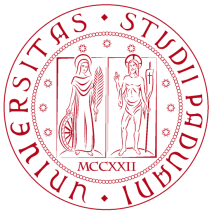


Università degli Studi di Padova

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITY OF PADOVA

DEPARTMENT OF INFORMATION ENGINEERING

Master degree in ICT for Internet and Multimedia

AI driven Generation and Classification of Short Sound Messages for Internet of Audio Things

Supervisor
Prof. Leonardo Badia

Co-supervisor
Prof. Sergio Canazza Targon

Master Candidate
Manuele Favero

Student ID:
2096408

Academic Year
2023–2024
15 October 2024

To Sofiane B., who has accompanied and supported me through countless challenges over these years.

Abstract

This thesis explores the automated generation and classification of short sound messages, referred to as TIScode, for the Internet of Audio Things (IoAuT). A TIScode is a brief audio sequence, lasting 4-5 seconds, carrying digital information that can be recognized by a specific smartphone application. The work focuses on developing methodologies for the various stages of the TIScode pipeline, including generation, transmission, and ultimately, reception and decoding. For the generation phase, MusicGen, a state-of-the-art autoregressive transformer model, is proposed, along with an analysis of its degrees of freedom to maximize the variety of distinct audio tracks that can be generated. Additionally, a channel coding system based on the quantization of sound features and certain high-level features extracted through convolutional neural networks (CNNs) is introduced. These features are mapped to create a unique bitmap for each TIScode, simplifying the decoding process. An algorithm is presented for the recognition phase, combining sound feature analysis with frequency-based peak analysis to enhance detection accuracy. Experimental results, obtained through simulations and field tests, demonstrate the effectiveness of the system in retrieving the digital information encoded within sound messages.

Contents

List of Figures

List of Tables

Listing of acronyms

AoI Age of Information

ARTL Asymmetrical Round Trip based Localization

ASV Autonomous Surface Vehicles

AUV Autonomous Underwater Vehicles

BPM Beats Per Minute

CENS Chroma Energy Normalized

CNN Convolutional Neural Network

CREPE Convolutional Representation for Pitch Estimation

CQT Constant-Q Transform

DTN Delay-Tolerant Networking

ELF Extremely Low Frequencies

FBR Focus Beam Routing

FEC Forward Error Correction

HARQ Hybrid Automatic Repeat reQuest

ISI Intersymbol Interference

IoAuT Internet of Audio Things

IoMusT Internet of Musical Things

IoS Internet of Sounds

IoT Internet of Things

IoUT Internet of Underwater Things

LAN Local Area Networks

LDB Localization with Directional Beacons

LED Light-Emitting Diode

Li-Fi Light Fidelity

LIDAR Light Detection and Ranging

LSLS Large-Scale Localization Scheme

LSTM Long Short-Term Memory

MFC Microbial Fuel Cells

MIMO Multiple Input Multiple Output

MM Musical Metaverse

NLP Natural Language Processing

OWC Optical Wireless Communication

QoE Quality of Experience

RAT Radio Access Technology

RF Radio Frequencies

RMS Root Mean Square

RNN Recurrent Neural Networks

RSS Received Signal Strength

STFT Short Time Fourier Transform

TDoA Time Difference of Arrival

TIScode Transmit In Sound Code

ToA Time of Arrival

UCL Underwater Convergence Layer

UHF Ultra High Frequencies

UPS Underwater Positioning Scheme

UWA Underwater Acoustic

UWAR Underwater Augmented Reality

V-VLC Vehicular Visible Light Communication

VHF Very High Frequencies

VLF Very Low Frequencies

VLC Visible Light Communication

VQT Variable-Q Transform

WASN Wireless Acoustic Sensor Networks

XR eXtended Reality

ZCR Zero Crossing Rate

1 Introduction

This thesis project is the result of a collaboration with Ogenus S.R.L. The idea for this work stems from the observation of how, in recent years, alternative communication methods have increasingly gained prominence over traditional Radio Frequencies. Examples include acoustic communication, particularly in underwater scenarios, and communication through visible light spectrum.

These new ways of communicated combined with the technological advancements in the field of the Internet of Things (IoT) have opened new possibilities in various sectors, including the one of the audio communication. The convergence of IoT and audio technologies has led to the emergence of innovative concepts such as the IoAuT[1]. These paradigms are characterized by the integration of devices capable of producing, analyzing, and transmitting audio data in real-time, enabling a wide range of applications.

From the fusion of these concepts and technologies arises the idea of the TIScode (Transmit In Sound Code). The TIScode is a short jingle, a melody lasting 4 to 5 seconds, capable of carrying information. At the time of its creation, a digital piece of information is assigned to it, which can only be obtained once it has been decoded by a specific application on our smartphone.

The innovation lies in the fact that it's not required to unlock the phone, turn on the recorder, activate the camera, or perform any other action. The app is able to recognize the opening marker, a small sound preceding the actual audio containing the information, allowing it to activate, record autonomously, and upon unlocking the device, access to the content of the TIScode.

This can be applied in numerous contexts, from transmitting information from the radio while we are driving, to receiving offers when in a shopping mall. It can also be useful for new types of authentication that differ from commonly used visual codes as the QR codes; there will be no need to aim the camera at the computer screen, as the entire process will be handled automatically by the TIScode system, ensuring a new layer of security for our authentication.

Research papers addressing such new topics as the practical implementation of technologies related to the Internet of Audio Things are rarely found. This provides us with the opportunity

to introduce and experiment with audio communication techniques and methods where there is still limited exploration.

In this thesis, we will study how the use of the latest advancements in artificial intelligence research can serve as a useful tool in the generative phase of our short sound messages, as well as in the classification phase, by recognizing high-level features of our TIScodes.

The primary objective of this thesis is to explore a system for encoding and decoding the TIScode. For channel encoding, we will utilize a large number of sound features that will be instrumental in characterizing the sound and describing our dataset. These features will be quantized and mapped into bit strings, which will be useful for constructing an efficient channel encoding system that allows for detection or correction of a high number of bits.

During the recognition phase, we will also leverage spectral analysis based on the identification of peaks in the frequency domain and their temporal distances. The score obtained from this technique, combined with feature analysis, will provide us with the best match candidate once we consult the database, enabling us to determine the success rate of our system in retrieving the digital information associated with the TIScode.

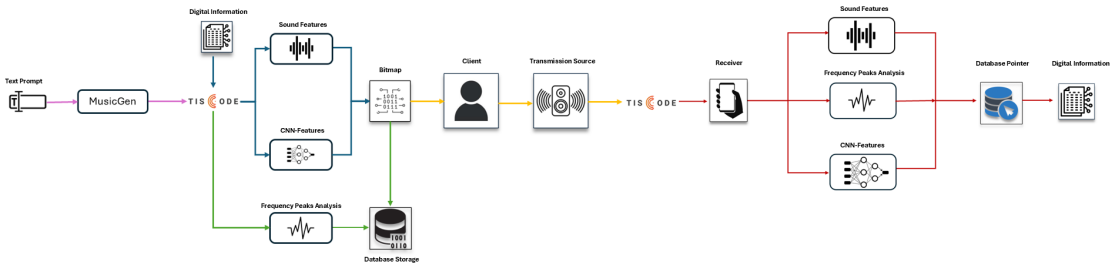


Figure 1.1: TIScode Pipeline

The structure of the thesis includes an overview of alternative communication technologies beyond radio frequencies, with a focus on audio and musical communication in Chapter 2. The role of artificial intelligence in the generation and classification of sounds is also analyzed, with a brief historical digression on the advancements made in recent decades , and a particular attention to recent transformer-based models, such as MusicGen, a Google model, used for the automatic generation of audio tracks.

