People`s Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of Echahid Hamma Lakhdar - El Oued
Faculty of Technology
Department of Electrical Engineering

Dissertation

ACADEMIC MASTER

Domain: Science and Technology

Division: Electrical Engineering

Specialty: Electrical Control

Presented by:

1. Messak Ikbal
2. Othmani Khaled
3. Touil Oussama

Entitled:

Design of a Smart Device for a Deaf Drivers

Dissertation Submitted in Partial Fulfillment of the Requirements for the Master Degree in ……...

Publicly defended in: ……/…… /2025

Board of Examiners:

Dr. Dr. Allag Nassiba.	Echahid Hamma Lakhdar University.	**Chairman**
Dr. Cherif Hakima	Echahid Hamma Lakhdar University.	**Supervisor**
Dr. Maamir Madiha	Echahid Hamma Lakhdar University	**Examiner**

Academic Year: 2024/2025

Abstract:

Deaf and hard-of-hearing drivers face significant challenges in recognizing auditory signals on the road, which are essential for safe and responsive driving. This project proposes an intelligent assistive system that captures environmental sounds and classifies them in real time using high-quality microphones and sound cards connected to a Raspberry Pi 4. A machine learning model, built using a Convolutional Neural Network (CNN), is trained to distinguish between critical warning sounds (such as sirens and horns) and irrelevant noise. When a warning sound is detected, the system alerts the driver through a visual LCD screen and a vibration motor. This solution enhances driving safety, independence, and situational awareness for the hearing-impaired community.

الملخص:

يواجه السائقون الصم وضعاف السمع تحديات كبيرة في التعرف على الإشارات السمعية على الطريق، والتي تعتبر ضرورية لضمان القيادة الآمنة والمتجاوبة. يقترح هذا المشروع نظاماً مساعداً ذكياً يلتقط الأصوات البيئية ويصنفها في الوقت الحقيقي باستخدام ميكروفونات عالية الجودة وبطاقات صوتية متصلة بجهاز Raspberry Pi 4. يتم تدريب نموذج للتعلم الآلي تم إنشاؤه باستخدام شبكة CNN للتمييز بين أصوات التحذير الحرجة (مثل صفارات الإنذار والأبواق) والضوضاء غير ذات الصلة. عند اكتشاف صوت تحذير، يقوم النظام بتنبيه السائق من خلال شاشة LCD مرئية ومحرك اهتزاز. يعمل هذا الحل على تعزيز سلامة القيادة والاستقلالية والوعي الظرفي لمجتمع ضعاف السمع.

إهـــــداء

الحمد لله الذي علّمني ما لم أكن أعلم، وأرشدني حين ضللت، وثبّتني حين ضعفت، ورفعني بلطفه حين ظننت أنني لن أقدر.

الحمد لله أولاً وآخراً، ظاهرًا وباطنًا، سرًا وعلانية... فمنه البداية، وإليه المنتهى.

أما بعد...

فإلى روحي التي تعبت، وصبرت، وأصرّت، وسهرت الليالي تنسج حلمها رغم كل ما واجهته...

إلى نفسي التي تعلّمت ألا تستسلم، أن تنهض في كل مرة، أن تؤمن بأن الحلم لا يخذل من يُخلص له...

أُهديك هذا الإنجاز يا أنا، فأنتِ تستحقينه أكثر مما يظن الجميع.

ثم...

إلى أبي الحبيب يا من كان ظلي في الشمس، ونوري في الظلام. يا من كنت أستمد منك الثبات في لحظات الضعف، والصبر في أوقات الضيق.

أهديك هذا النجاح، فكم من مرة أخفيت تعبك عني لتُشعرني بالأمان. إن كل لحظة وصلت فيها، كانت بدعائك، وتربيتك، ورضاك.

إلى أمي الحنونة يا من خُلقت من الحنان كله، يا من لم تنَم عيناكِ يومًا إلا وقد سبقتني دعواتكِ إلى السماء.

صوتك، لمستك، قلبك... كانوا لي وطنًا حين تاهت بي الأيام. نجاحي هو باقة من ورود الحب أضعها عند قدميك، فأنتِ كل الحكاية.

إلى أخواتي العزيزات كنتنّ لي سُترة حبٍّ، ورفقة نقية، وملاذًا دافئًا في كل لحظة.

ضحكاتكنّ كانت سلوى أيامي، واهتمامكنّ نَسج غيمة سلام فوق رأسي... فشكرًا لكنَّ من القلب.

وإلى أطفالهنّ الأحبة كنتم مصدر البهجة في عزّ التعب، نوافذ نور في أيامي، وتذكير دائم بجمال الحياة رغم كل المسؤوليات.

إلى إخوتي الأعزاء كنتم لي سندًا حقيقيًا، جدارًا قويًا أتكئ عليه في الأوقات الصعبة، وجمرة حماس أشعلت في قلبي روح المثابرة. جزيل الشكر والحب لكم، ولـأطفالكم الأعزاء، الذين زيّنوا لحظاتي بابتساماتهم وضحكاتهم.

إلى أستاذتي المشرفة الفاضلة كنتِ النور الذي أضاء طريقي في مشروع التخرج، والصوت الهادئ الذي دفعني للمزيد حين تعبت.

شكرًا من القلب على صبرك، على احتوائك، وعلى أنك آمنت بقدرتي ووجهتني بحب.

إلى أستاذاتي طوال أعوام الدراسة لكنَّ في قلبي امتنان لا يُقاس، وتقدير لا يُنسى. علمتنني أكثر من المناهج... علمتنني الصبر، والاحترام، والإصرار.

إلى أصدقائي الأعزاء أنتم رفاق الدرب، وشركاء التعب، ومن جعلوا الدراسة أوقاتًا لا تُنسى.

كنتم لي دعمًا، ودفئًا، وصوتًا يخفف عني كل شيء. لكم مني كل الحب. إلى كل من أهداني كلمة طيبة

ربّ كلمة كانت نورًا، ربّ نظرة كانت عزاءً، ربّ ابتسامة كانت طوق نجاة... شكرًا لأنكم كنتم هناك، حتى دون أن تدروا.

وفي الختام... هذا الإنجاز ليس نهاية، بل بداية لحياةٍ اخترتُ أن أعيشها بالإيمان، والعلم، والحب. من خريج قسم الهندسة الكهربائية – دفعة 2025 – بكل فخر...

- مســـاك إقبـــال

إهـــــداء

الحمد لله أولًا وآخرًا، ظاهرًا وباطنًا،

الحمد لله الذي بنعمته تتم الصالحات، وبفضله تحققت الأمنيات، وبه تُنال الغايات.

إلى من كانا بعد الله سرّ هذا الإنجاز،

إلى والديّ العزيزين، من علّماني أن الطموح لا سقف له، وأن الإيمان بالهدف هو أول خطوة في طريق النجاح،

إلى إخوتي وأحبتي، من شاركوني لحظات التعب والفرح، وكانوا العون والرفقة الصادقة.

إلى أولئك الذين كانوا الضوء في عتمة دربي، واليد التي ساندتني حين تعثرت...

إلى أساتذتي الأفاضل،

إلى من علّمني حرفًا ففتح لي أفقًا،

إلى من آمن بقدراتي ووجّهني بصبر وصدق،

إلى كل معلم ومعلمة غرسوا في قلبي حبّ العلم،

وإلى أستاذي المشرف، الذي كان لنصائحه وتوجيهه الأثر الكبير في إخراج هذا العمل إلى النور،

كل التقدير والعرفان.

إلى كل من آمن بي، ورافقني في رحلتي،

أهدي هذه المذكرة... عربون شكر، وامتنان لا تفيه الكلمات.

- عثماني خالد

إهـــــداء

إلى أولئك الذين كانوا النور الذي أضاء طريقي، والسند الذي شدّ من أزري في كل مراحل هذا المشوار...

إلى والديّ العزيزين، رمز التضحية والعطاء، الذين زرعوا في نفسي قيمة العلم، وغرسوا في قلبي الصبر والإصرار.

ما كان لهذا العمل أن يرى النور لولا دعاؤكما الصادق وتضحياتكما العظيمة.

إلى إخوتي وأخواتي، رفاق الدرب والدعم الهادئ، الذين كانوا دومًا مصدر طمأنينة وفرح في لحظات التعب والإنهاك.

إلى أساتذتي الأفاضل، منارة العلم وبناة الفكر، الذين لم يبخلوا بعلمهم وتوجيههم، فكان لهم بالغ الأثر في تشكيل هذا العمل.

إلى أصدقائي الذين تقاسموا معي التعب والجهد واللحظات الصعبة، وشاركوا الفرحة بصدق، فكنتم جزءًا من هذا الإنجاز.

إلى كل من دعمني بكلمة، أو دعا لي بخير، أو كان له في القلب أثر جميل...

أُهدي هذا العمل المتواضع، عربون وفاء وامتنان، وتقديرًا لكل لحظة صدق ومحبة مرّت بي في هذا الطريق.

- طويــل أسـامة

Thanks, and Appreciation

In the name of Allah, the Most Gracious, the Most Merciful

All praise is due to Allah, by whose grace good deeds are completed, and who granted us the strength and guidance to accomplish this modest work.

We, the authors of this dissertation, would like to express our deepest gratitude and heartfelt appreciation to our esteemed supervisor, **Dr. Cherif Hakima**, for her continuous support, valuable guidance, and constructive feedback throughout every stage of this research. Her dedication, patience, and commitment to academic excellence have been a true source of inspiration. We are truly honored to have worked under her supervision and are grateful for the knowledge and insight she generously shared with us.

We would also like to extend our sincere thanks and appreciation to our respected professors: **Mr. Ghanabzia Ahmed**, **Mr. Sassi Zouheir**, and **Mr. Hamza Adaïka**, for their scientific support, valuable advice, and insightful remarks that greatly contributed to the success of this work. Their guidance and encouragement were essential throughout the research process.

Our heartfelt thanks also go to the honorable members of the **defense committee**, who honored us by accepting to evaluate our work, devoting their time and effort, and providing us with critical feedback that will undoubtedly enrich our research and guide us in future endeavors.

We further extend our gratitude to all the professors and staff members of the **Department of Electrical Engineering at the University of Echahid Hamma Lakhdar – El Oued**, whose teaching, encouragement, and support have significantly shaped our academic journey and contributed to the successful completion of this dissertation.

To our beloved families, we owe profound gratitude. Their unwavering support, heartfelt prayers, and constant belief in us gave us strength through every challenge. Their encouragement was a constant source of motivation, and this achievement would not have been possible without them.

We are also grateful to our colleagues and friends for their cooperation, meaningful discussions, and moral support throughout our studies. Their presence created a positive and collaborative environment that made this journey more enriching.

We pray that this work will be a useful contribution to our field of study and serve as a foundation for further academic and professional growth. Indeed, Allah is the bestower of success.

TABLE OF CONTENTS

LIST OF FIGURES I

LIST OF TABLES III

LIST OF ABBREVIATIONS IV

GENERAL INTRODUCTION 1

CHAPTER I: LITERATURE REVIEW

I.1 INTRODUCTION 3

I.2 OVERVIEW OF PREVIOUS SYSTEMS AND STUDIES 3

I.3 INTRODUCTION TO AUDIO SIGNALS 6

I.3.1 HUMAN AND VEHICULAR SOUND SOURCES 7

I.3.2 ANALOG-TO-DIGITAL CONVERSION (ADC) 7

I.4 CONCLUSION 10

CHAPTER II: FREQUENCY-DOMAIN ANALYSIS FOR SOUND CLASSIFICATION

II.1 INTRODUCTION 11

II.2 FREQUENCY-DOMAIN ANALYSIS METHODS FOR VEHICLE HORN SOUND CLASSIFICATION 11

II.2.1 FAST FOURIER TRANSFORM (FFT) 11

II.2.2 SHORT-TIME FOURIER TRANSFORM (STFT) 12

II.2.3 SPECTROGRAM 12

II.2.4 MEL-FREQUENCY CEPSTRAL COEFFICIENTS (MFCCS) 14

II.2.5 MEL SPECTROGRAM 15

II.2.6 WAVELET TRANSFORM 15

II.3 ARTIFICIAL INTELLIGENCE AI 16

TABLE OF CONTENTS

Π.3.1 DIFINTION 16
Π.3.2 MACHINE LEARNING (ML) 17
Π.3.3 DEEP LEARNING DL 18
Π.3.4 THE RELATIONSHIP BETWEEN ARTIFICIAL INTELLIGENCE, MACHINE LEARNING AND DEEP LEARNING 19
Π.3.5 THE DIFFERENCE BETWEEN DEEP LEARNING AND MACHINE LEARNING 20
Π.4 CONCLUSION 21
CHAPTER III: DESIGN OF A SMART DEVICE FOR ASSISTING DEAF DRIVERS
III.1 INTRODUCTION 23
III.2 GENERAL ARCHITECTURE OF THE PROPOSED SYSTEM 23
III.3 SYSTEM COMPONENTS 24
III.4 PROGRAMMING ENVIRONMENT AND REQUIRED LIBRARIES 26
III.5 STAGES OF THE PROJECT 27
III.5.1 SOUND COLLECTION PHASE 27
III.5.2 FEATURE EXTRACTION STAGE 28
III.5.3 MODEL TRAINING AND TRANSFORMATION PHASE 31
III.5.4 TESTING MODEL 34
III.6 SYSTEM SETTINGS OF RASPBERRY PI 36
III.7 CONCLUSION 38
GENERAL CONCLUSION 39

LIST OF FIGURES

CHAPTE I: LITERATURE REVIEW

Figure I.1: A continuous analog signal converted into a digital signal through sampling ..8

Figure I.2: An example of how aliasing occurs when the sampling rate is insufficient 8

CHAPTER II: FREQUENCY-DOMAIN ANALYSIS FOR SOUND CLASSIFICATION.

