment, I simulated a position that produces a sound similar to human urine. After several tests and recordings, I chose the best position. The discharge time for this test was approximately one-second per ml. This position is shown in the figure 3.5. I connected the microphone to the laptop and opened the Audacity software in figure Figure 3.2. I chose a sampling rate of 96Khz because the Nyquist theorem states that the sampling frequency must be at least twice the maximum frequency of the signal being sampled. Therefore, I selected a sampling frequency that is half of the maximum frequency of the microphone's range to meet the Nyquist criterion for accurate recording(figure 3.3).

3.2 Dataset Description

To classify audio events into lower 150ml or equal and greater than 150 we developed a dataset with audio signals comprised of two classes: volume ≥150ml (Class 1) and volume <150ml (Class 0) . Class 1 represents 55% of the total dataset. , while Class 0 represents the remaining 44%. . Next, we detail how we built the dataset for each class. Figure 3.5 shows the experimental setup used to collect the audio signals in a home toilet. Class 1 consists of 32 voiding sounds collected with the Ultramic last second of each voiding session removed to avoid discontinuities and Class 0 consists of volume<150 ml-voiding sound events that typically occur in a traditional home toilet. consists of 26 voiding sounds collected. we selected a range between 20 and 300 ml to record the simulation of urine volume. For example, to record at 300 ml, I pressed the record button in Audacity software, and then we started emptying the water and saved the audio. I repeated this process for 290 ml and continued reducing the volume every 10 ml until 20 ml. To optimize data acquisition, It is recommended to split the process into two sessions for larger datasets. This approach helps to ensure the quality and accuracy of the collected data. After collecting data, I removed the parts of the audio that contained silence or noise at the start and end of the recording. This was necessary as such parts would affect the quality of the result. I thoroughly checked the quality of all the audio files and created an Excel spreadsheet that contained three columns. The first column listed the names of the audio files, the second column contained the volume levels related to the audio files, and the third column contained either a 0 or a 1. I have generated an audio file containing recordings corresponding to the column names listed in the Excel file(Figure 3.6).