In Chapter 3, we will explore the theoretical background and all the necessary information to understand the construction of our model for sound generation, as well as channel encoding and decoding. We will delve into how we built the dataset of TIScodes that we will use throughout the thesis and analyze the features employed, including both CNN-based features and traditional sound features.

Finally, in Chapter 4, we will analyze all the material at our disposal and explain step by step the reasons and methods for creating the bitmap representations of our TIScodes. By examining the Hamming distance, we will assess the effectiveness of our approach and determine the extent to which we can implement error correction. We will also create a simulation algorithm for the

recognition phase to test our system.

Additionally, we will demonstrate how we will use generalized frequency responses from certain smartphones to create simulations that are as realistic as possible. We will conclude with some field tests to showcase results related to realistic testing scenarios.

The objective of this work is to demonstrate how artificial intelligence can facilitate the management and transmission of complex audio data, thereby enhancing the efficiency of IoAuT systems. We will illustrate how sound features create a concise and effective representation of our short sound messages, and how the analysis of frequency peaks complements this approach, making it suitable for this task and fully explain every part of the TIScode pipeline (Figure 1.1).

2

Alternative Communication Technologies: Beyond Radio Frequencies

In recent decades, radio communication has dominated much of wireless interactions due to its reliability and wide range of applications, from mobile devices to satellite networks [2]. However, with the exponential increase in connectivity demand and the emergence of new environmental and technological challenges, it has become necessary to explore alternative solutions that can overcome the intrinsic limitations of radio waves, such as spectrum congestion, interference, and poor propagation in certain environments.

This section aims to explore emerging communication technologies that offer innovative solutions by leveraging different physical principles and transmission channels. These technologies, including acoustic communication, visible light communication, and audio communication, are gaining relevance in various sectors. The use of these alternative modes enables new application scenarios in specific contexts, such as underwater environments, highly reflective indoor spaces, or secure, high-performance systems [3][4].

In this chapter, related works concerning alternative methods to data transmission and telecommunications beyond traditional radio waves will be analyzed, explaining the limitations of the latter in certain environments and exploring the potential and use of alternatives. The state of the art regarding the use of artificial intelligence in music will also be examined, focusing on both classification and generation, with a brief historical digression on the various steps taken to achieve the current advancements.

Finally, a detailed explanation of TIScode, the core of this thesis, will be provided, along with how it can be an integral part of an Internet of Audio Things system and how artificial intelligence can be useful for its proper functioning.

2.1 Underwater Acoustic Communication

Underwater wireless communications present new and distinct challenges when compared to wired and wireless communications through the atmosphere, requiring sophisticated communication devices to achieve relatively low transmission rates, even over short distances [5]. Indeed, the underwater environment possesses a number of distinguishing features that make it unique and rather different from terrestrial radio propagation where traditional communication systems are deployed. Under water, several phenomena may influence communications, such as salt concentration, pressure, temperature, amount of light, winds and their effects on waves. From the physics viewpoint, for frequency ranges employed by mobile services, TV, radio, and satellite communications, the seawater is highly conductive, thus seriously affecting the propagation of electromagnetic waves. As a result, it is not easy to establish communication links for distances beyond 10 m in the ocean in both Very High Frequencies (VHF) and Ultra High Frequencies (UHF) ranges, or even in high frequencies. At lower frequencies, namely at Extremely Low Frequencies (ELF) and Very Low Frequencies (VLF), the electromagnetic-wave attenuation can be considered low enough to allow for reliable communications over several kilometers. Unfortunately, these frequency ranges from 3 Hz to 3 kHz and from 3 kHz to 30 kHz are not wide enough to enable transmissions at high data rates. The underwater RF communication is also heavily affected by propagation loss. Due to these problems technologies possess severe constraints on data rates and on propagation distances. These are the main reasons for the small number of products using RF communication technology so far [6].

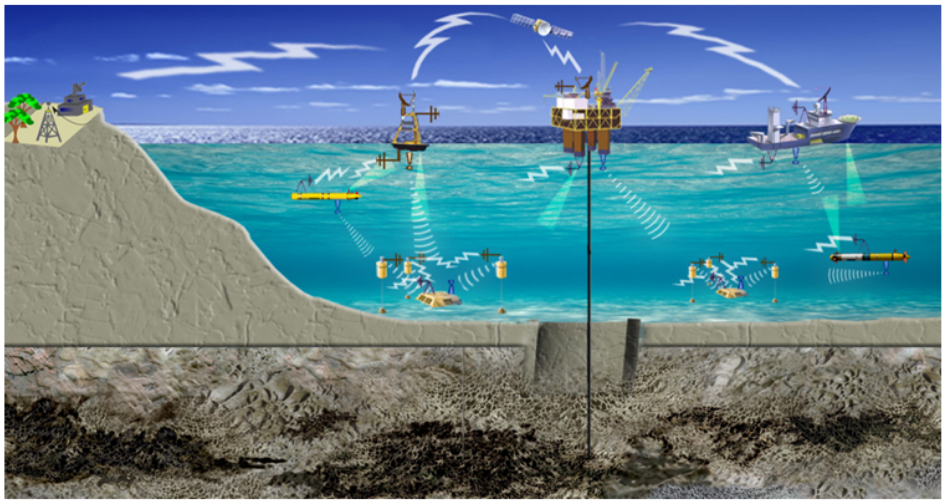


Figure 2.1: Underwater Communication Scenario [6]

The need for underwater wireless communications however exists in applications such as remote control in off-shore oil industry, pollution monitoring in environmental systems, collection of scientific data recorded at ocean-bottom stations and unmanned underwater vehicles, speech transmission between divers and mapping of the ocean floor for detection of objects and discovery

of new resources as can be seen in Figure 2.1 [3]. Underwater Acoustic (UWA) communications and networks have recently received considerable attention from the research community, since they make it possible to wirelessly convey information from under water to the surface, thus enabling several ocean-related applications, in particular for sensing and monitoring of seismic activity, oil leakages, chemical pollutions, and so on [7]. Radio communication uses electromagnetic waves (radio waves) for transmission. Electromagnetic waves do not require a physical medium and can propagate even in a vacuum, such as in space, while Acoustic communication uses sound waves to transmit information. Sound waves are mechanical vibrations that propagate through a physical medium such as air, water, or soil.
Compared to Radio Frequencies (RF), which suffer from high attenuation and dependency on water turbidity, respectively, acoustic communication offers longer range because acoustic waves experience less attenuation in water compared to radio waves or light. This makes it ideal for applications such as long-range ocean monitoring, submarine communication, and underwater search and recovery operations. Furthermore UWA systems can be more cost-effective to implement than other underwater wireless communication technologies, such as optical systems, which require more sophisticated and expensive equipment. However, acoustic communication is characterized by low data transmission speeds, significant Doppler effects, and high delay spread, leading to severe Intersymbol Interference (ISI). The low propagation speed of acoustic waves in water, which is several orders of magnitude lower than that of electromagnetic waves in air, significantly contributes to the Doppler effect, increased delay spread, and reduced temporal coherence of the channel. These challenges require sophisticated signal processing techniques on the receiver side to ensure reliable communication [3].