Figure Π.1: A segment of time-domain data projected onto the frequency domain 11

Figure Π.2: This image illustrates the STFT process.. 12

Figure Π.3: Detection of a Strong Seismic Signal in the 4–8 Hz Band at Station EDM EHZ (UW).. 13

Figure Π.4: Mel Spectrogram.....15

Figure Π.5: AI Work.. 17

Figure Π.6: ML Work.. 18

Figure Π.7: AI, ML and DL.. 19

Figure Π.8: The CNN Components.....21

CHAPTER III: DESIGN OF A SMART DEVICE FOR ASSISTING DEAF DRIVERS

Figure III.1: A Smart Assistive System for Individuals with Hearing Impairments 24

Figure III.2: Collect Wav sounds 28

Figure III.3: : The shape of the two files obtained after the categorization process.. 28

Figure III.4: Extraction code for mel spectrogram... 30

Figure III.5: The result for mel spectrogram.....31

Figure III.6: Download the necessary libraries for Model.....31

Figure III.7: Code training.. 32

Figure III.8: Result obtained 32

Figure III.9: The curves represented by the model's accuracy and loss ratios 33

Figure III.10: Full code 35

Figure III.11: Result Véhicule Détecte.. 35

Figure III.12: Result No Véhicule Détecte.. 35
Figure III.13: Install Raspberry pi operating system... 38
Figure III.14:. Install Raspberry pi operating system.. 36
Figure III.15:. Connect a card a micro SD.. 36
Figure III.16:. Connect Raspberry Pi to the screen.. 37
Figure III.17: Image of the entire project.....37

LIST OF TABLES

CHAPTER I: LITERATURE REVIEW

Table I.1: Bit Length and their number of levels and step size for a 5V reference range ...9

CHAPTER III: DESIGN OF A SMART DEVICE FOR ASSISTING DEAF DRIVERS

Table III.1: Component of the smart Device for Deaf Drivers.. 24

List of abbreviations

AI : Artificial Intelligence**.**

AAC: Augmentative and Alternative Communication**.**

ACVA: Assistive Communicative Vibrating Aid.

HL: Hearing Loss.

DHH: Deaf or Hard of Hearing.

ADC: Analog-to-Digital Conversion.

FFT: Fast Fourier Transform**.**

STFT: Short-Time Fourier Transform (STFT).

MFCCs: Mel-Frequency Cepstral Coefficients.

MFC: Mel-Frequency Cepstrum.

WT: Wavelet Transform.

ML: Machine Learning.

DL: Deep Learning.

CV: Computer Vision.

SD: Smart Device.

CNN: Convolutional Neural Network.

General introduction

With the rapid development of smart technologies and the increasing need for driver assistance systems, technologies directed at people with special needs, especially the hard-of-hearing and deaf, have become essential to ensure their safety and improve the quality of their driving. The topic of this note is to design a smart system to help hard-of-hearing and deaf drivers recognize important ambient sounds while driving, such as sirens and car horns, by converting audio signals into visual and haptic alerts in real time.

The importance of this research stems from the fact that it contributes to supporting a segment of society that may face real difficulties while driving due to their inability to hear warning sounds that could save lives or prevent accidents. Providing a system that helps these drivers to perceive their auditory environment in an alternative way enhances safety and social inclusion, and reduces the risk of traffic accidents.

The motivations for choosing this topic are the desire to employ artificial intelligence and modern technologies, such as machine learning and audio signal processing, to create a practical and effective solution that serves a group that suffers from real difficulties in daily life. The fact that this system can be implemented on low-cost hardware such as the Raspberry Pi makes it more realistic and widely applicable.

This note consists of three main chapters:

The first chapter: In this chapter, we will provide an overview of previous studies and existing systems that aim to support hearing-impaired individuals, particularly in the context of driving. This literature review summarizes key findings, technological approaches, and identified gaps in the development of assistive technologies.

The second chapter: In this chapter, we will explore the basics of audio signal processing and the key frequency-domain techniques used to extract meaningful features from sound. These methods are essential for enabling artificial intelligence systems to classify and interpret environmental sounds such as sirens and car horns, especially in assistive applications for hearing-impaired individuals.

The third chapter: In this chapter, we will present the preparation and design of the proposed smart system, which includes its hardware and software components, as well as the stages of data collection, model training, testing, and implementation on the Raspberry Pi board. The aim of this chapter is to explain the steps involved in building a system capable of recognizing road-related sounds—such as car horns, sirens, and engine noises—and converting them into visual and haptic alerts, thereby supporting hearing-impaired drivers and enhancing their safety while driving.

Finally, this work concludes with a general conclusion that presents the main results obtained and highlights the significance of the proposed system. The conclusion emphasizes how such a solution can contribute to improving road safety for hearing-impaired drivers, promoting their autonomy, and encouraging further research and innovation in the field of assistive.

Chapter I
Literature Review

I.1 Introduction:

Deaf and hard-of-hearing (DHH) individuals encounter substantial challenges in daily life, particularly in contexts where auditory cues play a critical role—such as driving, education, and social interaction. These challenges can affect not only safety but also independence and quality of life. In response, assistive technologies have emerged as powerful tools to bridge the communication gap, enhance situational awareness, and foster greater social inclusion for individuals with hearing impairments.

This chapter presents a comprehensive review of recent advancements in this field, with a particular focus on intelligent systems designed to support DHH users in mobility and safety-critical environments. Notable solutions include wearable devices, smart gloves, auditory alert systems, and multimodal feedback technologies powered by artificial intelligence and the Internet of Things (IoT).

Given that many of these systems rely on the ability to detect, analyze, and interpret environmental sounds such as vehicle horns or emergency sirens, a foundational understanding of audio signals is essential. Thus, this chapter also introduces the fundamental concepts of audio signal representation and processing. It covers the nature of analog and digital audio signals, the frequency ranges of human and vehicular sounds, and the principles of analog-to-digital conversion (ADC), which enable sound signals to be used effectively in AI-based classification models. By integrating technological innovation with signal processing theory, this chapter provides both a practical and theoretical foundation for designing inclusive and intelligent driving assistance systems for the DHH community.

I.2 Overview of previous systems and studies:

With the increasing demand for assistive driving technologies tailored to the needs of hearing-impaired individuals, a wide array of research efforts has emerged, aiming to bridge the gap between traditional auditory-based alert systems and alternative sensory channels such as visual or haptic feedback.

In one notable initiative, researchers developed the SI-LERT system (Assistive Device for Hearing Impaired Drivers), designed to help deaf drivers detect emergency vehicles. This device uses a Raspberry Pi 4 and a USB microphone to recognize sirens, sending a signal over WiFi to an ESP32 microcontroller embedded in a smart glove. Upon detection, the glove activates micro-

vibrators and displays an emergency vehicle image on an OLED screen, offering a dual feedback mechanism. While promising, the system requires further development, particularly regarding the size and diversity of the dataset used [1].

Similarly, another study proposed a solution to detect auditory cues and identify their direction using an Arduino Uno board integrated with MATLAB, which is compatible with Arduino and allows sound signal acquisition and analysis. This device was designed to detect horns, ambulance sounds, and sirens, which often lack visual signals. The aim was to help deaf drivers perceive where sounds originate, thus improving their awareness on the road [2].

Expanding on wearable technologies, researchers developed a wearable assistant device capable of detecting emergency vehicle sirens using edge computing. They introduced an EfficientNet-based fuzzy rank ensemble model implemented on an Arduino Nano 33 BLE Sense board. The system classified seven types of audio inputs—three types of sirens and four vocalizations—using data from the CREMA-D and Large Scale Audio datasets, totaling 8756 files. Features were extracted using Short-Time Fourier Transform (STFT) to create spectrograms, and upon detecting sirens, the device vibrated and displayed alerts on an OLED panel. The model achieved 97.1% accuracy offline and 95.2% on the edge device [3].

In the domain of communication, hearing-impaired individuals often rely on sign language as a primary mode of interaction. However, communication with non-sign language users remains difficult in the absence of interpreters. To address this, some studies focused on AI-based systems that recognize body gestures such as hand movements and translate them into speech or text using programmable interfaces and learning algorithms. These systems allow users to customize their sign language libraries, making them more inclusive and adaptive to personal needs [4].

To facilitate communication in group environments like classrooms or workplaces, the ACVA (Assistive Communicative Vibrating Aid) was proposed. This wearable bracelet includes an OLED display and a vibration module, and converts clap gestures into WiFi-based broadcast signals. It aims to bridge the gap between augmentative and alternative communication (AAC) and natural behavioral interactions, thereby helping hearing-impaired users draw attention effectively [5].

Further advancing gesture recognition, researchers developed a smart speaking glove for one-handed deaf or mute users communicating in Arabic Sign Language. The glove integrates flexible sensors and a six-dimensional motion tracking unit to convert hand gestures into corresponding

text and speech outputs. The device was tested in cooperation with the Zawia Center for the Deaf and Dumb, focusing on common phrases such as greetings and food-related signs. The glove showed strong performance in mechanical and electrical evaluations [6].

Another study introduced Vibe, a multimodal mobile-based alert system for the deaf and hard of hearing. Vibe gathers sound data from the environment via wall-mounted or mobile microphones, and translates it into visual, tactile, or auditory alerts based on the user's preference. Its design was guided by user personas and structured scenarios to ensure usability. The project developed the system architecture and specifications as a foundation for a working prototype [7].

To better understand the impact of hearing loss (HL) on driving behavior, one PhD thesis presented a three-part study including a questionnaire, driving simulator, and field test. The study concluded that HL does affect road safety and mobility, particularly under complex driving conditions. Drivers with HL exhibited more visual attention behaviors such as frequent mirror-checking and scanning, and responded positively to tactile feedback integrated into GPS systems, which helped them navigate more safely [8].

On the social and economic side, the rise of the gig economy has provided new job opportunities for the DHH community. A study on DHH Uber drivers revealed that many benefit from features in the Driver app that allow them to declare their hearing status. Though only 1.1% of drivers enabled the DHH feature, internal data showed that DHH drivers generally spent more hours on the road. Understanding how DHH drivers interact with app accessibility features offers insights into improving technology for broader inclusion [9].

In domestic applications, an Arduino-based system was proposed to help deaf mothers monitor infants. The device uses a microphone module and moisture sensor, transmitting alerts via a 434 MHz RF module to a receiver with a vibrator, notifying the mother when the baby cries or wets the bed. This project supports hearing-impaired caregivers in home environments [10].

More ambitiously, researchers proposed an integrated assistive device for individuals with visual, auditory, or vocal impairments using Raspberry Pi and Google APIs. The system converts text to speech, images to sound, and speech to text, depending on user needs. It combines a camera, microphone, LCD display, and software libraries to provide all-in-one accessibility, enabling users to read, speak, or listen through digital interaction [11].

Focusing again on driving challenges, several reports highlighted that deaf drivers may have difficulty hearing essential traffic sounds like sirens and horns. They may also face communication

issues with law enforcement or difficulty interpreting GPS instructions relying on voice. Some proposed solutions include vibration-based alert systems, modified vehicles, or training with professionals who understand the needs of deaf drivers. These accommodations can enhance confidence and safety behind the wheel [12].

Additionally, haptic feedback systems have gained interest in supporting the DHH population. A comprehensive PRISMA-based literature review categorized current haptic devices and identified trends in tactile communication research. The review found that haptic devices could supplement or replace auditory information for people with limited hearing, thus offering a promising direction for further study in assistive technologies and human-computer interaction [13].

Finally, recent work introduced an IoT-based horn detection and classification system to assist both deaf and distracted drivers. By recognizing the direction and intensity of honking sounds, the system uses machine learning models with majority voting to display alerts on the dashboard. The study reported 95% accuracy using a dataset of 600 audio files, demonstrating its practicality for four-wheel vehicles and its potential in reducing accidents [14].

To further understand how these systems interpret environmental sounds, it is essential to explore the fundamental nature of audio signals and their processing for use in AI classification systems.

I.3 Introduction to Audio Signals:

With the growing reliance on intelligent systems for assistive driving, particularly for individuals who are deaf or hard of hearing (DHH), understanding the structure and properties of audio signals becomes foundational. These systems rely on accurate interpretation of environmental sounds—such as horns, sirens, and engine noise—to provide timely and effective feedback through visual or haptic channels.

An audio signal is a representation of sound that may exist in two primary forms: analog and digital. Analog signals are continuous waveforms that vary smoothly over time, while digital signals are sequences of discrete binary values derived from sampling and quantizing the analog input. Audio signals typically span the frequency range of 20 Hz to 20,000 Hz, which corresponds to the standard range of human hearing [15].