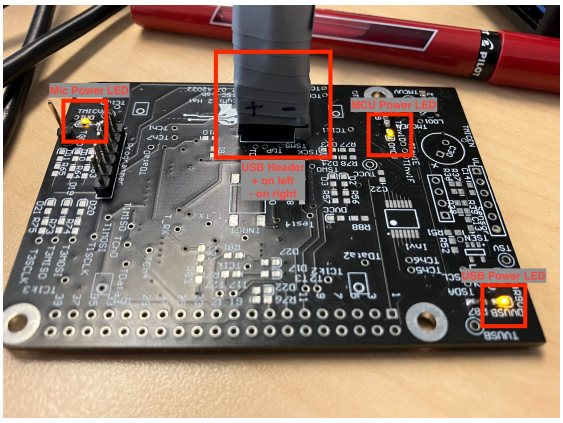

Figure 3.1: External microphone named Puffin

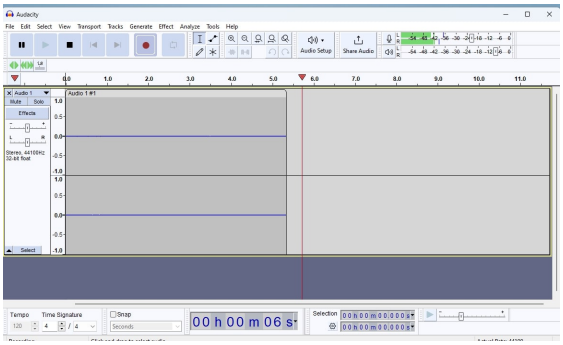

Figure 3.2: audacity software

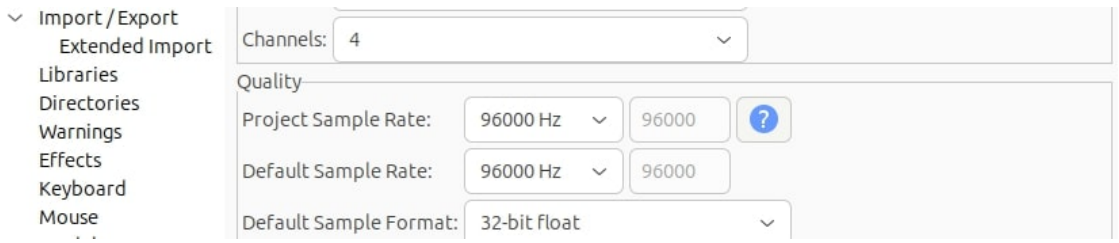

Figure 3.3: Sample rate of audio

Figure 3.4: Syringe to simulate urine discharge

Figure 3.5: A visual representation of the testing environment

11	v64	210	1
12	v65	200	1
13	v66	190	1
14	v67	180	1
15	v68	170	1
16	v69	160	1
17	v70	150	1
18	v71	140	0
19	v72	130	0
20	v73	120	0
21	v74	110	0
22	v75	100	0
23	v76	90	0
24	v77	80	0
25	v78	70	0

Figure 3.6: extraction labels from excel file

3.3 Tools

4

Results

4.1 Preprocessing

In this section, we delve into developing a supervised machine-learning model to identify voiding events from audio clips containing frequencies outside the range of human hearing. The objective is to classify incoming audio clips in real-time into two categories: those with a volume equal to or greater than 150 and those with a volume lower than 150ml. Our model must fulfill several key criteria: perform inference in real-time, operate on low-power devices with limited computational resources, and minimize false negatives to ensure voiding events are not overlooked. Given the size of our dataset, we explore traditional machine-learning models for this classification task. To determine the optimal model meeting our requirements, we train three different machine learning models: a Support Vector Machine (SVM), a Random Forest (RF) with 1000 trees, and Xgboost. The initial stage involves generating features to describe each audio sample. We utilize the SciPy Python library to extract a linear binned Fast Fourier Transform (FFT) for each audio clip. audio signals were captured using 96kHz.We set the lower frequency threshold for FFT computation to 8kHz and 16kHz to eliminate human audible frequencies. Determining the upper limit involves assessing the significance of various spectral bands using feature selection techniques, which rank each band based on its information content.

4.2 Libraries

I have utilized several key libraries in my thesis project to facilitate data analysis, visualization, and machine-learning tasks. Here's a brief explanation of each library and its role in my project: NumPy (np): NumPy is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. I have used NumPy extensively for data manipulation, array operations, and numerical computations in my project. Pandas (pd): Pandas is a powerful data manipulation and analysis library built on top of NumPy. It offers data structures like DataFrame and Series, which allow for easy handling and manipulation of structured data. I have utilized Pandas for data preprocessing, cleaning, and organizing tabular data in my thesis project. Matplotlib (plt): Matplotlib is a widely-used plotting library in Python, known for its flexibility and customization options. It provides functions for creating various types of plots, including line plots, scatter plots, histograms, and more. I have used Matplotlib to visualize data distributions, trends, and relationships between variables in my thesis project. Seaborn (sns): Seaborn is a statistical data visualization library that works closely with Pandas data structures. It provides high-level functions for creating informative and attractive statistical graphics. Seaborn offers seamless integration with Pandas dataframes and can generate complex visualizations with minimal code. I have leveraged Seaborn to create insightful visualizations, such as distribution plots, pair plots, and heatmaps, to explore and analyze relationships within my dataset. scikit-learn (sklearn): Scikit-learn is a comprehensive machine learning library that provides efficient implementations of various supervised and unsupervised learning algorithms. It offers tools for data preprocessing, model selection, evaluation, and performance metrics. In my thesis project, I have utilized scikit-learn's Support Vector Machine (SVM) implementation (SVC) for classification tasks and its metrics module to evaluate model performance, particularly accuracy scores. SciPy (scipy): SciPy is an open-source library that builds on NumPy and provides additional functionality for scientific computing. One of its submodules, scipy.fft, offers efficient algorithms for computing the discrete Fourier transform (DFT) and its inverse. I have used the fft function from Scipy.fft module to perform Fourier transforms for analyzing signals and extracting frequency-domain information in my thesis project. Os module: The

line import Os imports the Os module, which provides a way to interact with the operating system, such as navigating file systems, manipulating paths, and executing commands. Overall, these libraries have played crucial roles in enabling data analysis, visualization, and machine learning tasks throughout my thesis project, providing powerful tools and functionalities to explore, understand, and model complex datasets effectively.