In [7] B. Tomasi et al. propose and evaluate the use of Hybrid Automatic Repeat reQuest (HARQ) schemes to address the high error rates and time-varying channel conditions typical of underwater acoustic communication. HARQ combines Forward Error Correction (FEC) with retransmissions to dynamically adapt the code redundancy based on the channel's needs [5]. To account for the temporal correlation of underwater acoustic channel conditions, the article proposes the use of a Markov statistical model. This model represents quantized channel states and their transition probabilities, capturing the time-varying dynamics of the channel. The performance of HARQ schemes is evaluated using both simulations and theoretical analysis based on the Markov model. The results show that HARQ significantly improves throughput and reliability, especially in challenging channel conditions. The Age of Information (AoI) is another crucial factor to take account of in underwater communication scenarios, where the timeliness of received information is essential for real-time monitoring and control. Given the unstable and delayed nature of underwater acoustic channels, optimizing AoI ensures that the most recent data is used for critical decisions [8][9].

There is also a need for energy-efficient routing in these challenging environments due to bandwidth limitations and energy constraints of underwater nodes and this problem is faced by J. Jornet et al. in [10]. The idea of this work is, instead of transmitting over long distances, which requires high transmission power, introducing a multi-hop routing that divides the transmission into shorter hops. This approach reduces overall energy consumption, as transmission power is a

function of distance. The paper also references previous work demonstrating that the capacity of an acoustic relay link increases with the number of hops used to cover a given distance, particularly in noise- and interference-limited scenarios. The paper proposes Focus Beam Routing (FBR), a dynamic, location-aware routing protocol designed for underwater acoustic networks. In FBR, nodes are assumed to know their own positions, which can be obtained through underwater positioning systems or pre-existing information for static nodes. However, nodes do not require knowledge of other nodes' positions, reducing the signaling overhead and complexity associated with global routing information discovery and maintenance. The paper evaluates FBR's performance through simulations, comparing it to pre-established path routing using Dijkstra's algorithm for minimum energy consumption. The results demonstrate that FBR achieves energy consumption per bit and average end-to-end packet delay performance close to the pre-established path approach, highlighting its effectiveness in dynamically discovering energy-efficient paths with minimal network knowledge.

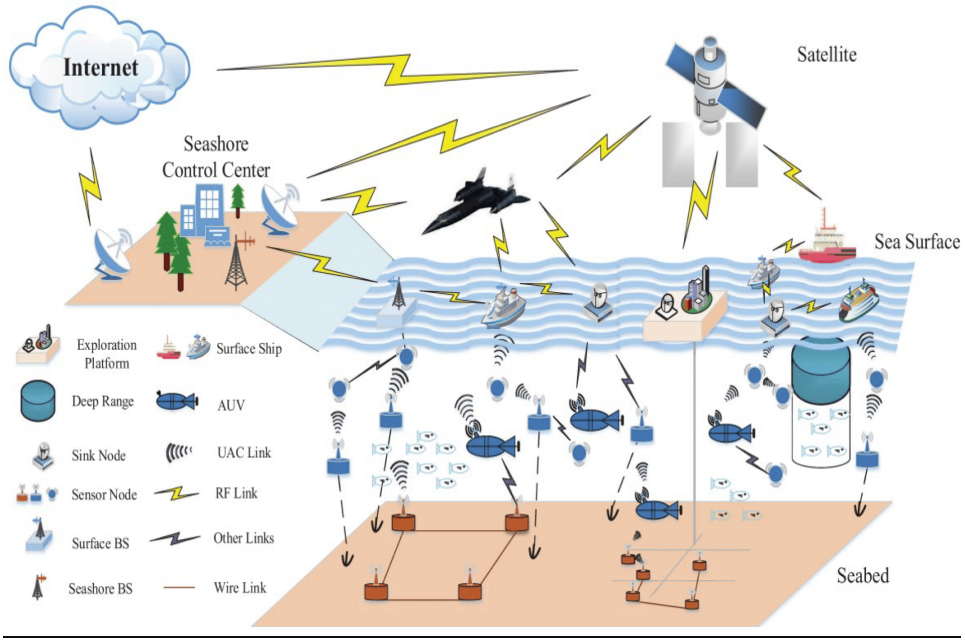


Figure 2.2: Internet of Underwater Thing Scenario [11]

2.1.1 Internet of Underwater Things

Unlike terrestrial IoT that primarily uses radio waves, the Internet of Underwater Things (IoUT) mainly relies on acoustic links due to the poor propagation of radio waves underwater. However, acoustic links present challenges such as high propagation delays, limited bandwidth, and susceptibility to noise, turbulence, and varying channel conditions. Recharging or replacing batteries for underwater objects is complex due to biofouling, corrosion, and inaccessibility. To address this issue, environmental energy harvesting combined with supercapacitors is promising for ex-

tending battery life. IoUT can utilize unique energy harvesting techniques such as solar energy for surface vehicles and ocean thermal energy. Other methods include underwater wireless power transmission or piezoelectric energy harvesting from water flows and Microbial Fuel Cells (MFC) that generate energy from bacterial activity [12].

Compared to the high density of devices in IoT, the IoUT tends to have a sparser network due to high costs and challenges associated with underwater deployments. This sparse nature makes it difficult to establish and maintain communication between IoUT devices. The absence of GPS signals underwater requires specialized localization techniques for the IoUT. Terrestrial methods such as Received Signal Strength (RSS), Time of Arrival (ToA), and Time Difference of Arrival (TDoA) face challenges due to propagation delays, the Doppler effect, and timing synchronization. Alternative localization techniques specifically designed for underwater wireless sensor networks have been proposed, such as Localization with Directional Beacons (LDB), Underwater Positioning Scheme (UPS), Large-Scale Localization Scheme (LSLS), and Asymmetrical Round Trip based Localization (ARTL) [13].

M. Domingo in [14] proposed an IoUT architecture consisting of three layers:

1. *Perception layer*: This layer includes underwater sensors, vehicles, surface stations (sinks), monitoring stations, data storage tags, tags, and hydrophones that collect and identify information from the underwater environment.

2. *Network layer*: This layer facilitates the transmission of data collected from the perception layer using various networking technologies, including wired/wireless networks, the Internet, cloud computing platforms, satellite communications, and heterogeneous networks (Bluetooth, ZigBee, WLAN, WiMAX). It addresses challenges of intermittent connectivity, limited bandwidth, and protocol interoperability in underwater communications. Technologies such as Delay-Tolerant Networking (DTN) and Underwater Convergence Layer (UCL) are discussed to enable reliable communication in underwater environments.

3. *Application layer*: This layer provides smart solutions based on IoUT data to meet user needs. A RESTful architecture (Representational State Transfer) is described for representing and accessing data from IoUT devices, allowing human operators to interact with and control underwater objects via the web. Various application servers, such as the acoustic server, radio server, RFID server, and monitoring server, are discussed to process and store collected data for various purposes.

IoUT devices have very different applications scenario, they can be used in aquariums where acoustic service tags on fish can provide visitors with real-time information, enhancing the educational experience and allowing caretakers to monitor the health of individual fish. In fish farms tags could help track fish, manage health records, and monitor water quality to ensure food safety and operational efficiency. Underwater sensors and Autonomous Underwater Vehicles (AUV) can collaborate to inspect pipelines, detect leaks, and monitor structural conditions. Multimedia acoustic sensor networks enable the acquisition and analysis of images and videos for disaster prevention and support during inspection activities [15]. Underwater Augmented Reality (UWAR) assists

divers by providing visual information, enhancing situational awareness and safety during repair operations. Underwater sensor networks, AUVs, and surface vehicles can also work together to enhance port security by detecting underwater intrusions, monitoring shipping traffic, and identifying suspicious activities. Sensors such as Light Detection and Ranging (LIDAR) mounted on Autonomous Surface Vehicles (ASV) help detect explosives or illegal objects on vessels [16]. Underwater laser imaging networks and multimedia content from acoustic sensor networks provide valuable visual information for coastal monitoring and tactical surveillance.[14].