I.3.1 Human and Vehicular Sound Sources:

Audio-based detection systems must account for a variety of sound sources. Among these, the human voice is a significant element. Produced by the vibration of the vocal cords and shaped by the vocal tract, its frequency varies with age and gender:

- Men: 85–180 Hz
- Women: 165–255 Hz
- Children: 250–400 Hz

Under conditions such as singing or shouting, vocal frequencies may exceed 1,000 Hz [16]. Vehicular sounds, both intentional and incidental, are also crucial in classification systems:

- **Horn sounds vary with vehicle type:**
  - Cars: 250–4,000 Hz (typically 500–1,500 Hz)
  - Trucks and buses: 100–500 Hz
  - Bicycles: 2,000–5,000 Hz; electronic horns may reach up to 10,000 Hz [17]
- **Non-honking sounds include engine and tire noise:**
  - Combustion engines: 20–1,000 Hz
  - Electric motors: 100–5,000 Hz
  - Tire-road interaction: 200–1,000 Hz [18]

I.3.2 Analog-to-Digital Conversion (ADC):

a) Sampling

Sampling is the process of measuring the amplitude of the analog signal at regular time intervals. The sampling rate fs, measured in samples per second (Hz), determines the maximum frequency that can be accurately represented. According to the Nyquist theorem, the sampling rate must be at least twice the highest frequency in the analog signal:

$$f_{Nyquist} = 2f_{Max}$$

Where,

- $f_{Nyquist}$ = Nyquist frequency
- f_{Max} = The max frequency that appears in the signal

Failing to meet this requirement results in aliasing, where the digitized signal deviates significantly from the original. **FigureI.1** illustrates how a continuous analog signal is sampled and converted into a digital signal [19].

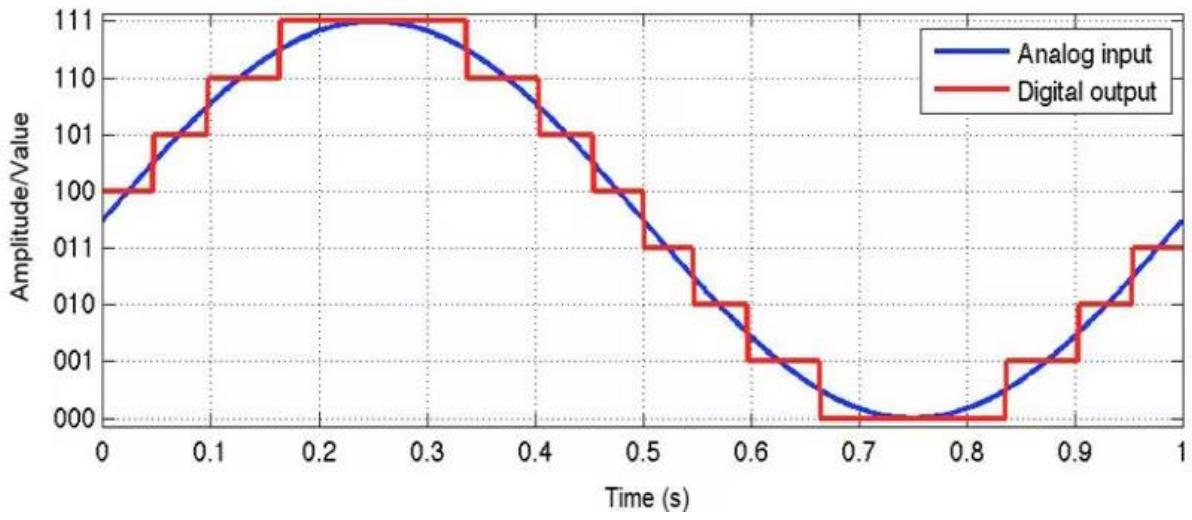
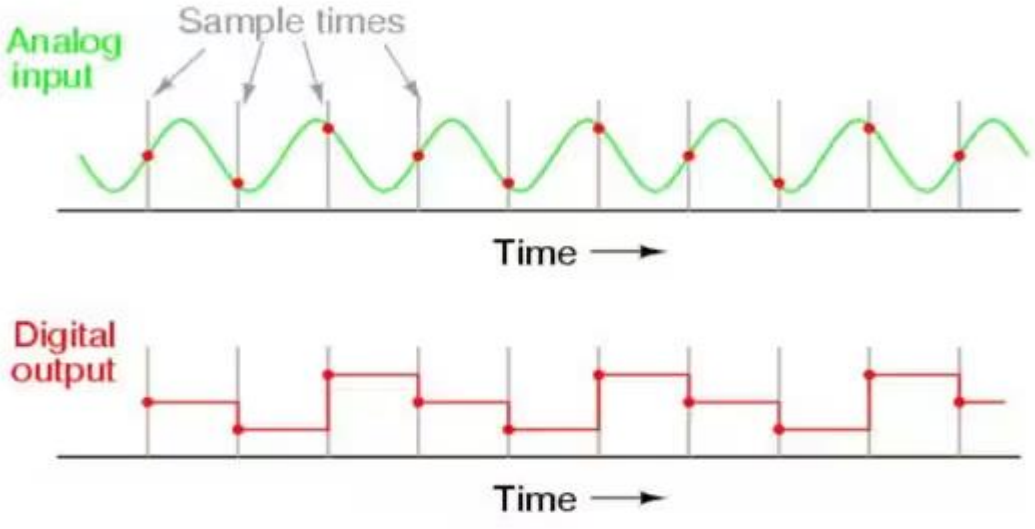


Figure I.1 A continuous analog signal converted into a digital signal through sampling [19].

When the sampling rate is too low, aliasing artifacts occur. These distortions cause the digital output to misrepresent the frequency and shape of the original signal, leading to incorrect interpretation in classification systems. This phenomenon is illustrated in **FigureI.2** [19].



FigureI.2 An example of how aliasing occurs when the sampling rate is insufficient [19].

b) Quantization

After sampling, each analog value is quantized—that is, approximated to the nearest available digital level. The number of quantization levels depends on the bit resolution of the ADC. For example, a 12-bit ADC offers 2^{12} = 4096 distinct levels. The step size (resolution in volts) is given by:

Step Size = V_{Ref} / 2^n

Where,

- **V_{Ref}** is the reference voltage (e.g., 5V)
- **n** is the bit depth.

Table I.1: Common ADC bit lengths with corresponding quantization levels and voltage step sizes for a 5V reference.[19].

Bit Length	Levels	Step Size (5V Range)
8-bits	256	19.53 mV
10-bits	1024	4.88 mV
12-bits	4096	1.22 mV
16-bits	65536	76.29 µV
18-bits	262144	19.07 µV
20-bits	1048576	4.76 µV
24-bits	16777216	0.298 µV

c) Encoding and Storage

Finally, the quantized values are encoded into binary format and stored in digital audio file types such as WAV, MP3, or FLAC.

- Lossless formats (e.g., WAV) retain full detail, ensuring high fidelity.
- Lossy formats (e.g., MP3) reduce file size at the cost of some detail.

Proper encoding ensures compatibility with classification algorithms and real-time applications.

I.4 Conclusion:

This chapter reviewed the current landscape of assistive technologies designed to support deaf and hard-of-hearing (DHH) individuals, with a particular emphasis on solutions that enhance safety, communication, and environmental awareness—especially in driving contexts. A wide range of innovations has been explored, including wearable devices, smart gloves, haptic feedback systems, and AI-enabled sound classification tools.

To support these systems, a foundational understanding of audio signals and their digital representation was also introduced. The chapter discussed key technical principles such as analog versus digital signals, human and vehicular sound frequency ranges, and the stages of analog-to-digital conversion: sampling, quantization, and encoding. These concepts are crucial for transforming real-world auditory information into usable data for intelligent classification and response systems.

Chapter Π
Frequency-Domain Analysis for Sound Classification

II.1 Introduction

This chapter provides an overview of key frequency-domain techniques used in audio signal processing, with a focus on their role in intelligent sound recognition systems. It introduces fundamental methods such as Fourier-based transforms and spectrogram analysis, which allow for the extraction of time-frequency features from audio signals. In addition, the chapter outlines how these features are integrated into artificial intelligence applications—particularly machine learning and deep learning models—to enable tasks such as sound classification, speech recognition, and environmental awareness.

II.2 Frequency-Domain Analysis Methods for Vehicle Horn Sound Classification:

II.2.1 Fast Fourier Transform (FFT):

Engineers frequently examine vibration in relation to frequency. A computing method called the fast Fourier transform (FFT) breaks down a signal into its sine and cosine waves in order to convert time-domain data into the frequency domain. Through this calculation, engineers can see the frequency components of the signal instead of the total of those components.

Engineers can identify the excitation frequencies and amplitudes of a complicated signal with the aid of the FFT. Additionally, it draws attention to amplitude and frequency variations as well as harmonic excitation within the chosen frequency range [20].

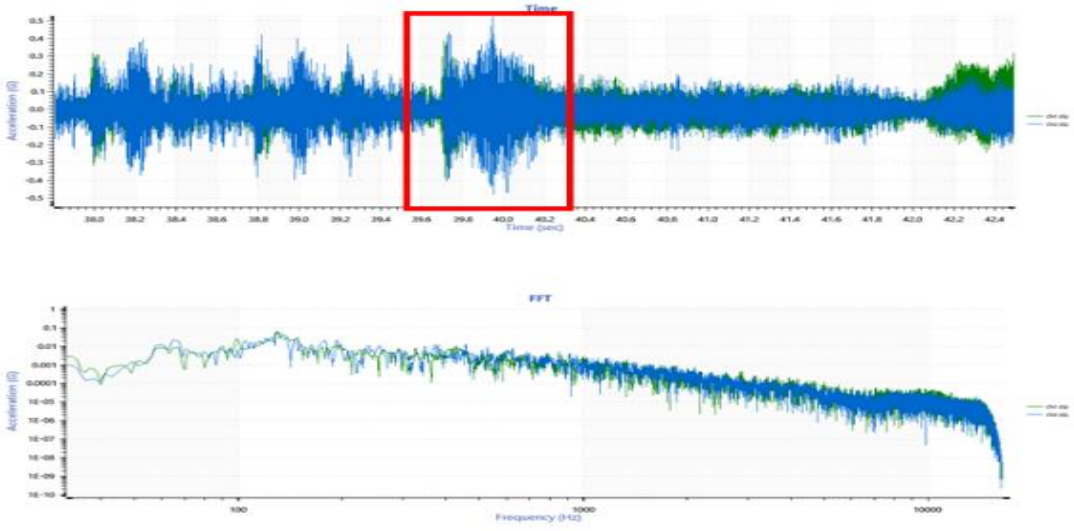


Figure II.1 A segment of time-domain data projected onto the frequency domain [20].

II.2.2 Short-Time Fourier Transform (STFT)

The STFT divides a long audio signal into small, overlapping time segments using a windowing function—typically a Hann or Hamming window. Each segment is analyzed separately using the Fourier Transform, which generates a localized frequency spectrum. By assembling the spectra of all segments, the STFT creates a **spectrogram**: a 2D visualization of frequency (vertical axis), time (horizontal axis), and amplitude (color intensity).

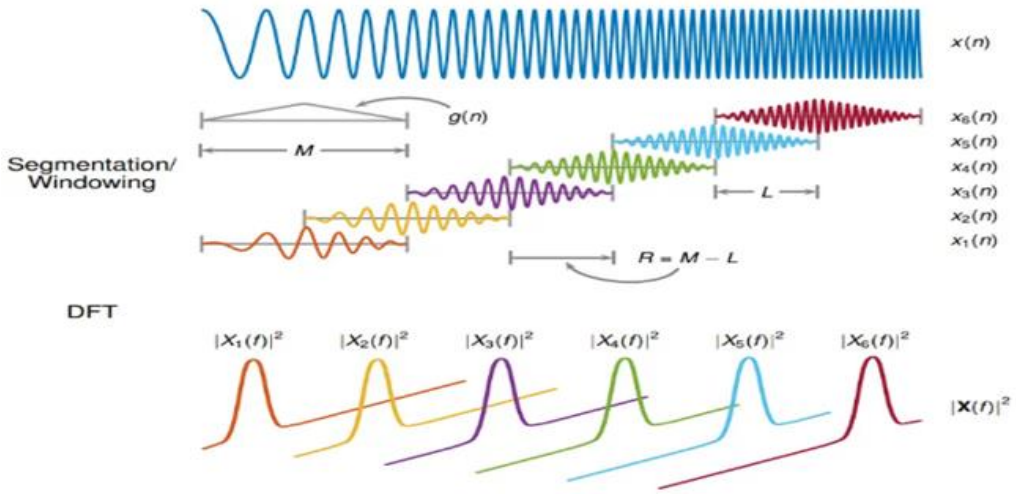


Figure II.2 This image illustrates the STFT process [21].

Several key parameters affect STFT performance:

- **Window size** balances time and frequency resolution.
- **Window overlap** improves continuity but increases computation.
- **Window function** type influences spectral leakage.