4.3 Labeling

I extracted the Diagnosis column from the Data Frame using df and assigned It to the variable y. This allows for the isolation and analysis of the Diagnosis data separately from the rest of the dataset. The extracted column can then be printed to the console or used for further analysis and processing as needed in the project. I employed the `os` module in Python to navigate and interact with the file system. The code snippet demonstrates how the `os.listdir()` function can retrieve a list of filenames within a specified directory. By providing the path of the directory containing the audio files through the `directory` variable, I obtained a list of filenames stored within that directory. Subsequently, a for loop iterated over each filename in the list (`file_list`), printing each filename to the console. This approach facilitated the exploration and identification of audio files present within the specified directory, laying the groundwork for further processing and analysis of the audio data in subsequent stages of the project.

4.4 FFT Feature Extraction

In this section of my thesis project, I implemented a feature extraction process for audio data using the Librosa library in Python.

The `features_extractor_FFT` function is defined to extract FFT features from the audio signal. This function takes the following inputs:

- `f_vector`: Frequency vector
- `fft_us`: FFT values
- `nbins`: Number of bins

It returns the FFT features and frequency bins.

```
def features_extractor_FFT(f_vector, fft_us, nbins):
    # Function implementation goes here
    pass
```

Computing FFT features: For each audio file, the code computes FFT (Fast Fourier Transform) features using the `librosa. stft` function and extracts FFT frequencies using `librosa.fft_frequencies`. After extracting features from all audio files, the list of features (`X`) is converted to a Numpy array and then normalized using `StandardScaler` to ensure uniform scaling of feature values. Finally, the shape of the normalized features (`X_normalized`) is printed to verify the dimensions of the feature matrix.

4.5 MFCC FEATURE EXTRACTION

the process of extracting MFCC features from audio files and storing them for further analysis in the thesis project. It showcases the implementation of feature extraction techniques crucial for analyzing and modeling audio data effectively. Additionally, error handling is incorporated to manage exceptions that may arise during the feature extraction process, ensuring robustness in processing audio files.

4.6 NORMALIZATION FFT AND MFCC

After extracting MFCC and FFT features from audio files, we perform normalization to ensure that the extracted features are standardized, possessing zero mean and unit variance. This normalization process is essential for preparing the MFCC and FFT features for input into machine learning models. Standardization ensures that each feature contributes equally to the analysis and prevents features with larger scales from dominating the model training process. By scaling the features to have a mean of zero and a standard deviation of one, we create a consistent and uniform scale for all features, facilitating more stable and efficient model training. In essence, normalization enhances the interpretability and performance of machine learning models by ensuring that the features are appropriately scaled and centered, enabling the models to make more accurate and reliable predictions based on the input data

4.7 Splitting Data

to split a dataset into separate training and testing sets, which is a fundamental step in machine learning model development. By splitting the data, we can train our model on one subset (training set) and evaluate its performance on another subset (testing set) to assess its generalization ability. It's essential to have a separate testing set to avoid overfitting, where the model performs well on the training data but fails to generalize to unseen data.