IoUT is a growing technology and so standardization efforts are needed to address interoperability among heterogeneous underwater systems. The development of common protocols, gateways for protocol conversion, and a unified IP-based network architecture are crucial for seamless integration and communication. Addressing security vulnerabilities specific to aquatic environments, such as high bit error rates, variable propagation delays, and limited bandwidth, is critical. Robust mechanisms for authentication, encryption, and intrusion detection are essential to ensure data security and system integrity [17].

2.2 Visible Light Communication

Visible Light Communication (VLC) is an emerging wireless communication technology that uses visible light (wavelengths between 400 and 700 nm) to transmit data. Unlike conventional RF systems, VLC leverages Light-Emitting Diode (LED) as transmitters, exploiting their ability to rapidly switch on and off to convey information. The communication process is imperceptible to the human eye but can reach high data rates due to the fast switching speeds of LED. This technology is highly energy-efficient, as it utilizes the existing lighting infrastructure, making it a promising candidate for energy-saving and sustainable communication systems. VLC offers several advantages over RF communication, such as abundant bandwidth, immunity to electromagnetic interference, and enhanced security, as visible light does not penetrate walls, reducing the risk of eavesdropping. This technology is particularly well-suited for indoor applications like smart lighting, data communication in offices, and even vehicle-to-vehicle communication, where headlights and taillights could transmit information. However, VLC faces challenges such as signal interference from ambient light sources, line-of-sight restrictions, and the need for constant illumination. VLC is part of the broader field of Optical Wireless Communication (OWC), which also includes infrared and ultraviolet communication.

The sources for a VLC link are usually white-light LED that either use red, green and blue LED mix to provide the desired colour, or a single LED (usually blue) that excites a yellow phosphor to create an overall white emission. The 'triplet' approach allows the colour to be altered by varying the colour to the LEDs, and also allows different data to be sent on each device. However, maintaining colour balance can be challenging and the devices are complex. The bandwidth is around 2 MHz for the white and around 20 MHz for the blue component only. This is due to the long decay time of the phosphor and provides a limitation on the overall bandwidth available. In addition the Blue LED die is not designed for high speed operation and is very large in area (and thus has high equivalent capacitance) compared with devices used for high speed

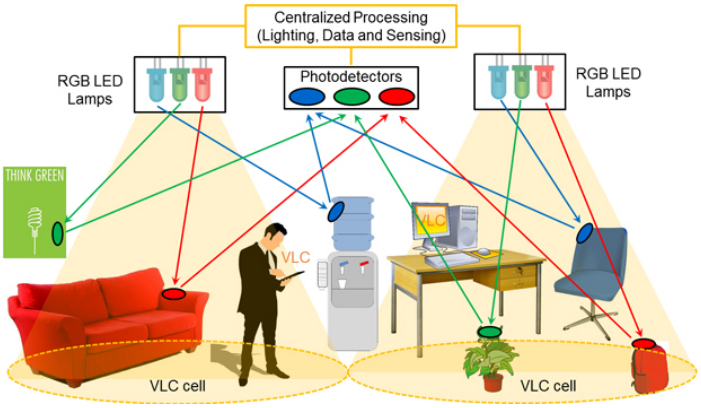


Figure 2.3: Visible Light Communication Scenario[18]

communications. The VLC channel consists of a number of line of sight paths from the units to the terminal together with a diffuse channel formed by the light from the source reflecting of the multiple surfaces within the room. These two channels can be modelled separately and combined to obtain the overall power received at the terminal and the bandwidth of the channel. The receiver consists of an optical element to collect and concentrate the radiation onto the receiver photodetector. This converts the radiation into photocurrent which is then pre and post-amplified before data recovery. The optical element usually either a lens or non imaging concentrator has a maximum "gain limited by constant radiance considerations so the photodetector area should be as large as possible given the required receiver bandwidth. Achieving sufficient photodetector area at the LED constrained system bandwidth is relatively straightforward, so the receiver does not provide a significant constraint [19].

Possible VLC applications are described by L.u. Khan et al. in [20]:

- Underwater Communications: Given the poor propagation of radio waves in water, VLC presents a viable alternative for underwater communications, such as controlling AUV.
- Hospitals: In healthcare environments, where electromagnetic waves can interfere with sensitive equipment like magnetic resonance imaging scanners, VLC offers a safe and reliable alternative.
- Digital Signage: LED displays, already present in airports, stations, and other public places, can be utilized to transmit real-time information through VLC.
- Identification Systems: Visible light can be employed to create identification systems within buildings, for example, to indicate room numbers or provide information about the structure.
- Local Area Networks (LAN): VLC can be used to create high-speed local area networks, as demonstrated by a prototype that offers speeds exceeding 10 Gbps.

VLC can be exploited also in the context of vehicular applications, known as Vehicular Visible Light Communication (V-VLC). Transportation systems are undergoing significant technological

transformation, with vehicles becoming more intelligent and connected and in this context, V-VLC emerges as a complementary technology to radio-frequency-based communications, capable of enhancing safety, efficiency, and driving comfort. Key advantages of V-VLC include the wide availability of spectrum in the visible range, the use of LED lighting systems already present in modern cars, and reduced interference thanks to the directional nature of light signals. Despite its potential, V-VLC presents unique challenges, particularly regarding the need for a clear line of sight between transmitter and receiver, which can be difficult to maintain in real-world driving scenarios. Factors influencing V-VLC communications are light distribution from lighting modules, the presence of obstacles, vehicle mobility, ambient light, and weather conditions [21].

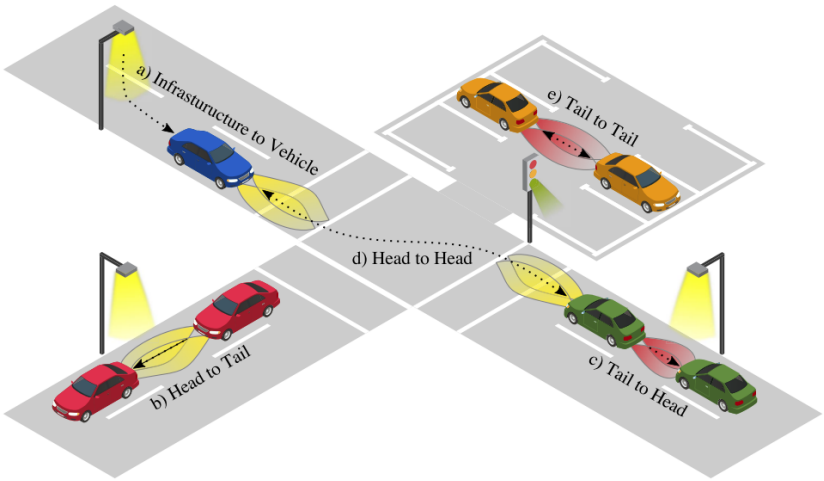


Figure 2.4: Vehicle Visible Light Communication Scenario[22]

In Figure 2.4 Vehicular VLC communication patterns and potential applications can be seen.

a) *Infrastructure to vehicle communication* can be used for information exchange between VLC-enabled vehicles and intelligent traffic lights, traffic signs, or road lighting. Potential applications include intersection assistance.