STFT is widely used in applications such as speech analysis, music processing, machine fault diagnosis, medical signal monitoring (e.g., EEG, ECG), and telecommunications. Its ability to capture time-localized frequency patterns makes it an essential tool in audio classification and feature extraction systems [21].

II.2.3 Spectrogram:

A spectrogram is a visual representation of the "loudness" or signal strength of a signal across time at different frequencies found in a specific waveform. One can observe how energy levels change over time in addition to whether there is more or less energy at, say, 2 Hz vs. 10 Hz. Spectrophotograms are frequently used in different fields to show the frequencies of sound waves captured by microphones

and produced by people, machines, animals, whales, airplanes, etc. Spectrophotograms are being utilized more and more in the seismic field to examine the frequency content of continuous signals captured by a single seismometer or by a collection of seismometers in order to identify and describe various earthquake or other earth vibration kinds.

In essence, spectrograms are two-dimensional graphs with colors standing in for a third dimension. Along the horizontal axis, time moves from left (oldest) to right (youngest). The tic points along the horizontal axis represent 1-minute intervals, and each of our sub-groups of spectrograms for earthquakes and volcanoes displays 10 minutes of data. With the lowest frequencies at the bottom and the highest at the top, the vertical axis symbolizes frequency, which is also referred to as pitch or tone. The third dimension, color, represents the amplitude (or energy or "loudness") of a certain frequency at a given time; brighter hues up to red correspond to progressively greater (or louder) amplitudes, while dark blues correlate to low amplitudes [22].

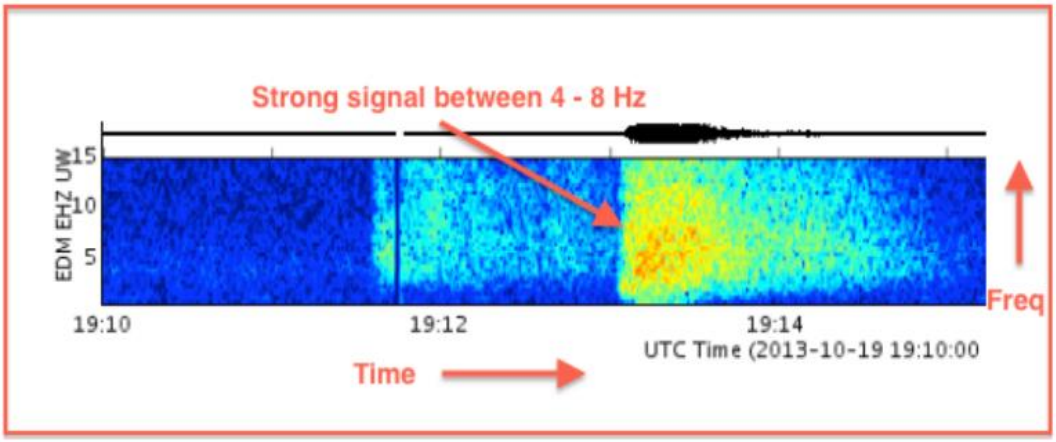


Figure II.3 Detection of a Strong Seismic Signal in the 4–8 Hz Band at Station EDM EHZ (UW) [22].

The raw seismogram, which is shown above the spectrogram, has the same horizontal time axis as the spectrogram (with the same tick marks) and a vertical axis that represents wave amplitude. Seismograms, sometimes referred to as webicorder-style plots, are similar to this plot. Since it displays both the raw waveforms for individual events and the strength or "loudness" at various frequencies, the spectrogram-seismogram combo is a very powerful visualization tool. The frequency content of an event can be important in determining the cause of a signal [22].

II.2.4 Mel-Frequency Cepstral Coefficients (MFCCs)

The mel-frequency cepstrum (MFC), which is based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency, is a representation of a sound's short-term power spectrum in sound processing.

An MFC is composed of a collection of coefficients known as mel-frequency cepstral coefficients (MFCCs). They come from a nonlinear "spectrum-of-a-spectrum" that is a kind of cepstral representation of the audio sample. The MFC differs from the mel-frequency cepstrum in that its frequency bands are evenly spaced on the mel scale, which more closely resembles the response of the human auditory system than the normal spectrum's linearly-spaced frequency bands. Better sound representation, for instance, may be possible through frequency warping in audio compression, which may lower the bandwidth needed for transmission and the amount of storage needed for audio signals. MFCCs are often obtained in this way:

a. Take a signal's windowed extract and perform the Fourier transform on it.
b. Use triangle overlapping windows or, alternatively, cosine overlapping windows to map the powers of the spectrum acquired above onto the mel scale.
c. Compile the power logs for every mel frequency.
d. Treat the list of mel log powers as a signal and take its discrete cosine transform.
e. The amplitudes of the resultant spectrum are known as the MFCCs.

Speech recognition systems, such those that can automatically identify numbers spoken into a phone, frequently use MFCCs as features. Applications for music information retrieval, including genre classification and auditory similarity metrics, are also increasingly using MFCCs [23].

II.2.5 Mel Spectrogram

The Mel spectrogram is a time-frequency representation of audio that maps frequencies to the Mel scale, which reflects human auditory perception of pitch [24]. It is computed by applying a Short-Time Fourier Transform (STFT) to the signal, followed by a Mel filter bank and logarithmic scaling [25]. Unlike linear spectrograms, it provides a perceptually meaningful frequency axis, making it ideal for applications such as speech recognition, sound classification, and music analysis [26],[27]. Its compact and informative format is especially useful in deep learning models like CNNs for audio-based tasks.

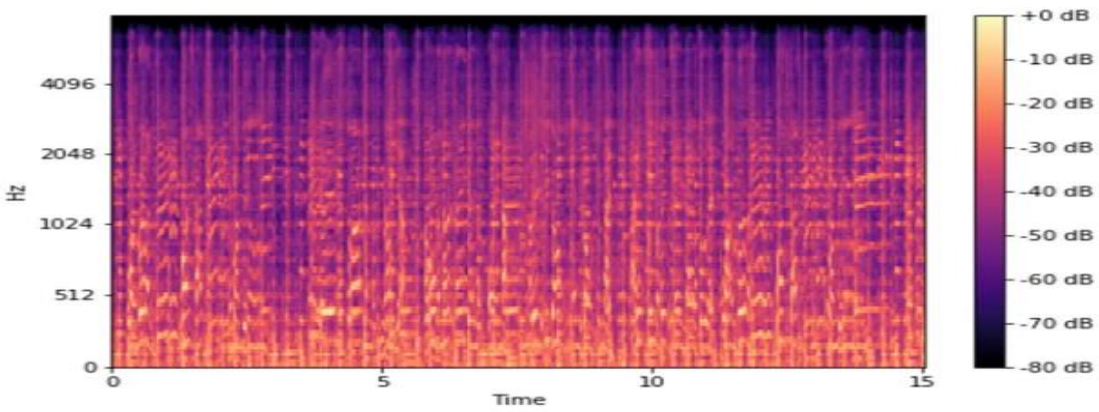


Figure II.4 Mel Spectrogram [28].

II.2.6 Wavelet Transform:

Wavelet Transform

The Wavelet Transform (WT) is a powerful technique used to analyze non-stationary audio signals by providing a time-frequency representation that captures both rapid and slow variations within the signal. Unlike the Fourier Transform, which offers global frequency information, the WT uses scaled and shifted versions of a localized wavelet function—known as the "mother wavelet"—to extract features at multiple resolutions.

In the context of vehicle sound classification, WT can effectively decompose complex horn or engine signals into sub-bands that reflect different frequency behaviors over time. For instance, sudden bursts from a honking sound may be isolated in higher-frequency bands, while low-frequency engine hums are represented in lower bands. This multiresolution capability makes wavelet analysis particularly useful for distinguishing overlapping or temporally varying sound events.

By selecting appropriate wavelet functions and thresholding techniques, irrelevant noise can be filtered out, and only the significant components of the signal—such as the spectral signature of a car or truck horn—can be retained for further classification using machine learning models [29]

Among the various frequency-domain analysis techniques, Mel Spectrograms stand out as one of the most effective tools for audio and speech-related tasks. Unlike other methods such as FFT or STFT, which provide general spectral information, Mel Spectrograms are designed to reflect the nonlinear characteristics of human auditory perception by applying a Mel scale filter bank to the frequency axis.

This makes them particularly well-suited for tasks like horn sound classification, where capturing subtle differences in timbre and tone is essential. Their detailed time-frequency representation and strong performance in applications such as speech recognition, speaker identification, and environmental sound classification make Mel Spectrograms a preferred choice in modern audio analysis systems.

II.3 Artificial intelligence AI

Once features have been extracted from audio signals using frequency-domain techniques, the next step involves classifying these features using artificial intelligence methods. This section provides a foundation in AI concepts relevant to sound classification.

II.3.1 Defintion:

The art and science of developing computer programs and systems that can carry out intricate activities that are comparable to those completed by humans—and occasionally even outperforming human cognitive capacities in particular domains—is referred to as "Artificial Intelligence" [30].

A variety of contemporary approaches and methodologies, including machine learning, artificial neural networks, voice and image recognition, and other technologies, are frequently used in the construction of these systems and applications [31].

In a variety of scientific, social, industrial, economic, medical, and other domains, artificial intelligence seeks to enhance and expand automated performance and production as well as offer intelligent answers to both big and little issues. In addition, it is a fascinating area of study for academics, scientists, engineers, and businesspeople, and it has promise for raising living standards across a range of civilizations [32].

- **How does Artificial intelligence (AI) work :**

In order to comprehend how artificial intelligence (AI) functions, one must study a variety of AI domains, including machine learning, deep learning, neural networks, natural language processing, computer vision, and cognitive computing. These domains allow AI systems to process and interpret large amounts of data, facilitating intelligent decision-making [33]. AI functions by using particular algorithms that gather data and gradually improve their performance. These algorithms serve as the basis for managing intricate business tasks and online transactions.

The diagram shown in Figure II.5 illustrates how does Artificial intelligence (AI) Work:

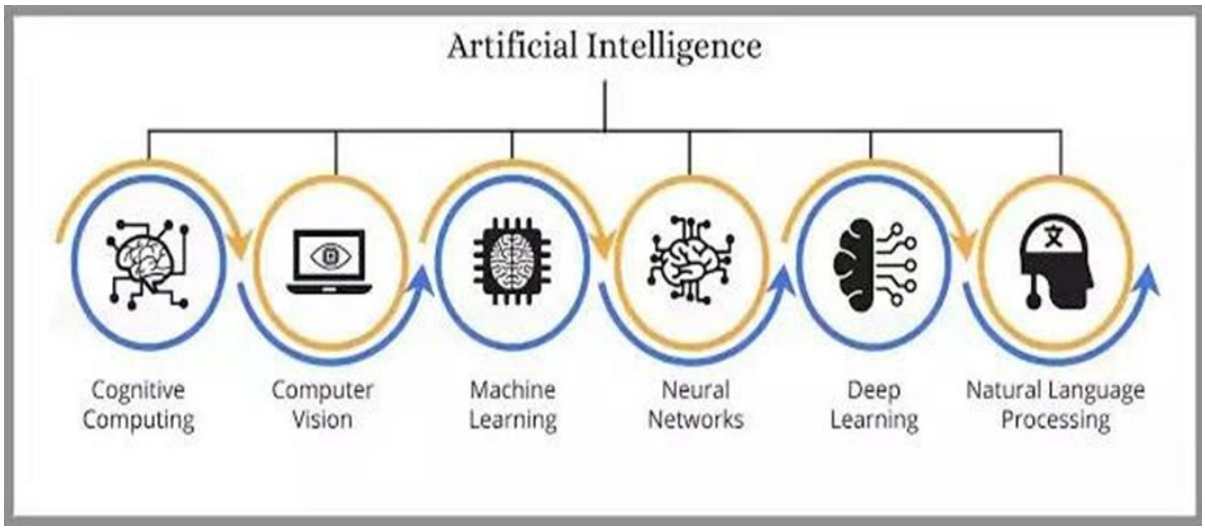


Figure II.5: AI Work [33].

II.3.2 Machine Learning (ML)

As a branch of artificial intelligence, machine learning focuses on creating models and algorithms that let computers learn from data and make judgments or predictions without explicit programming. This type of statistical analysis makes predictions or judgments by analyzing big datasets, finding patterns and correlations, and using algorithms and models. [33].

Reinforcement learning, supervised learning, and unsupervised learning are a few popular categories of machine learning algorithms. The system is taught using labeled data in supervised learning, where the right response is given. Unsupervised learning requires the algorithm to discover patterns and associations on its own after being taught on unlabeled data. Reinforcement learning involves giving the algorithm feedback in the form of incentives or penalties, which allows it to learn via trial and error [34].

There are several uses for machine learning in a variety of industries, including as marketing, banking, and healthcare. It is employed for applications including anomaly detection, natural language processing, picture and audio recognition, and predictive modeling [36].

❖ **How does Machine Learning (ML) work**

One of the main components of machine learning is the ability to generalize from the training data to unseen data, which is achieved by training machine learning algorithms on a dataset to produce a model that can perform tasks or make predictions based on new input data. During the

training process, the algorithm learns patterns, relationships, and features from the training dataset. After receiving new, unseen data as input, the model can use the learned patterns to perform the desired task or make predictions [37].