4.8 Evaluation accuracy of model

first, we use MFCC feature extraction and create a model with SVM, Random Forest, and Xgboost. the results of these models are in Table 4.1. In the second step, we use FFt feature extraction and create a model with SVM, Random Forest, and Xgboost. (Table 4.2) In the third step, we set the minimum frequency (`fmin`) to 8kHz and proceed to build models using Support Vector Machine(`SVM`), Random Forest, and XGBoost algorithms(Table 4.3). In the subsequent step, we iterate through the process again, this time setting the minimum frequency (`fmin`) to 16 kHz and constructing models accordingly. Following the augmentation process, an Excel file is generated to store augmented audio filenames alongside their corresponding labels. Subsequently, the augmented audio files are combined with the original audio files. FFT features are extracted from both augmented and original audio samples. A machine learning model is then constructed using various algorithms, taking into account the Fmin values of 8 and 16 for feature extraction.

algorithms	**Train accuracy**	**Test accuracy**
Random Forest	0.8695	0.75
SVM	0.8478	0.75
Xgboost	0.5869	0.4166

Table 4.1: Comparing Random Forest, Support Vector Machine, and XGBoost algorithms through the lens of MFCC extraction

algorithms	Train accuracy	Test accuracy
Random Forest	0.8695	0.75
SVM	0.8695	0.75
Xgboost	0.5869	0.4166

Table 4.2: Comparing Random Forest, Support Vector Machine, and XGBoost algorithms through the lens of FFT extraction

algorithms	Train accuracy	Test accuracy
Random Forest	0.8333	0.9782
SVM	0.4166	0.86
Xgboost	0.5869	0.4166

Table 4.3: Comparing Random Forest, Support Vector Machine, and XGBoost algorithms using FFT extraction with a minimum frequency of 8 kHz

4.9 AUGMENTING AUDIO

we investigate a different method by evaluating the model's accuracy using augmented audio data. The provided code offers a comprehensive framework for augmenting audio files through diverse techniques, presenting a valuable resource for enriching the diversity and quality of audio datasets crucial for machine learning applications like speech recognition, sound classification, and audio generation. At its core, the `augment_audio` function efficiently processes audio files sourced from a designated directory, expertly applying a suite of augmentation methods defined within the code(Code 4.1). These techniques, ranging from adding white noise to inverting polarity, cater to various augmentation needs, ensuring versatility in dataset enhancement. The output of this process is a collection of augmented audio files stored in an output directory, ready to bolster the training and evaluation of machine learning models. By embracing this approach, researchers and practitioners can harness the power of augmented audio data to foster more robust and accurate machine-learning models across a spectrum of audio-related tasks(Table 4.4). Finally, I use GridSearchCV through random forest algorithms(Table 4.5).

```
import os
import librosa
import numpy as np
import soundfile as sf
import random
# Specify the directory
```

```
directory = 'audio_new'
output_directory = 'augmented-audio'

# Create the output directory if it doesn't exist
os.makedirs(output_directory, exist_ok=True)

# Function to add white noise to audio signal
def add_white_noise(signal, noise_percentage_factor):
    noise = np.random.normal(0, signal.std(), signal.size)
    augmented_signal = signal + noise * noise_percentage_factor
    return augmented_signal

# Function to perform time stretching on audio signal
def time_stretch(signal, sr, time_stretch_rate):
    return librosa.effects.time_stretch(signal, rate=
    time_stretch_rate)

# Function to perform pitch scaling on audio signal
def pitch_scale(signal, sr, num_semitones):
    return librosa.effects.pitch_shift(signal, sr, num_semitones)

# Function to apply random gain (amplification or attenuation) to
    audio signal
def random_gain(signal, min_factor=0.1, max_factor=0.12):
    gain_rate = random.uniform(min_factor, max_factor)
    augmented_signal = signal * gain_rate
    return augmented_signal

# Function to invert polarity of audio signal
def invert_polarity(signal):
    return signal * -1

# Function to augment audio files
def augment_audio(audio_file_path, output_directory):
    signal, sr = librosa.load(audio_file_path, sr=96000)

    # Augment the audio (example: inverting polarity)
    augmented_signal = invert_polarity(signal)

    # Save the augmented audio
    output_path = os.path.join(output_directory, os.path.basename(
    audio_file_path))
    sf.write(output_path, augmented_signal, sr)
```

```
47
48 # Iterate through audio files in the directory and augment them
49 try:
50     for root, dirs, files in os.walk(directory):
51         for audio_file in files:
52             if audio_file.endswith('.wav'):
53                 # Construct the full file path
54                 audio_file_path = os.path.join(root, audio_file)
55
56                 # Augment the audio and save the augmented files
57                 augment_audio(audio_file_path, output_directory)
58 except Exception as e:
59     print(f"An error occurred: {e}")
```