Communication patterns between vehicles include:

b) Head to tail communication

c) Tail to head communication

They can be used for emergency electronic brake light and platooning applications, respectively.

d) *Head to head communication* is possible if vehicles face each other. It can be used for intersection coordination and maneuver coordination between vehicles.

e) *Tail to tail communication* is possible if two vehicles are opposed to each other, plausible in parking situation. Note that, bidirectional communication is possible in all vehicle to vehicle cases.

2.2.1 Li-Fi for Internet of Things

VLC applications often used Li-Fi that is a communication technology that utilizes visible light to transmit data. The principle behind Li-Fi involves modulating the light emitted by LED sources to convey information. By varying the intensity of the light at a speed imperceptible to the human eye, data can be encoded and sent effectively. One of the remarkable features of Li-Fi is its potential to offer extremely high data transmission speeds, often exceeding those of Wi-Fi. Experimental implementations have achieved speeds reaching several gigabits per second. This technology has a range of applications, including public lighting, aviation, educational institutions, and healthcare settings, where radio waves may cause interference or are not viable for use. In terms of security, Li-Fi presents advantages as well. Since light cannot penetrate walls, it offers a greater level of security compared to Wi-Fi, significantly reducing the risk of external interception. However, there are some limitations to consider; Li-Fi requires a direct line of sight between the transmitter and the receiver, and its performance can be affected by environmental conditions such as obstacles or variations in brightness [23].

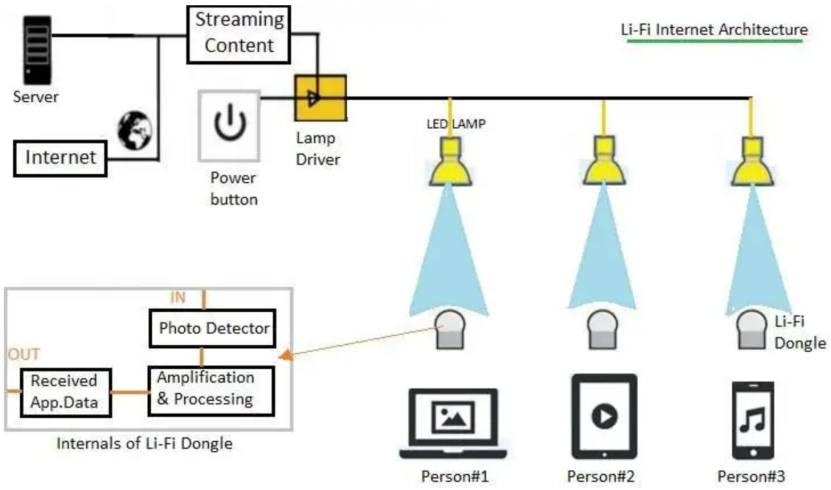


Figure 2.5: Li-Fi for Internet of Things application [24]

In [25] A. Petrosino et al. explore the use of Li-Fi for Internet of Things, proposing its use in:

- *E-Health:* Li-Fi can support the secure and reliable transmission of health data in real-time, for instance, for patient monitoring and telemedicine.
- *Industrial Environments:* In smart factories, Li-Fi can ensure high-speed and low-latency communications for controlling robots and other industrial devices.
- *Home Automation:* Li-Fi can be used to create secure and energy-efficient smart home networks, controlling lighting, appliances, and other devices.
- *Public Transport:* Li-Fi can provide real-time information to public transport passengers, such as arrival times and traffic conditions.

- *Hostile Environments:* Li-Fi can be utilized in environments where RF waves are discouraged or prohibited, such as mines, power plants, and hospitals.

Although promising, Li-Fi technology still faces several challenges before it can be widely adopted within the IoT ecosystem. The creation of Li-Fi networks for IoT, based on a small cell architecture with a high density of LED, requires managing optical interference between cells, a complex issue in the context of light transmission. Additionally, handling the handover of IoT nodes in indoor scenarios with mobile devices and multiple access points requires further research to ensure service continuity and maximize system efficiency, especially in dense environments with real-time data exchange. Accurate assessment of throughput performance in IoT systems with interference management strategies, such as Multiple Input Multiple Output (MIMO), represents another crucial area of research. While some studies propose MIMO to enhance throughput and energy efficiency, the effectiveness of this approach, borrowed from RF systems, needs further validation, particularly through large-scale testing [25].

2.3 Audio and Music Communication

Audio and sound have long been essential means of communication, facilitating not only human interaction but also enabling sophisticated machine-to-machine interactions in various fields. With the rapid evolution of technologies such as the IoT, the role of sound in digital communication has expanded into new paradigms. This shift is most prominently seen in the rise of the Internet of Sounds (IoS) and the IoAuT, two overlapping fields that highlight the convergence of sound, audio processing, and IoT technologies.

The Internet of Sounds encompasses systems where interconnected devices, termed Sound Things, can sense, process, and exchange sound data to facilitate audio-related services globally. This paradigm merges musical and non-musical applications, connecting various stakeholders from entertainment to healthcare and environmental monitoring [26].

Meanwhile, the Internet of Audio Things focuses on the creation of networks of Audio Things—devices specifically designed to handle audio production, analysis, and reception. These devices are embedded in physical objects and communicate through complex infrastructures to enable sound-based interactions in both real-time and distributed environments. This vision promotes novel applications in smart homes, smart cities, and beyond [1].

The applications of the IoAuT are varied and interdisciplinary. IoAuT is positioned at the intersection of the IoT, sound and music processing, artificial intelligence, and human-computer interaction. Essentially, it consists of networks of computing devices embedded in physical objects (Audio Things) dedicated to the production, reception, analysis, and understanding of audio in distributed environments. These "Audio Things" are connected by an infrastructure that enables multidirectional communication, both locally and remotely, with the aim of promoting and facilitating audio-based services and applications globally .

One of the most significant examples of IoAuT is Wireless Acoustic Sensor Networks (WASN), used for acoustic monitoring and acoustic scene analysis in contexts such as urban noise pollution

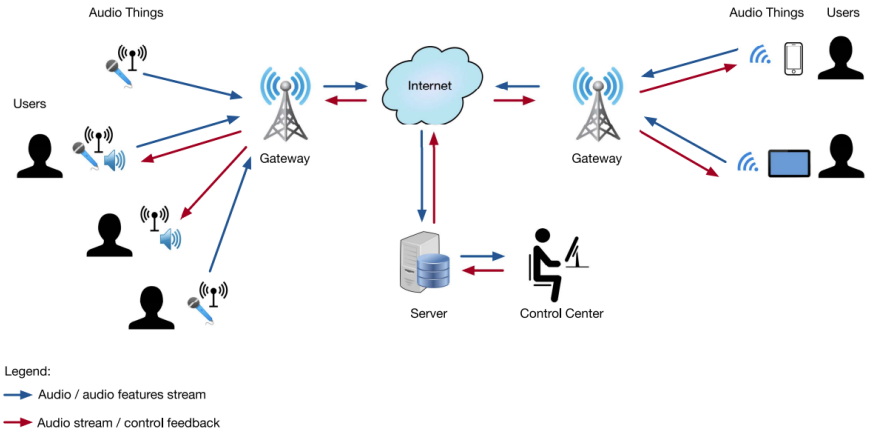


Figure 2.6: Schematic representation of an architecture supporting IoAuT ecosystems [1]

monitoring, environmental surveillance, anomaly detection, and wildlife monitoring. Another application is the so called "sonification", where data is transformed into sound to provide feedback on electricity consumption or to monitor production processes.