The diagram shown in Figure II.6 illustrates how does Machine Learning (ML) work:

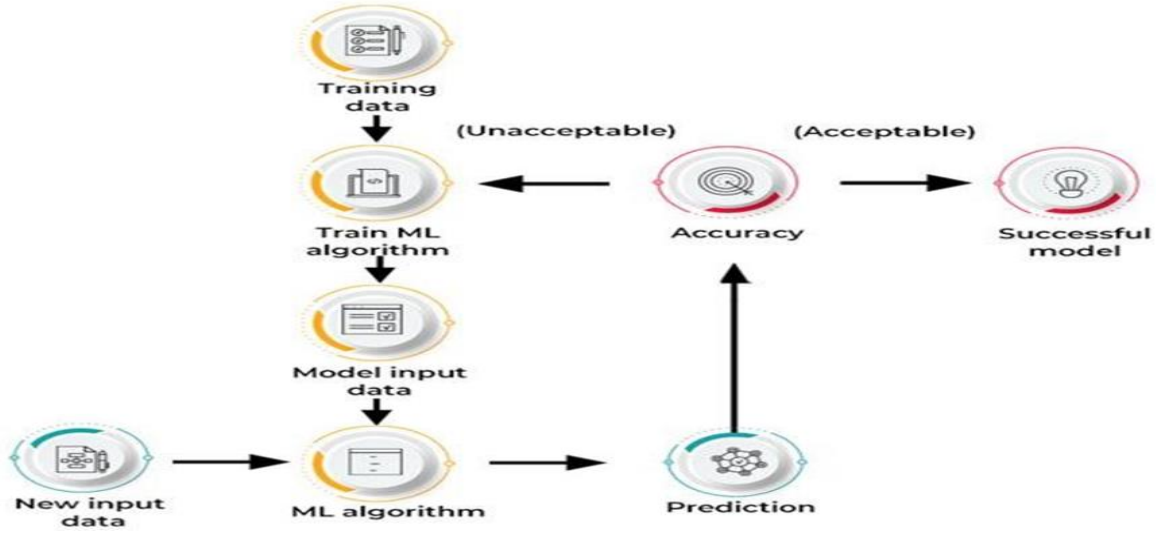


Figure II.6 ML Work [37].

❖ **Machine Learning Components :**

Machine learning typically involves three main components:

a. Model: A machine learning system's model is a mathematical depiction of the connection between its inputs and outputs. The method used for the specific purpose informs the model's architecture, and it is trained on a dataset to identify patterns and correlations [38].
b. Algorithm : The machine learning system uses an algorithm, which is a collection of guidelines or processes, to learn from the data and use that knowledge to generate predictions or choices. The kind of issue being handled and the properties of the data are taken into consideration while selecting the method [38].
c. Data : The machine learning system learns and makes predictions or choices using this raw input. The machine learning system's performance depends on both the amount and quality of the data [39].

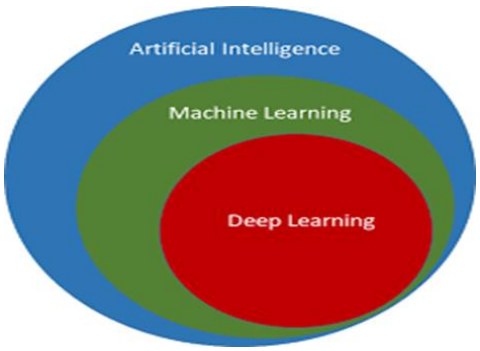
II.3.3 Deep learning DL

Deep learning is a branch of machine learning that models and resolves complicated issues using multi-layered artificial neural networks. It can learn to identify patterns in data without

explicit programming since it is modeled after the composition and operations of the human brain. Numerous applications, such as computer vision, natural language processing, speech recognition, and gaming, have seen notable breakthroughs because to deep learning. Additionally, it has demonstrated cutting-edge performance in a variety of difficult tasks, including language translation, picture and speech recognition, and gaming [35].

II.3.4 The Relationship between Artificial intelligence, machine learning and deep learning

An outline of advancements in deep learning, machine learning, and artificial intelligence is given in this section. It has useful information for anyone who are interested in these subjects. Figure II.7 shows the main differences between these three scientific disciplines [40].



FigureII 7 AI, ML and DL[40].

- **Artificial Intelligence (AI)**

  One essential element of computer science is artificial intelligence (AI) [41],[42]. The ability of computers to do activities like teaching, logical thinking, self-correction, and self-programming—tasks that are often associated with human intelligence—represents a scientific accomplishment. Making it possible for computers to carry out activities that were previously completed by people is the main goal of artificial intelligence [43]. AI is essential to many scientific and practical domains, and its impact is seen in daily life across a range of sectors, including industry, agriculture, and health. Surgeons, for example, use robots to carry out extremely precise procedures. Learning, pattern recognition, reasoning, problem-solving, visual perception, and language understanding are among the skills that AI has the potential to improve [40].

- **Machine Learning (ML)**

  A subset of artificial intelligence, machine learning entails creating IT systems that can recognize patterns and connections in data without the need for explicit programming [44], [45]. Machine learning has shown itself to be quite helpful over time in a number of fields, including research, business, and enhancement [46],[47]. Automatic knowledge generation, algorithm training, connection discovery, and pattern recognition are among its capabilities. Predictions and process optimization can be achieved by applying the patterns and correlations found by machine learning to previously unexplored datasets [40].

- **Deep Learning (DL)**

  A subfield of machine learning called "deep learning" aims to develop models that can solve challenging real-world issues at a level that is similar to that of the human brain. Artificial neural networks and simulation-based learning strategies are used to do this [48], [45]. Deep learning models can analyze and understand enormous volumes of data to provide precise predictions and judgments because they are made to resemble the composition and operations of the human brain. These models may handle complex and difficult issues in a variety of disciplines by using deep neural networks with numerous layers to extract complex patterns and representations from the data [40].

  Deep learning is a branch of machine learning that relies on building models composed of multiple layers of neural networks, aiming to automatically learn complex data representations. Among the most prominent and powerful deep learning models are Convolutional Neural Networks (CNNs), which have proven to be highly effective in handling spatially structured data such as images and audio. CNNs are based on the concept of gradually extracting features through multiple layers of convolution and pooling, enabling them to recognize fine and abstract patterns in data without the need for manual feature engineering.

II.3.5 Convolutional Neural Network (CNN or ConvNet)

Convolutional neural networks (CNNs) are artificial intelligence systems based on multi-layer neural networks that can identify, recognize, and classify objects as well as detect and segment objects in images. In fact, CNN or ConvNet is a popular discriminative deep learning architecture that could be learned directly from the input object without the obligation for human feature extraction. This network is frequently used in visual identification,

medical image analysis, image segmentation, NLP, and many other applications since it is specifically designed to deal with a range of 2D shapes . It is more effective than a regular network since it can automatically identify key elements from the input without the need for human participation [49].

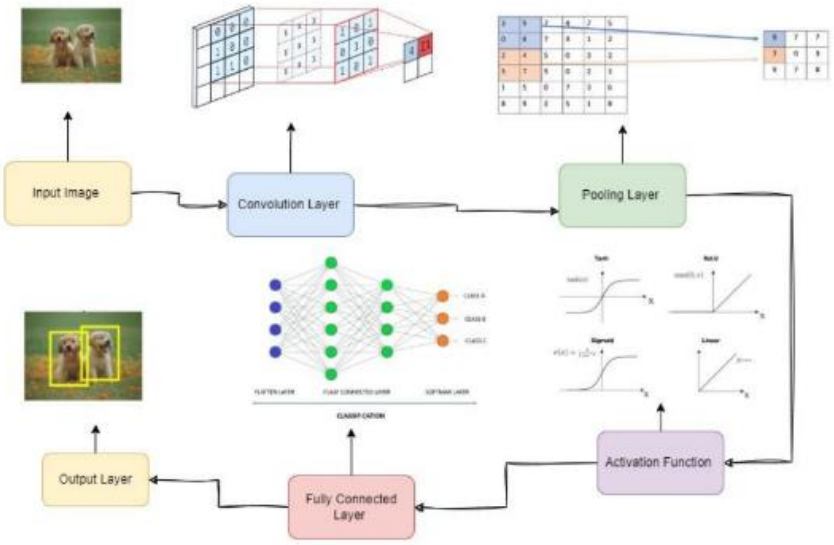


Figure II.8: The CNN Components [49].

II.4 Conclusion

This chapter presented a structured overview of fundamental techniques in audio signal processing, with a particular focus on frequency-domain methods and their role in intelligent sound classification systems. Key transformations such as the Fast Fourier Transform (FFT), Short-Time Fourier Transform (STFT), spectrograms, Mel spectrograms, MFCCs, and Wavelet Transform were introduced as powerful tools for extracting time-frequency features from complex and non-stationary audio signals.

These techniques enable the analysis of environmental sound signals—such as vehicle horns—by generating representations that reveal meaningful internal patterns suitable for automatic processing. The chapter also discussed how these features can be integrated into artificial intelligence approaches, particularly machine learning and deep learning, to build accurate and adaptive sound classification systems.

In the next chapter, we will develop an intelligent system that analyzes vehicle sounds using Mel spectrogram features, which will be transformed into image-like inputs for a **Convolutional** Neural Network (CNN). This system aims to support hearing-impaired drivers by enhancing their awareness of surrounding traffic sounds during driving.

Chapter III

Design of a Smart Device for Assisting Drivers with Hearing Impairments

Chapter III: Design of a Smart Device for Assisting Drivers with Hearing Impairments.

III.1 Introduction:

This chapter presents the practical implementation of an intelligent assistive system designed to support drivers with hearing impairments. The system aims to detect critical traffic sounds, such as vehicle horns and emergency sirens, and translate them into visual and tactile signals. By combining embedded hardware components with artificial intelligence techniques—specifically Mel Spectrogram analysis and Convolutional Neural Networks (CNNs)—the system enhances situational awareness for the driver. This chapter details the overall system architecture, key components, software environment, data processing stages, and the deployment process on a Raspberry Pi**.**

III.2 General Architecture of the Proposed System:

The proposed intelligent system is designed to assist drivers with hearing impairments in perceiving critical environmental sounds—such as sirens and car horns—through non-auditory means, ensuring safe and effective driving. This system is based on a pre-trained artificial intelligence model that utilizes Mel Spectrogram and Convolutional Neural Networks (CNNs) and is deployed on a Raspberry Pi 4 board.

The system operates by continuously capturing ambient audio through a built-in microphone. The captured sounds are then converted into spectral representations using Mel Spectrogram methode. These representations are analyzed by the AI model to determine the presence of nearby vehicles. Upon detection, the system sends two types of alerts to the driver: a tactile signal via a vibration motor embedded in the steering wheel and a visual message, such as "Vehicle Detected", displayed on an LCD screen.

Figure III.1 illustrates the architecture of the proposed smart system, highlighting the integration of its various components to fulfill its assistive function.

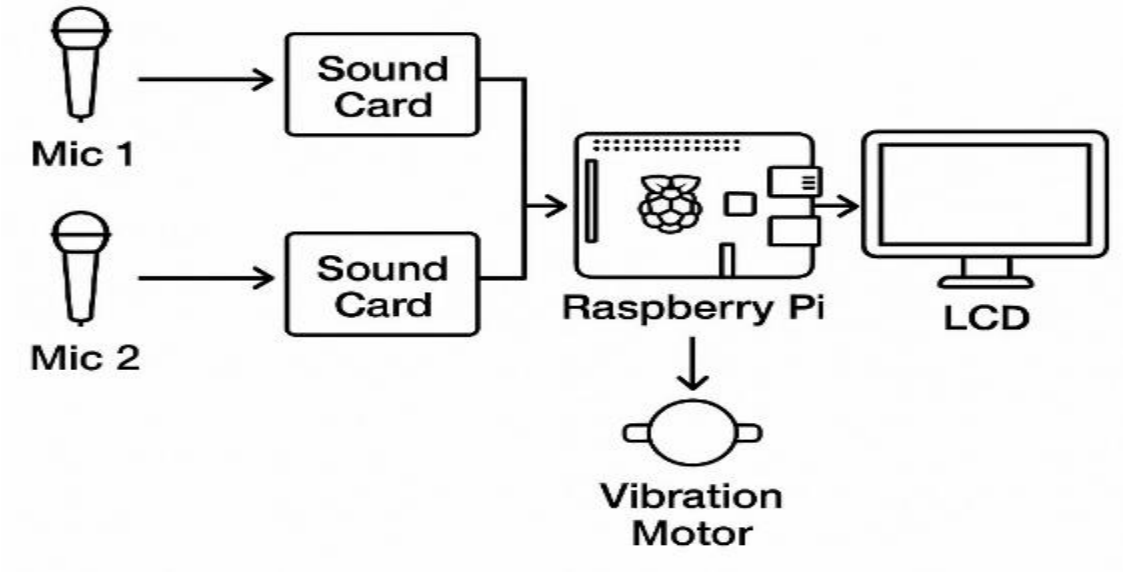


Figure III.1 A Smart Assistive System for Individuals with Hearing Impairments**.**

III.3 System Components:

The system is built using a set of key electronic and technological components, summarized in the following table:

Table III.1: Component of the smart Device for Deaf Drivers

Components	Function
Raspberry Pi 4 Model B 	The Raspberry Pi 4 Model B is a powerful single-board computer equipped with a Broadcom BCM2711 quad-core Cortex-A72 (ARM v8) 64-bit SoC, available with 2GB, 4GB, or 8GB of RAM. It features dual-band 2.4GHz and 5GHz wireless LAN, Bluetooth 5.0, two USB 3.0 ports, two USB 2.0 ports, Gigabit Ethernet, dual micro-HDMI ports supporting up to 4K display, a USB-C power connector, a 3.5mm audio/video jack, and a 40-pin GPIO header. With significantly enhanced processing power, memory, and connectivity compared to its predecessors, the Raspberry Pi 4B is widely used in advanced DIY projects, educational tools, prototyping, and embedded systems.