Code 4.1: Python code for augmenting audio files

algorithms	Train accuracy	Test accuracy
Random Forest	0.8695	0.75
SVM	0.8260	0.75
Xgboost	0.5869	0.4166

Table 4.4: Comparing Random Forest, Support Vector Machine, and XGBoost algorithms using FFT extraction with a minimum frequency of 8 kHz covering the augmented audio dataset comprehensively

Algorithm	Train accuracy	Test accuracy	Validation
GridSearchCV	98.91	95.83	95.83

Table 4.5: The Random Forest Classifier utilizes FFT feature extraction along with GridSearchCV for hyperparameter tuning, covering the augmented audio dataset comprehensively.

After utilizing the mentioned algorithms and augmented data, I proceeded to adjust several parameters in the feature extraction process using FFT and then reconstructed the model(Table 4.6). You can review the updated code in the following location code 4.2.

```
1 import os
2 import librosa
3 import numpy as np
4 from sklearn.preprocessing import StandardScaler
5
6 # Specify the directory
```

```
directory = 'audio-new'

# Define the FFT feature extraction function
def features_extractor_fft_lin_bin(f_vector, fft_us, nbins):
    # (1) Log-space array from fmin to fmax, to get nbins values
    fmin = 0  # Set fmin to 0 kHz
    fv = np.linspace(fmin, f_vector[len(f_vector) - 1], num=nbins)

    # (2) Map the previous values to the freq. values obtained from the fft
    f_vector_bin = []
    for val in fv:
        nf = f_vector[((f_vector > val))]
        if nf.size > 0:
            valf = nf[0]
            f_vector_bin.append(valf)
    f_vector_bin.append(f_vector[len(f_vector) - 1])

    # (3) Get index of the f_values
    fft_vector_bin = []
    for index in range(len(f_vector_bin) - 1):
        ii = np.where((f_vector >= f_vector_bin[index]) & (f_vector < f_vector_bin[index + 1]))
        fft_values = fft_us[ii]
        feat = np.sum(fft_values)
        fft_vector_bin.append(feat)

    # (4) Get in the values of the fft in those indices. Sum values within each bin
    return fft_vector_bin, fv

# Lists to store features
X = []

# Iterate through audio files in the directory
for audio_file in os.listdir(directory):
    if audio_file.endswith('.wav'):
        # Construct the full file path
        audio_file_path = os.path.join(directory, audio_file)

        # Load the audio file
        audio_signal, sr = librosa.load(audio_file_path, sr=96000)

```

```
        # Choose the number of FFT points
        n_fft = 2048

        f_vector, fft_us = librosa.fft_frequencies(sr=sr, n_fft=n_fft), np.abs(librosa.stft(audio_signal, n_fft=n_fft))

        # Choose the number of bins
        nbins = 1000

        # Now you can use these values in your feature extraction function
        fft_feats, freq_bins = features_extractor_fft_lin_bin(f_vector, fft_us, nbins)

        # Append features to the list
        X.append(fft_feats)

        # Print the FFT features
        #print("File:", audio_file_path)
        #print("FFT Features:", fft_feats)
        #print("=" * 30)

# Convert the list of features to a numpy array
X = np.array(X)

# Normalize the features
scaler = StandardScaler()
X_normalized = scaler.fit_transform(X)

# Now X_normalized contains your normalized feature values
print("X_normalized Shape:", X_normalized.shape)
```

Code 4.2: FFT FUTURE EXTRACTION

Ultimately, the SVM model was selected as the optimal choice, demonstrating a test accuracy of 0.833 and a train accuracy of 0.8478.(code 4.3). Subsequently, for evaluating the model, the F1 score and recall were computed. The results are presented in Table 4.6.