Semantic audio technologies enable the extraction of structured meaningful information from audio, enabling communication and interoperability between heterogeneous Audio Things. Finally, there are also web-based digital audio applications in IoAuT. The Web Audio API, for instance, enables real-time audio processing in web browsers, opening new possibilities for distributed musical performances and bodily interactions with sounds and online repositories [27]. Although IoAuT is still a relatively young field, its potential applications are vast and promising.

Another field of study related to sound is Internet of Musical Things (IoMusT). IoMusT primarily focuses on musical applications, such as live performances, music pedagogy, and studio productions. Even if it seems similar to IoAuT, it differs from it in several key aspects. IoMusT focuses on musical and creative aspects, whereas IoAuT focuses on the processing and transmission of general audio information. Furthermore IoMusT is inherently multisensory and incorporates elements such as haptic feedback and virtual reality, while IoAuT focuses exclusively on the audio signal.

2.3.1 Musical Metaverse

As technologies like IoT and IoMusT continue to evolve, the integration of interconnected devices and immersive virtual environments becomes increasingly relevant. The metaverse represents a natural extension of these advancements, enabling more complex, multisensory musical interactions that blend the physical and digital worlds. In [28] L.Turchet defines Musical Musical Metaverse (MM) as an interoperable and persistent network of multi-user environments that blend physical and digital realities for musical purposes. The MM is described as the result of the convergence between Musical eXtended Reality (XR) and IoMusT (Internet of Musical Things), which

enable multisensory and networked musical interactions between various actors in the musical world, such as musicians, audiences, record labels, and others.

Musical Metaverse consist of:

- *Physical Layer:* Collects real-world data through Musical Things, such as smart musical instruments, wearable devices, and environmental sensors. This data is then transmitted to the other layers.
- *Link Layer:* Acts as a bridge between the physical and virtual layers, managing communication, data integration, processing, and synchronization between the two worlds.
- *Virtual Layer:* Provides the immersive experience through avatars, virtual musical environments, and digital musical goods and services.

A critical issue for the Musical Metaverse is the need to ensure continuous cooperation between the different actors in order to provide a real and continuative multisensory experience. Users seek immersive engagement in the Metaverse, and real time communication and fluidity are critical parameters for assessing the Quality of Experience (QoE). To optimize QoE, it is essential to tailor the design of the resource allocation scheme based on the diverse interests of users [29].

The metaverse can lead to the creation of new musical activities, such as immersive virtual concerts, remote music learning, real-time collaborative music creation, and new ways to interact with music and artists. It can also foster new musical communities, reducing the need for travel to rehearsals and concerts through remote musical experiences. Finally, it offers easier access to musical experiences for people with disabilities.

2.3.2 Artificial Intelligence for Sound and Music Communication

The idea of using sounds and music for communication and as a medium for transporting information, as we have discussed so far, faces the limitation of how sounds and music are generated and classified.

Music generation is a task that have been studied for decades and that has attracted the interest of many researchers. One of the first examples of automated music generation, was created by L. Hiller and L. Isaacson in 1979. It used rule-based algorithms to generate music from predefined sequences. In this early phase, the approach was primarily based on rigid rules and formal logic, simulating music composition through sequential processing [30].

During 90's Markov models were introduced to generate musical sequences based on probabilities. These models could predict the next note based on previous ones, creating stochastic predictions that improved the variability of the generated music [31].

Several systems were built to generate musics, among the most important there are: Melomics, a system capable of generating complete musical compositions using evolutionary algorithms and the one developed by D. Cope that was capable of analyzing existing compositions and creating new works in the same style. The algorithm used pattern matching to fragment and recombine musical phrases [32][33].

With the advent of machine learning the focus shifted from using rigid rules to creating supervised learning models, where algorithms learned the characteristics of music through labeled datasets [34].

Later, in the second half of the 10's, with the development of deep learning we had three famous software:

- *Magenta* (2016): launched by Googlie, it is an open-source project to explore the role of machine learning in art and music. Magenta introduced Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) models to generate music based on data sequences [35].
- *WaveNet* (2016): Another major development by Google DeepMind, WaveNet used convolutional neural networks to synthesize audio, significantly improving sound quality compared to previous models [36].
- *OpenAI MuseNet* (2019): A major breakthrough using transformers to generate music in multiple styles. The model demonstrated the ability to learn long-term musical relationships, producing complex compositions [37].

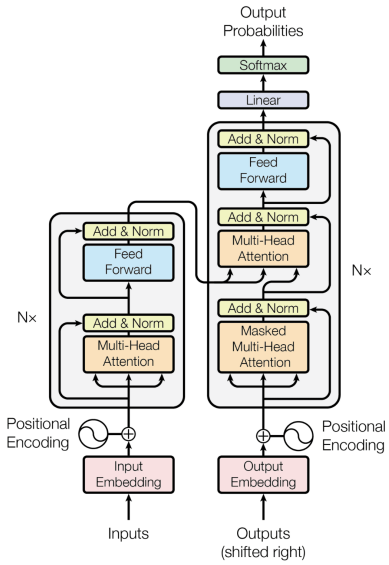


Figure 2.7: Transformers Architecture

Deep learning has significantly transformed also music classification by improving the accuracy and efficiency of identifying musical genres, instruments, and other audio characteristics. Traditional methods relied heavily on handcrafted features, which were limited in their ability to capture the complexities of music. With deep learning, especially using Convolutional Neural Network (CNN)[38] and RNNs[39], models can automatically learn features directly from raw audio data, enabling more nuanced and sophisticated classification. For example, CNNs have been particularly effective in analyzing spectrograms, which represent audio as images, to detect patterns

related to different genres or instruments. RNNs, especially those with LSTM[40] units, excel in capturing the temporal dependencies in music, making them ideal for tasks such as recognizing the structure of a piece or its rhythm [41]. Additionally, transformer-based models have started to show promise in further enhancing music classification by learning long-range dependencies across musical compositions. Deep learning models also allow for multi-label classification, which means they can identify multiple attributes (e.g., genre, instruments, mood) simultaneously. This has led to advancements in music recommendation systems, automatic playlist generation, and enhanced music search capabilities, all based on deep, data-driven analysis of musical content [42].

Now, the contemporary generation sound models are Transformer-Based Models. Transformers are a type of deep learning model introduced in 2017 that have revolutionized the field of Natural Language Processing (NLP) and are now applied to a wide range of tasks, including music generation. Unlike traditional models that process data sequentially, transformers use an attention mechanism to capture relationships between elements in a sequence, regardless of their position (Figure 2.7). This allows transformers to handle long-range dependencies more efficiently. They are composed of layers of self-attention and feed-forward networks, making them highly parallelizable and scalable for large datasets [43]. MusicGen is a Google model that represents one of the most recent advancements. Based on transformer models, MusicGen can generate complex and diverse music from textual input, allowing the creation of coherent, high-quality musical pieces as we will see later in Section 2.3.2.