USB Sound Card 	The USB Sound Card is an external audio interface that provides sound input and output capabilities through a USB connection. It typically includes a headphone jack and a microphone input, enabling audio capture and playback without relying on the computer's built-in sound hardware. USB sound cards are compact and plug-and-play, making them ideal for use in embedded systems such as the Raspberry Pi, especially when high-quality or additional audio input/output is needed. In the context of a smart system, the USB sound card plays a crucial role in capturing environmental audio for further processing by the artificial intelligence model.
Mic_Only_Headset_Jack 	The Mic-Only Headset Jack is an audio interface designed specifically for connecting a mono or stereo microphone input through a 3.5mm audio jack. Unlike standard headphone jacks, this component is dedicated solely to capturing audio signals, making it ideal for voice input in embedded systems and smart devices. When used with a USB sound card or directly with a Raspberry Pi that supports microphone input, the Mic-Only Headset Jack enables clear and focused audio capture, which is essential for applications such as voice recognition, environmental sound analysis, or real-time audio processing in assistive technologies.
Vibration Motor 	This motor is mainly designed to generate strong vibrations.It works by spinning an eccentric (unbalanced) weight attached to its shaft. As the shaft spins, the off-center weight creates a wobbling force, which feels like a vibration.
7-Inch LCD Screen 	A 7-inch LCD (Liquid Crystal Display) screen is a flat-panel display that uses liquid crystals to produce images. With a resolution of 1024x600 pixels, it offers clear and sharp visuals suitable for various applications. These screens are commonly used in embedded systems, portable devices, and as user interfaces in industrial equipment. They can display information, videos, or serve as touch interfaces when equipped with a touch-sensitive layer.

The integration of these components enables the system to translate auditory signals into clear visual and tactile notifications, enhancing situational awareness for drivers with hearing impairments and supporting timely decision-making while driving.

III.4 Programming Environment and Required Libraries:

The development of the smart assistive device for drivers with hearing impairments was carried out using the Python programming language due to its flexibility and the wide availability of open-source libraries suitable for artificial intelligence and signal processing tasks. The implementation was done in a cloud-based environment using Google Colaboratory (Google Colab), which offers the power of Jupyter notebooks with access to high-performance computational resources such as GPUs and TPUs.

Google Colab was particularly useful for this project, providing the following advantages:

- Writing, executing, and debugging Python code for audio signal processing.
- Extracting Mel spectrograms using the librosa library for sound classification.
- Training deep learning models with TensorFlow or Keras, using free GPU acceleration.
- Easy team collaboration via Google Drive integration.
- Avoiding the need for local hardware setup, which is especially beneficial for machine learning projects requiring substantial computational resources.

Libraries Used:

A range of Python libraries were employed to perform various tasks throughout the development process. The following is a summary of the key libraries and their functions:

- **os**: A built-in Python module used for interacting with the operating system, including file and directory management.
- **numpy:** A fundamental library for numerical computations, particularly useful for array manipulations and data preprocessing.
- **librosa:** Specialized in audio and music signal processing. It was used to extract features like Mel spectrograms from audio signals.

- **librosa.display:** A submodule used to graphically visualize audio signals such as waveforms and spectrograms.
- **matplotlib.pyplot:** A plotting library used to visualize audio features and model outputs via graphs and charts.
- **tensorflow:** An open-source deep learning framework developed by Google, used to design and train the neural network model for sound classification.
- **tensorflow.keras:** A high-level API within TensorFlow that simplifies model development, compilation, and training workflows.
- **tensorflow.keras.layers:** Provides essential neural network layers like Dense and Conv2D, used in defining the model architecture.
- **sklearn.model_selection.train_test_split:** A function from the Scikit-learn library used to split the dataset into training and testing subsets.
- **sklearn.preprocessing.LabelEncoder:** Converts categorical text labels (e.g., "Horn", "Engine") into numerical values for model compatibility.
- **time:** A standard library module used to measure execution time and monitor the duration of various operations.
- **sounddevice:** Facilitates real-time audio input and output with low latency, ideal for capturing ambient sounds.
- **playsound**: A simple module used for playing back recorded or processed audio clips, aiding in result verification and user feedback.

These tools and libraries collectively formed the foundation for programming the smart device and optimizing its performance to detect and respond to relevant environmental sounds effectively.

III.5 Stages of the project:

III.5.1 Sound collection phase:

In this stage, we will collect the sounds that we will later train the model with, and the obtained set of sounds will go through several stages to be ready for the training process, we will include

our own sounds for the set and we will use URecorder to record the sounds in Wav format as shown in the following figure:

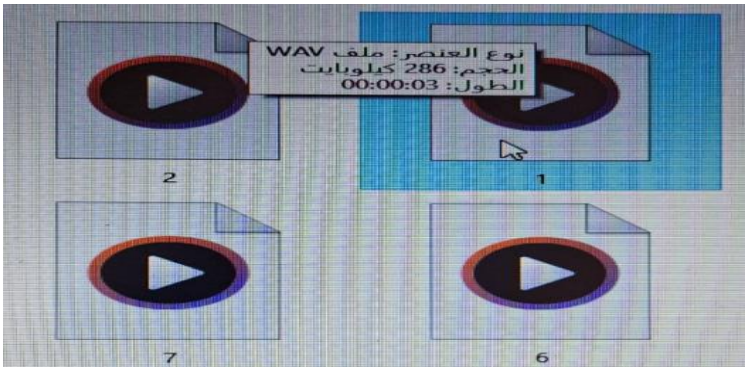


Figure III.2: Collect Wav sounds.

We categorize the sounds we collected into two parts: the first is the presence of a vehicle with horn sounds (car, ambulance, truck, and motorcycle) and the second is the absence of a vehicle (road noise and vehicle engines).

Figure III.3: The shape of the two files obtained after the categorization process.

III.5.2 Feature extraction stage:

After we have collected the sounds, the feature extraction phase comes to the mel spectrogram.

❖ **Extraction code for mel spectrogram:**

```
import librosa
import librosa.display
import matplotlib.pyplot as plt
import numpy as np
import os
import shutil

# Input folders
son_folder = "/content/drive/MyDrive/dataseet_wav/car_horn"  # Vehicle sounds
route_fina_folder = "/content/drive/MyDrive/dataseet_wav/road_sound"  # Road sounds

# Output base folder
output_path_mel = "/content/drive/MyDrive/MEL_SPECTROGRAM_DATA"  # New output folder for Mel spectrograms
os.makedirs(os.path.join(output_path_mel, "vehicle"), exist_ok=True)
os.makedirs(os.path.join(output_path_mel, "non_vehicle"), exist_ok=True)

# Parameters
n_fft = 2048  # Number of FFT points (adjust as needed)
hop_length = 512 # Hop length (adjust as needed, typically n_fft / 4)
n_mels = 128 # Number of Mel bands (adjust as needed)

# Function to generate Mel spectrogram and save it
def process_and_save_mel_spectrogram(audio_path, label, index):
    audio_data, sample_rate = librosa.load(audio_path)
```

```
  # Generate Mel spectrogram
  mel_spectrogram = librosa.feature.melspectrogram(y=audio_data, sr=sample_rate, n_fft=n_fft, hop_length=hop_length, n_mels=n_mels)
  mel_spectrogram_db = librosa.amplitude_to_db(mel_spectrogram, ref=np.max)

  # Plot (no display)
  plt.figure(figsize=(2.24, 2.24), dpi=100)
  plt.axis('off')
  librosa.display.specshow(mel_spectrogram_db, sr=sample_rate, hop_length=hop_length)

  # Save the Mel spectrogram in the appropriate folder
  save_path = os.path.join(output_path_mel, label, f"{index}_{os.path.basename(audio_path).replace('.wav', '')}.png")
  plt.savefig(save_path, bbox_inches='tight', pad_inches=0)
  plt.close()

Process files from both folders
ndex = 0

Process vehicle-related files (SON folder)
or audio_file in os.listdir(son_folder):
  if audio_file.endswith('.wav'):
      audio_path = os.path.join(son_folder, audio_file)
      process_and_save_mel_spectrogram(audio_path, "vehicle", index)
      index += 1

# Process road-related files (ROUTE.FINA folder)
for audio_file in os.listdir(route_fina_folder):
    if audio_file.endswith('.wav'):
        audio_path = os.path.join(route_fina_folder, audio_file)
        process_and_save_mel_spectrogram(audio_path, "non_vehicle", index)
        index += 1
```

Figure III.4: Extraction code for mel spectrogram.

❖ **The result for Mel-spectrogram:**

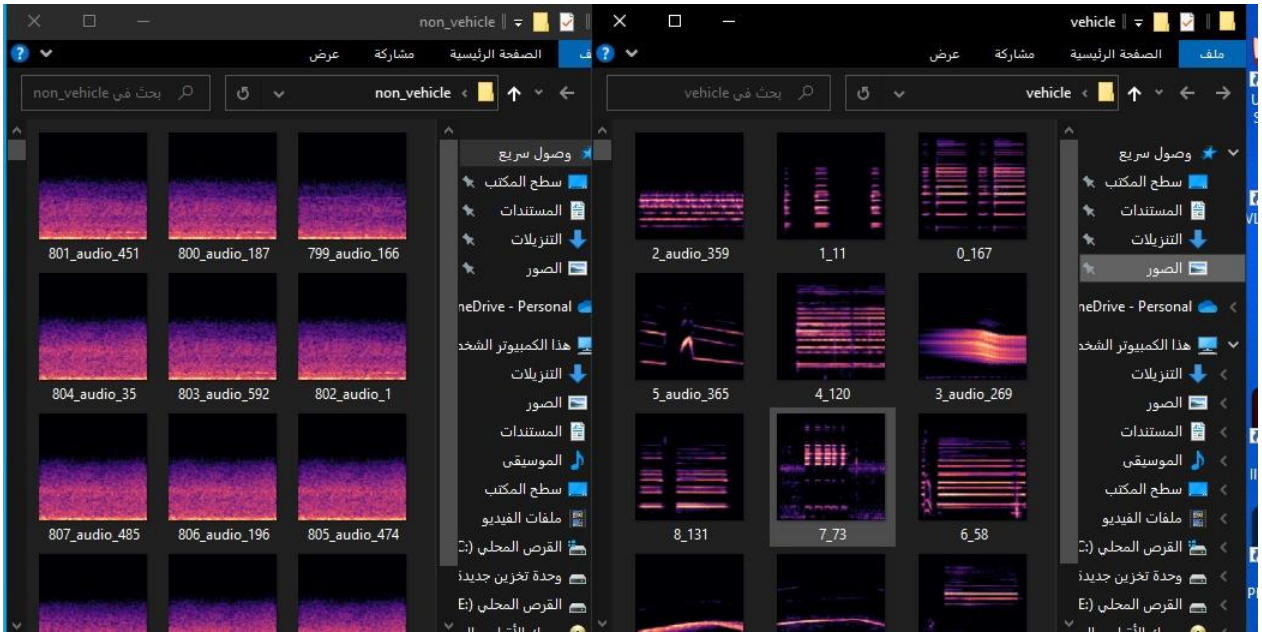


Figure III.5: The result for mel spectrogram.

III.5.3 Model training and transformation phase:

- **Install the necessary libraries:**

  We download the libraries needed for Model, as shown in Figure III.7:

```
# إذا لم تكن مثبتة بالفعل في بيئة (تثبيت المكتبات اللازمة Colab)
!pip install librosa tensorflow numpy matplotlib scikit-learn
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.11/dist-packages (2.18.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (2.0.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (1.6.1)
Requirement already satisfied: audioread>=2.1.9 in /usr/local/lib/python3.11/dist-packages (from librosa) (3.0.1)
Requirement already satisfied: numba>=0.51.0 in /usr/local/lib/python3.11/dist-packages (from librosa) (0.60.0)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from librosa) (1.14.1)
Requirement already satisfied: joblib>=1.0 in /usr/local/lib/python3.11/dist-packages (from librosa) (1.4.2)
Requirement already satisfied: decorator>=4.3.0 in /usr/local/lib/python3.11/dist-packages (from librosa) (4.4.2)
Requirement already satisfied: soundfile>=0.12.1 in /usr/local/lib/python3.11/dist-packages (from librosa) (0.13.1)
Requirement already satisfied: pooch>=1.1 in /usr/local/lib/python3.11/dist-packages (from librosa) (1.8.2)
Requirement already satisfied: soxr>=0.3.2 in /usr/local/lib/python3.11/dist-packages (from librosa) (0.5.0.post1)
Requirement already satisfied: typing_extensions>=4.1.1 in /usr/local/lib/python3.11/dist-packages (from librosa) (4.13.2)
Requirement already satisfied: lazy_loader>=0.1 in /usr/local/lib/python3.11/dist-packages (from librosa) (0.4)
Requirement already satisfied: msgpack>=1.0 in /usr/local/lib/python3.11/dist-packages (from librosa) (1.1.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (25.2.10)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.6.0)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (18.1.1)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.4.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from tensorflow) (24.2)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.3 in /usr/local/lib/python3.11/dist-packages (from
```

Figure III.6: Download the necessary libraries for Model.