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
import numpy as np
```

```

# Assuming X is a list of 2D arrays (MFCC features for each audio
    file)
# Each element of X is a 2D array

# Encode labels
label_encoder = LabelEncoder()
y_encoded = label_encoder.fit_transform(y)

# Split the data into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X_flat, y_encoded
    , test_size=0.2, random_state=96000)

# Build and train a Support Vector Machine (SVM) classifier with
    class weights
svm_classifier = SVC(C=2, class_weight='balanced', random_state
    =96000)
svm_classifier.fit(X_train, y_train)

# Predict on the test set
y_test_pred = svm_classifier.predict(X_test)

# Calculate accuracies
test_accuracy = accuracy_score(y_test, y_test_pred)

print("Test Accuracy:", test_accuracy)
```

Code 4.3: Support Vector Machine (SVM) Classifier

Algorithm	Train accuracy	Test accuracy	F1 Score	Recall
SVM	0.8478	0.8333	0.8	0.8

Table 4.6: SVM algorithms along fft feature extraction and evaluate with f1 score and recall.

5

conclusion and future of work

In conclusion, the project aims to develop a sophisticated model capable of not only detecting the sound of urination but also distinguishing it from other ambient sounds present in the environment. The future work involves leveraging advanced machine learning algorithms and signal processing techniques to train the model to automatically initiate the recording process upon detecting the specific acoustic signature associated with urination. This automated recording functionality will eliminate the need for manual intervention and enhance the overall quality of the collected data by ensuring that all relevant urination events are captured accurately and consistently. Additionally, the automated recording capability will contribute to safeguarding the privacy of individuals participating in the data collection process by minimizing the risk of inadvertently capturing unrelated sounds that could compromise privacy. The future work of this project aims to develop a sophisticated model capable of not only detecting the sound of urination but also distinguishing it from other ambient sounds present in the environment. By leveraging advanced machine learning algorithms and signal processing techniques, the model will be trained to automatically initiate the recording process upon detecting the specific acoustic signature associated with urination. This automated recording functionality will eliminate the need for manual intervention, such as pressing a recording button in software like Audacity. By implementing this automated recording feature, several significant benefits can be realized. Firstly, it will enhance the overall quality of the collected data by ensuring that all relevant urination events are captured accurately and consistently. This improvement in data quality is

essential for conducting robust analyses and deriving meaningful insights from the recorded audio samples. Furthermore, the automated recording capability will contribute to safeguarding the privacy of individuals participating in the data collection process. Since the model will autonomously trigger recording only during urination events, it minimizes the risk of inadvertently capturing unrelated sounds that could compromise privacy. This aspect is particularly important in healthcare settings where patient confidentiality and privacy are paramount. Additionally, the automation of the recording process will lead to a significant reduction in the time required to gather data. By eliminating the manual task of initiating recordings, researchers can streamline the data collection workflow and increase overall efficiency. This time-saving aspect is particularly beneficial in large-scale studies or clinical trials where substantial volumes of audio data need to be collected and analyzed. In summary, the extension of this project's future work to include the development of an automated recording model represents a significant advancement in urinary flow pattern analysis. By combining advanced technology with sound scientific principles, this innovative approach holds the potential to revolutionize the field of Uroflowmetry, facilitating more accurate diagnoses and personalized treatments for individuals with urinary tract conditions.

I, Mohammad Mohammadi, the author of this thesis, would like to express my sincere gratitude and appreciation to all those who have significantly contributed to its formation and development. **Laura Arjona Aguilera**, Associate Professor at the University of Deusto's Faculty of Engineering, has been my thesis supervisor and a huge help. She often offers advice and pointers on how to improve my work.

Sergio Canazza Targon, associate professor of Information Engineering at UNIPD, was an indispensable guide towards graduation. He was always available, responded promptly to my inquiries, and adeptly directed me toward optimal pathways.

Marcos Lazaro Alvarez Arteaga, Ph.D. student, Computer Science for Sustainable Information and Development Society, for his assistance with several aspects of my thesis and for being a friend and supporter.