2.4 TIScode

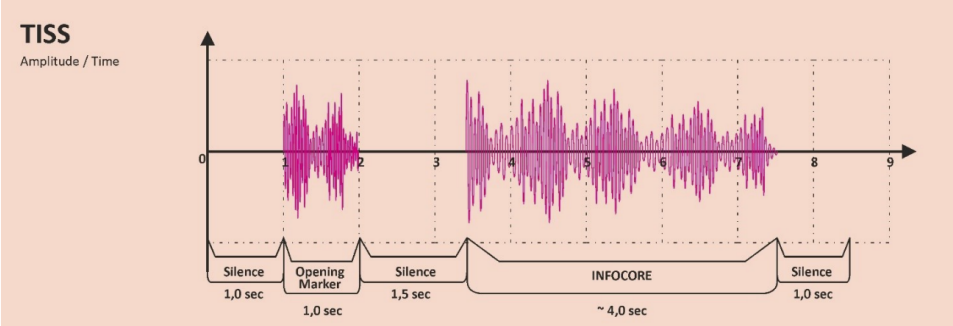


Figure 2.8: TISS representation and description [44]

A system that utilizes sound as a medium for communication and information transmission, where the sound is generated and classified with the assistance of artificial intelligence, is the TIScode [44]. Transmit In Sound Code (TIScode) is an innovative information transmission system, conceived and developed by OGENUS S.R.L, designed to send informational references via sound signals, which can be decoded by mobile devices such as smartphones or tablets. This technology, which falls within the scope of IoAuT, aims to simplify access to data or initiate interaction processes through the use of encoded sounds. These signals can contain binary data or bit strings destined for specific software applications.

The system is composed of several parts and operational modules described in [44]:

- *TIScode PRO:* This is a central platform that manages the service, integrated on a remote server. It handles the encoding of information and the generation of the sound signal called TISS (TIS Sound Signal). The TISS is the sound carrier that contains the encoded data.
- *TIScode App:* A software application installed on mobile devices, allowing end-users to receive the TISS through the device's microphone, decode it, and display the received information. The app performs various signal detection and interpretation steps, utilizing technologies like neural networks to manage noise and signal distortions.
- *InfoTIS:* The transmitted information, which can be a file, a bit string, or any other type of informational reference. The INFOTIS is stored in the TIStore, a digital archive, and is associated with a unique address identified as CNR (Code Number Reference).

The TISS is the sound signal (but also infra- or ultra-) at the output of the Encoder. Provided as an audio file in suitable digital formats, it represents the sound vector of the TIScode system. The term indicates both the digital file and its physical expression, when it is broadcast [44].

TISS contains two main sections (as can be seen in Figure 2.8):

- *Opening Marker:* An initial signal that activates the recording and decoding process.

- *INFOCORE:* The core of the signal that contains the encoded data. It is the part of the TISS that contains the coded information associated with the CNR, which is the address that uniquely identifies the memory where the InfoTIS is allocated. This is the fundamental data processed and encoded in the TISS.

Encoding and Decoding techniques will be explained in Chapter 3. The encoding will be performed through sound features analysis and quantization in order to create a useful bitmap for channel coding. Decoding will also implement frequency and peak analysis to enhance the detecting ratio of the sound messages.

(a) Real-life TIScode application for spreading promotions and offers

(b) Real-life TIScode application for secure authentication

Figure 2.9: Real-life TIScode applications

TIScode has a wide range of applications, mainly related to the fast and minimally invasive transmission of information [44]:

- *Public Broadcasting:* Used in public environments such as shopping malls or radio broadcasts to distribute information to multiple users simultaneously.
- *Personal Communication:* Transmitting data to individuals via personal devices in situations where other transmission methods (such as Wi-Fi or Bluetooth) may not be available.
- *Accessibility:* A system designed to facilitate information access for people with visual or motor disabilities through easily recognizable and decodable sound signals.

In Figure 2.9, we can see two real-life applications of the TIScode. In Figure 2.9a, there is a representation of a person at the supermarket receiving offers and promotions related to the products on sale, transmitted via a TIScode played through the market's speakers. In Figure 2.9b, a person authenticates themselves using a TIScode played by their own laptop speaker. Both actions do not require unlocking the phone or activating an application. The TIScode PRO app

first recognizes the opening marker and then the InfoCORE, allowing users to discover promotions the next time they unlock their phone or perform two-factor authentication without needing to enter anything or use the camera.

The innovation of TIScode lies in its ability to leverage an underutilized transmission channel (sound) to transmit synthesized codes that can be decoded quickly and efficiently.

Since this work mainly focuses on the generation and classification of the "InfoCORE", in the next chapters for a clearer understanding, the InfoCORE will be referred to with the name of TIScode, which would actually be the whole system.

3

Model Setup

In this chapter, the theoretical background will be explored in depth, and all the necessary concepts will be explained to understand how the transmission and recognition system was built, with the aim of better grasping its strengths and limitations in the following chapter.

In Section 3.1, it will be explained how TIScodes samples are generated in this project and why this system ensures that we can achieve a high number of TIScodes that are as diverse as possible. In Sections 3.2 and 3.3 we will analyze the features and properties selected for the characterization of the TIScodes, explaining how they can be extracted and their characteristics.

3.1 Sound Generation

The TIScode project aims to manage an extremely high number of short sound messages. Manually generating each sound is not feasible and may not be an adequate technique to ensure sufficient diversity. A significant variation among the sounds is necessary, especially considering the short duration of TIScodes. This brief duration presents the drawback of limited evolution, with few elements available to create a wide variety.

The diversity, or degrees of freedom, at our disposal are essential for producing distinctly different sounds. Such sounds are easily distinguishable from one another, as will be discussed in the following sections.

To maximize the degrees of freedom we can manage, it is pivotal to have a highly performant generator that offers various functions for our purposes. This is why Google's MusicGen has been selected to accomplish this task: to generate TIScodes samples and conduct tests within this thesis.

3.1.1 MusicGen

MusicGen[45] is an innovative language model capable of generating high-quality music from textual descriptions and melodic cues. By leveraging a token system that represents discrete musical units, MusicGen can model complex musical structures and generate results that align with user requests. The architecture, based on a single transformer and various codebook interleaving schemes, makes it an efficient and versatile model. This subsection is a summary to explain briefly, what is, how work and why MusicGen is suitable for the TIScode task, all the content is described more accurately in the work of J. Copet et al. that can be found at [45].

Dataset: MusicGen has been trained on a dataset of 20,000 hours of licensed music, comprising 10,000 high-quality music tracks, 25,000 tracks from ShutterStock, and 365,000 tracks from Pond5, all consisting of complete music tracks at 32 kHz with metadata such as textual description, genre, BPM, and tags. Music tracks were reduced to mono, unless specified otherwise. The textual metadata was used for model conditioning, and various strategies for textual data augmentation were implemented, such as adding additional metadata to the textual description and applying word dropout. or the main evaluation and comparison with previous works, MUSICGEN was evaluated on the MusicCaps benchmark. This dataset includes 5,500 samples of 10 seconds prepared by expert musicians and a balanced subset of 1,000 samples across different musical genres.

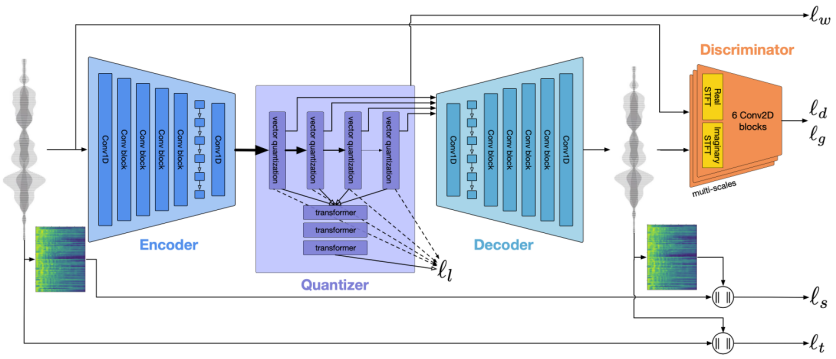


Figure 3.1: EnCodec Architecture [46]

Codebook Creation: Codebooks are essentially "sound dictionaries" learned by the model. Each codebook contains a set of discrete tokens representing different aspects of the audio signal. Codebooks are used to quantize continuous audio representations into discrete tokens. This process, known as "Residual Vector Quantization" (RVQ), is used to compress the audio signal and make it more efficient for the model. An EnCodec model, a convolutional auto-encoder with a latent space quantized,(which architecture can be seen in Figure 3.1) is used to generate various parallel sequences of discrete tokens representing the same audio signal. Each token sequence comes from a different codebook, encoding specific information such as pitch, harmonics, or timbres.