- **Training the model:**

After we get all the 1600 sounds we need, we move 20% of them Test and 80% of them Train.

```
[ ]  EPOCHS = 25  # (يمكن زيادته أو تقليله حسب النتائج) عدد دورات التدريب
     BATCH_SIZE = 32 # حجم الدفعة في كل خطوة تدريب

     print("\nبدء تدريب النموذج...")
     start_time_train = time.time()

     # تدريب النموذج
     history = model.fit(
         X_train,
         y_train,
         batch_size=BATCH_SIZE,
         epochs=EPOCHS,
         validation_data=(X_val, y_val) # استخدام بيانات التحقق لمراقبة الأداء
     )

     end_time_train = time.time()
     print(f"\nاكتمل التدريب في {end_time_train - start_time_train:.2f} ثانية.")
```

Figure III.7:. Code training.

- **The result obtained:**

```
بدء تدريب النموذج...
Epoch 1/25
40/40 ━━━━━━━━━━━━━━━━━━━━ 72s 2s/step - accuracy: 0.5696 - loss: 172.0896 - val_accuracy: 0.9375 - val_loss: 0.3277
Epoch 2/25
40/40 ━━━━━━━━━━━━━━━━━━━━ 74s 2s/step - accuracy: 0.9859 - loss: 0.0710 - val_accuracy: 0.9875 - val_loss: 0.1819
Epoch 3/25
40/40 ━━━━━━━━━━━━━━━━━━━━ 72s 2s/step - accuracy: 0.9928 - loss: 0.0347 - val_accuracy: 0.9906 - val_loss: 0.0968
Epoch 4/25
40/40 ━━━━━━━━━━━━━━━━━━━━ 71s 2s/step - accuracy: 0.9941 - loss: 0.0198 - val_accuracy: 0.9937 - val_loss: 0.0199
Epoch 5/25
40/40 ━━━━━━━━━━━━━━━━━━━━ 80s 2s/step - accuracy: 0.9940 - loss: 0.0235 - val_accuracy: 0.9875 - val_loss: 0.0641
Epoch 6/25
40/40 ━━━━━━━━━━━━━━━━━━━━ 85s 2s/step - accuracy: 0.9981 - loss: 0.0162 - val_accuracy: 0.9969 - val_loss: 0.0449
Epoch 7/25
40/40 ━━━━━━━━━━━━━━━━━━━━ 79s 2s/step - accuracy: 0.9979 - loss: 0.0076 - val_accuracy: 0.9969 - val_loss: 0.0232
Epoch 8/25
40/40 ━━━━━━━━━━━━━━━━━━━━ 78s 2s/step - accuracy: 0.9978 - loss: 0.0066 - val_accuracy: 1.0000 - val_loss: 0.0098
Epoch 9/25
40/40 ━━━━━━━━━━━━━━━━━━━━ 78s 2s/step - accuracy: 0.9981 - loss: 0.0041 - val_accuracy: 0.9969 - val_loss: 0.0104
Epoch 10/25
40/40 ━━━━━━━━━━━━━━━━━━━━ 73s 2s/step - accuracy: 0.9969 - loss: 0.0048 - val_accuracy: 0.9969 - val_loss: 0.0098
Epoch 11/25
```

Figure III.8:. Result obtained.

- **Evaluation results on verification data**:

```
[ ]  loss, accuracy = model.evaluate(X_val, y_val, verbose=0)
     print(f"\nنتائج التقييم على بيانات التحقق:")
     print(f"  Loss (الخسارة): {loss:.4f}")
     print(f"  Accuracy (الدقة): {accuracy:.4f} ({accuracy*100:.2f}%)")
```

```
نتائج التقييم على بيانات التحقق:
  Loss (الخسارة): 0.0213
  Accuracy (الدقة): (%99.37) 0.9937
```

Figure III.8:. Evaluation results on verification data.

- **The curves represented by the model's accuracy and loss ratios:**

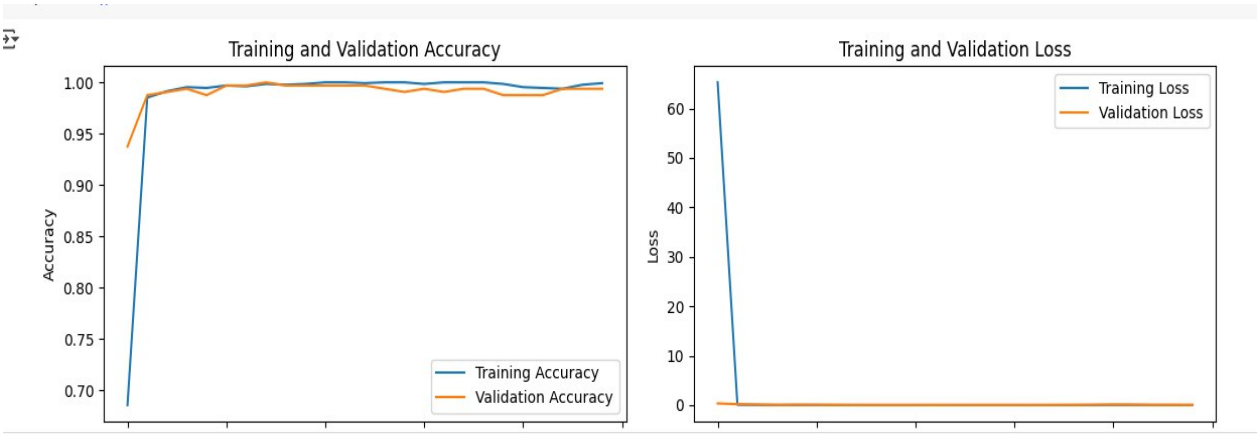


Figure III.9:. The curves represented by the model's accuracy and loss ratios.

Save the entire form:

```
[ ]  # (البنية + الأوزان + حالة المُحسّن) حفظ النموذج بالكامل
     model.save("simple_audio_cnn_model.h5")
     print("تم حفظ النموذج بنجاح باسم simple_audio_cnn_model.h5")

     # لتحميل النموذج لاحقًا:
     # loaded_model = keras.models.load_model("simple_audio_cnn_model.h5")
     # print("تم تحميل النموذج المحفوظ.")
```

```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`.
تم حفظ النموذج بنجاح باسم simple_audio_cnn_model.h5
```

Figure III.10:. Save the entire form.

III.5.4 Testing Model:

- **Full code:**

mama.py

```python
import sounddevice as sd
import numpy as np
import librosa
import tensorflow as tf
from tensorflow import keras
from playsound import playsound

# --- Audio Processing Constants ---
SAMPLE_RATE = 22050  # Standard sample rate for audio processing
DURATION_REC = 2     # Duration of each recorded audio clip in seconds
N_MELS = 128         # Number of Mel bands
HOP_LENGTH = 512     # Hop length for STFT
N_FFT = 2048         # FFT window size

EXPECTED_SAMPLES_REC = int(DURATION_REC * SAMPLE_RATE)

        # 2. Resample if needed
        if sr != target_sr:
            print(f"Resampling from {sr}Hz to {target_sr}Hz...")
            audio_data = librosa.resample(y=audio_data.astype(np.float32), orig_sr=sr, target_sr=target_sr)
            sr = target_sr

        # 3. Padding or truncating to fixed duration
        current_samples = len(audio_data)
        target_samples = int(duration * target_sr)
        if current_samples < target_samples:
            audio_data = np.pad(audio_data, (0, target_samples - current_samples), mode='constant')
        elif current_samples > target_samples:
            audio_data = audio_data[:target_samples]

        # 4. Extract Mel Spectrogram
        mel_spectrogram = librosa.feature.melspectrogram(y=audio_data, sr=sr, n_fft=n_fft, hop_length=hop_l

        # 5. Convert to dB
        log_mel_spectrogram = librosa.power_to_db(mel_spectrogram, ref=np.max)

        # 6. Ensure width matches model input
        expected_width = model_input_shape[1]
        current_width = log_mel_spectrogram.shape[1]
        if current_width < expected_width:
            pad_width_spec = expected_width - current_width
            log_mel_spectrogram = np.pad(log_mel_spectrogram, ((0, 0), (0, pad_width_spec)), mode='constant
        elif current_width > expected_width:
            log_mel_spectrogram = log_mel_spectrogram[:, :expected_width]

        # 7. Add batch and channel dimensions
            log_mel_spectrogram = log_mel_spectrogram[np.newaxis, ..., np.newaxis]

            # 8. Ensure float32 type
            log_mel_spectrogram = log_mel_spectrogram.astype(np.float32)

            return log_mel_spectrogram

        except Exception as e:
            print(f"Error while processing audio: {e}")
            return None

# --- Prediction Function ---
def predict_and_print(preprocessed_spectrogram):
    prediction = model.predict(preprocessed_spectrogram)
    probability_horn = prediction[0][0]

    if probability_horn > 0.5:
        print("\n===> Vehicle detected (Car horn likely) <===")
    else:
        print("\n===> No vehicle detected (Likely road noise) <===")

# --- Audio Recording Function ---
def record_audio(duration_sec=DURATION_REC, fs=SAMPLE_RATE):
    print("Recording started...")
    audio_data = sd.rec(int(duration_sec * fs), samplerate=fs, channels=1)
    sd.wait()  # Wait until recording finishes
    print("Recording finished.")
    return audio_data[:, 0], fs  # Return mono signal
```

```
88
89    # --- Start the test ---
90    audio_data_rec, samplerate_rec = record_audio()
91
92    # Optionally play the recorded audio - make sure  playsoundis installed
93    # playsound(audio_data_rec, samplerate_rec)
94
95    spectrogram_to_predict = preprocess_audio(audio_data_rec, samplerate_rec)
96    predict_and_print(spectrogram_to_predict)
97
```

Figure III.11:. Full code.

❖ **The results that emerged at the end of the experiment:**

- **Result Véhicule détecte:**

```
===> Vehicle detected (Car horn likely) <===
PS C:\Users\ounz\Desktop\project> & C:/Users/ounz/AppData/Local/Programs/Python/Python312/python.exe c:/Users/ounz/Desktop/project/mama.py
2025-05-15 19:19:40.271828: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical re
sults due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ON
EDNN_OPTS=0`.
sults due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ON
EDNN_OPTS=0`.
2025-05-15 19:19:54.859190: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU in
structions in performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile_metrics` will be empty until you train
 or evaluate the model.
Recording started...
Recording finished.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 417ms/step

===> Vehicle detected (Car horn likely) <===
```

Figure III.12 Result Véhicule Détecte.

- **Result No Véhicule détecte:**

```
PS C:\Users\ounz\Desktop\project> & C:/Users/ounz/AppData/Local/Programs/Python/Python312/python.exe c:/Users/ounz/Desktop/project/mama.py
2025-05-15 19:20:23.263794: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical re
sults due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ON
EDNN_OPTS=0`.
2025-05-15 19:20:26.846289: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical re
sults due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ON
EDNN_OPTS=0`.
2025-05-15 19:20:37.179397: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU in
structions in performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile_metrics` will be empty until you train
 or evaluate the model.
Recording started...
Recording finished.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 309ms/step

===> No vehicle detected (Likely road noise) <===
```

Figure III.13: Result No Véhicule Détecte.

III.6 System Settings of Raspberry Pi:

To set up the Raspberry PI system we follow the following steps:

- We reformat a card micro SD.
- We download the operating system of Raspberry pi. It is Raspbian by Raspberry pi Imager from the official website of Raspberry pi.

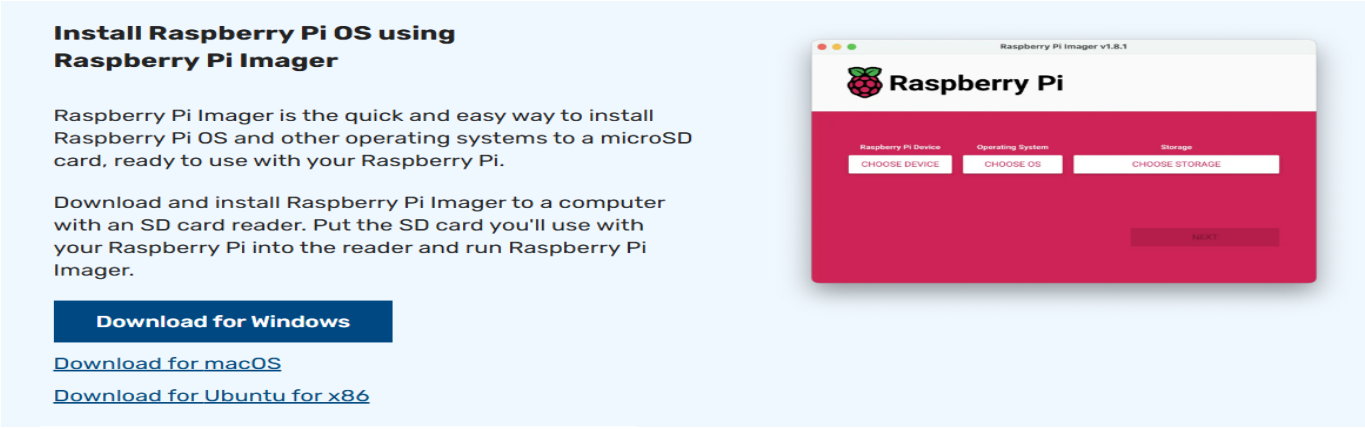


Figure III.14: Install Raspberry pi operating system.

- We connect card a micro SD to the Raspberry pi.

Figure III.15: Connect a card a micro SD.

First, we connect the keyboard and mouse to the USB ports. Then, we connect an HDMI cable to display the system interface on the screen. Finally, we provide power to the Raspberry Pi using a 5V power supply.

Fig III.16: Connect Raspberry Pi to the screen.

- We install the Raspbian operating system on the Raspberry pi and then adjust the system settings.
- The tried-and-true modal app on the computer to Raspberry Pi:

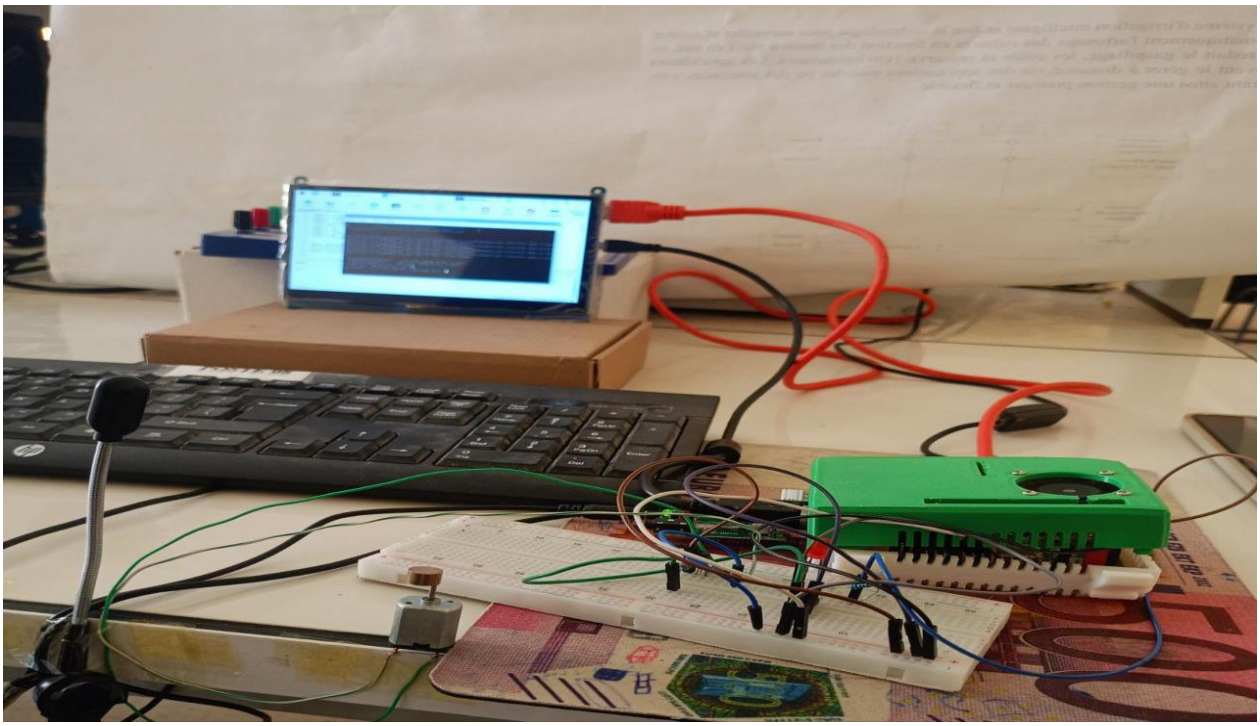


Figure III.17: Image of the entire project.

III.7 Conclusion:

In this final chapter presenting the work done, we built and trained a smart device design model for a deaf driver on a Raspberry Pi using the Python programming language and Simple-audio-CNN. By setting up the experiment environment by downloading the necessary libraries and tools, then collecting and setting up the database of voices and training the model using the CNN language in Google colab. After that, we converted the model to TF-Lite and optimized it in order to reduce space and speed up processing in the Raspberry Pi, which is a small device with low power consumption, and we obtained our classification model, after that we tested the model and obtained very acceptable results up to 60%, which proves that it can be used.

.

.

General Conclusion

In conclusion, we emphasize that the design and development of an intelligent system to assist deaf drivers is an important step towards enhancing traffic safety and social inclusion for people with special needs. Through this work, we were able to combine artificial intelligence, audio signal processing, and microcomputing techniques to create a device capable of recognizing important ambient sounds and converting them into real-time visual and haptic notifications.

The study started by defining the theoretical framework of the issue by reviewing the characteristics of hearing impairment and its impact on driving vehicles, then the technical aspects related to the representation and feature extraction of audio signals using tools such as FFT, Spectrogram, and Mel Spectrogram. An integrated system was built based on a Raspberry Pi board, an LCD display, and a vibration engine, supported by an AI model trained with powerful libraries such as TensorFlow and Librosa to analyze and classify sounds.

The results of the tests conducted proved that the model is able to distinguish between different sounds with an acceptable accuracy of up to 60%, which opens the way for future performance improvements by increasing the number of samples, improving the quality of the recordings, and using deeper and more complex models.

This project, despite its relative simplicity, is a promising basis for the development of intelligent assistance systems for people with sensory impairments, and can be further developed to include more advanced functions such as speech recognition, emergency communication, or even integration into autonomous driving systems.

We hope that this MoU will highlight the importance of developing smart and accessible solutions that ensure equal opportunities and take into account the needs of all segments of society, including people with special needs, towards a more inclusive and safe environment for all.

References

[1] Abhiram, S. P., Umamaheswari, S., Shybin, M., Campbell, A. S., Sreena, V. G., & Raji, N. R. (2024, August). SI-LERT (Assistive Device For Hearing Impaired Drivers). In 2024 7th International Conference on Circuit Power and Computing Technologies (ICCPCT) (Vol. 1, pp. 1516-1521). IEEE..

[2] Lee, J., Elzawawy, A., & Rahemi, H. A. (2014, July). A. Ds: Aid device for deaf drivers. In Twelfth LACCEI Latin American and Caribbean Conference for En Gineering and Technology (LACCEI.

[3] Gupta, B. B. (2021). Cloud Security.

[4] Battina, D. S., & Surya, L. (2021). Innovative study of an AI voice-based smart Device to assist deaf people in understanding and responding to their body language. SSRN Electronic Journal, 9, 816-822.

[5] Kartha, B., Goutham, B., Nair, P. S., & Nair, V. S. (2018, October). Assistive communicative vibrating aid for differently abled. In TENCON 2018-2018 IEEE Region 10 Conference (pp. 0727-0730). IEEE.

[6] Joksimoski, B., Zdravevski, E., Lameski, P., Pires, I. M., Melero, F. J., Martinez, T. P., ... & Trajkovik, V. (2022). Technological solutions for sign language recognition: a scoping review of research trends, challenges, and opportunities. IEEE Access, 10, 40979-40998.

[7] Ravid, U., & Cairns, P. (2008). The Design Development of a Mobile Alert Device for the Deaf and Hard of Hearing. UCL Interaction Centre, University College London.

[8] Thorslund, B. (2014). Effects of hearing loss on traffic safety and mobility. Linkopings Universitet (Sweden)

[9] https://dl.acm.org/doi/fullHtml/10.1145/3290605.3300759 (2019).

[10] Sharmila, V., Kannadhasan, S., Kannan, A. R., Sivakumar, P., & Vennila, V. (Eds.). (2024). Challenges in Information, Communication and Computing Technology: Proceedings of the 2nd International Conference on Challenges in Information, Communication, and Computing Technology (ICCICCT 2024), April 26th & 27th, 2024, Namakkal, Tamil Nadu, India. CRC Press.

[11] Philip, T. V., Ranganath, S., George, S. M., Manchala, S. P., & Gagana, T. S. (2024, October). Raspberry Pi based Assistive Device for Visually Impaired. In 2024 2nd International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS) (pp. 1509-1514). IEEE.

References

[12] https://intensivelessons.co.uk/can-deaf-people-drive (2023).

[13] https://www.mdpi.com/1424-8220/23/6/2968 (2023).

[14] https://rboutaba.cs.uwaterloo.ca/Papers/Conferences/2023/SalemIoT2023.pdf.

[15] Hodgson, Jay (2010). Understanding Records, p.1. ISBN 978-1-4411-5607-5.

[16] https://www.diwanalarab.com

[17] https://www.amazon.com

[18] https://middle-east.michelin.com

[19] https://www.arrow.com/en/research-and events/articles/engineering-resource- basics-of-analog-to-digital-converters.

[20] https://vibrationresearch.com/blog/fast-fourier-transform-fft-analysis/

[21] https://duohub.ai/glossary/s/short-time-fourier-transform-stft

[22] https://www.pnsn.org/spectrograms/what-is-a-spectrogram

[23] https://en.wikipedia.org/wiki/Mel-frequency_cepstrum

[24] Stevens, S. S., Volkmann, J., & Newman, E. B. (1937). A scale for the measurement of the psychological magnitude pitch. The Journal of the Acoustical Society of America, 8(3), 185–190.

[25] Davis, S., & 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), 357–366.

[26] Rabiner, L. R., & Schafer, R. W. (2011). Theory and Applications of Digital Speech Processing. Pearson/Prentice Hall.

[27] Müller, M. (2015). Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications. Springer.

[28] https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53

[29] Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing, 10(5), 293–302.

[30] Bause, M., Esfahani, B. K., Forbes, H., & Schaefer, D. (2019, July). Design for health 4.0: Exploration of a new area. In Proceedings of the design society international conference on engineering design (Vol. 1, No. 1, pp. 887-896). Cambridge University Press.

[31] Ryan, Mark. "In AI we trust: ethics, artificial intelligence, and reliability." Science and Engineering Ethics 26.5 (2020): 2749-2767.

[32] Russell, Stuart J. Artificial intelligence a modern approach. Pearson Education, Inc., 2010.

[33] https://www.atlearner.com/2020/07/what-is-ai.html

[34] Alpaydin, Ethem. Introduction to machine learning. MIT press, 2020.

[35] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. Deep learning. MIT press, 2016.

[36] Hastie, Trevor, et al. The elements of statistical learning: data mining, inference, and prediction. Vol. 2. New York: springer, 2009.

[37] https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-ml/ [37]

[38] : Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. "Deep learning."

MIT press, 2016

[39] Géron, Aurélien. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. " O'Reilly Media, Inc.", 2022.

[40] Aggarwal, Karan, et al. "Has the future started? The current growth of artificial intelligence, machine learning, and deep learning." Iraqi Journal for Computer Science and Mathematics 3.1 (2022): 115-123.

[41] https://www.zendesk.com/blog/machine-learning-and-deep-learning/ .

[42] N. Miailhe and C. Hodes, “The Third Age of Artificial Intelligence,” Open Edition Journal, Special Issue, vol. 17, pp. 6–11, 2017.

[43] A. N. Ramesh, C. Kambhampati, J. R. T. Monson, and P. J. Drew, “Artificial intelligence in medicine,” The Annals of The Royal College of Surgeons of England, vol. 86, pp. 334–338, 2004.

[44] N. Kühl, M. Goutier, R. Hirt, and G. Satzger, “Machine Learning in Artificial Intelligence: Towards a Common Understanding,” in Proceedings of International Conference on DSystem Sciences (HICSS-52), pp. 1–11, 2019.

[45] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman,“Building machines that learn and think like people,”Cambridge University Press,vol.40,no.E253,pp. 1–72, 2016.

[46] M. M. Mijwil, “Malware Detection in Android OS Using Machine Learning Techniques,” Data Science and Applications (DataSCI), vol. 3, no. 2, pp. 5– 9, 2020.

[47] S. N. Abd, M. Alsajri, and H. R. Ibraheem, “Rao-SVM Machine Learning Algorithm for Intrusion Detection System,” Iraqi Journal for Computer Science and Mathematics, vol. 1, no. 1, pp. 23–27, 2020.

[48] R. C. Fong, W. J. Scheirer, and D. D. Cox, “Using human brain activity to guide machine learning,” Scientific Reports, vol. 8, no. 5397, pp. 1–10, 2018.

[49] Taye, M. M. (2023). Theoretical understanding of convolutional neural network: Concepts, architectures, applications, future directions. Computation, 11(3), 52 .