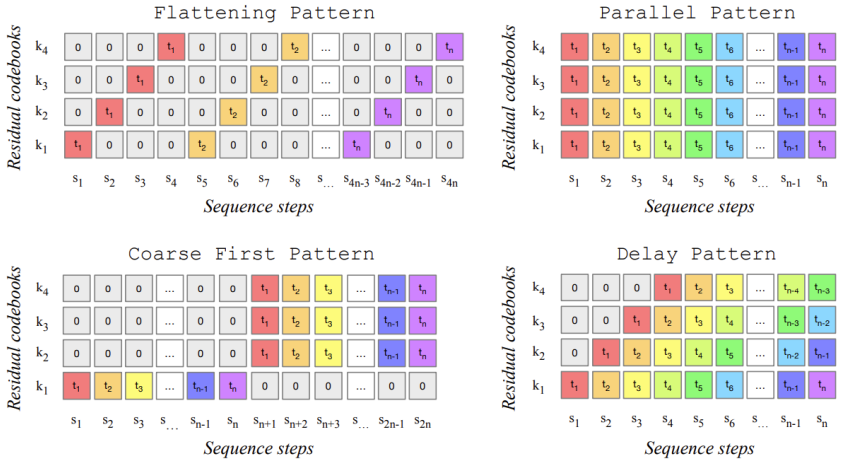


Figure 3.2: MusicGen's Codebook[45]

Model Architecture:

- *Autoregressive Transformer*: MusicGen is based on an autoregressive decoder [47] built on a transformer[43], conditioned by text or a melodic representation.

- *Token-Based Decoding*: the model's input is a sequence of discrete audio tokens, interleaved according to a specific pattern. The transformer processes this token sequence, learning the relationships between the tokens and generating predictions for subsequent tokens.

- *Codebook Interleaving Schemes*: these are essential for managing the different audio token streams. These schemes define how tokens from different codebooks are interleaved and fed into the model at each prediction stage. There are various schemes as "parallel," "delay," and "flattening" (Figure 3.2)

- *Textual Conditioning:* A pre-trained text encoder (T5, FLAN-T5, or CLAP) is used to generate a conditioning tensor from the input text. This tensor is then used to guide the music generation.

- *Melodic Conditioning:* A new unsupervised melodic conditioning method uses audio Chromagrams (see Subsection 3.2.1 for more detailed explanation) to control the harmonic and melodic structure of the generated music. This method is designed to avoid the need for manually annotated data and to enhance control over the generated output.

- *Codebook Projection and Positional Embedding:* each token is projected into a vector space via a learned embedding table, and a sinusoidal embedding is added to encode its position in the sequence.

- *Transformer Decoder:* the transformer decoder consists of several layers, each composed of a causal self-attention block and a cross-attention block that processes the conditioning signal [43].
- *Logit Prediction:* The output of the transformer decoder is transformed into probability predictions (logits) for the subsequent tokens in each codebook.

3.1.2 TIScode Dataset

The dataset generated with MusicGen and that will be used for all tests conducted in this thesis consists of 1,000 audio tracks. These tracks were generated with a maximum token limit of 512 per track, which translates to a duration of about 5 seconds.

One hundred of these tracks were generated using the "unconditional generation" function, which involves MusicGen generating random pieces without any input. These were created both to test this type of generation and to introduce an even more random variable into the dataset to stress the tests that will be described later.

The remaining 900 tracks were generated using 900 text AI-generated prompts. The prompts are structured like the examples below:

- "Indie pop track with upbeat melodies and jangly guitars",
- "Reggae track with laid-back rhythms and deep basslines",
- "Blues rock song with gritty vocals and soulful guitar solos",
- "Synthwave track with nostalgic 80s synths and driving beats"

It is important to analyze how many degrees of freedom we have in order to, as mentioned earlier, get an idea of the diversity of audio we can reach and generate. Here there are lists of the possible parameters to set in generation phase:

Genres (and variations):

- Pop (Synth pop, Indie pop, Dance Pop)
- Rock (Hard Rock, Punk Rock)
- Metal (Heavy Metal, Doom Metal, Thrash Metal)
- Jazz (Bebop, smooth jazz)
- Classical (Baroque, Contemporary Classical)
- Hip Hop (Trap, Drill, Boom Bap)
- Electronic (House, Dubstep, Techno, Trance)
- Reggae (Dance Hall, Roots Reggae, Dub, Reggaeton)
- Blues (Country Blues, Electric Blues)
- Country (Honky-Tonk, Country Rock)
- R&B
- Funk

- Folk (Celtic Folk, Indie Folk, Traditional Folk)
- Soul

There are about 15 genres with their main variations. The primary options were selected, which turned out to be the most consistent when input into MusicGen. In total, we have around 32 alternatives, corresponding to a 5-bit label.

Type of Production:

- Lo-Fi
- 8-bit
- Wide (broad and spacious sound)
- Vintage vintage (70s/80s/90s)
- Modern production (00s, 10s)

There are 8 different types of production available that can be mapped in a 3 bits label.

Instrument:

- Electric Guitar
- Acoustic Guitar
- Electric Bass
- Acoustic Drums
- Electronic Drums
- Piano
- Synthesizer
- Violin
- Saxophone
- Trumpet
- Flute
- Organ
- Harp
- Double Bass
- Harmonica
- Percussion

Here there are 16 instrument that can be mapped in 4 bits label.

Tempo:

- Slow: 60-80 BPM
- Moderate: 81-100 BPM
- Medium: 101-120 BPM
- Fast: 121-140 BPM

There are four main tempo division that can be mapped in a 2 bit label

Key:

- 3 bit that point to the note

- 2 bit that point to the octave (considering only the first 4 octaves)
- 1 bit to indicate the presence of the sharp (#) symbol

So 6 bits for the key are available.

Moods:

1. Energetic
2. Relaxing
3. Intense
4. Melancholic
5. Epic
6. Romantic
7. Dark
8. Festive

Finally there are 8 principal moods that can be mapped in 3 bits label.

From this analysis, we can deduce that, in the worst case, we have 20-23 bits with which to generate the audio tracks present in this dataset. This allows us to comfortably generate up to:

$$20 \text{ bits} \rightarrow 2^{20} = 1.048.576 \text{ combinations} \tag{3.1}$$

Here there is a summary table of our TIScode dataset:

Data	**Description**
Conditioned Generated Tracks	900
Unconditioned Generated Tracks	100
Audio Format	Mono
Extension Format	.WAV
Token Length	512
Time Length	5 seconds
Min. Degree of Freedom	20 bits
Available combinations	1.048.576

Table 3.1: TIScode Dataset Summary Table

3.2 Sound Feature Analysis

Now that we have a dataset capable of easily reaching a million highly diverse audio tracks during the generation phase, thanks to the many degrees of freedom we can control, we need to study how to effectively transmit this signal in order to build an efficient channel coding system.

Channel coding is the technique to transform a digital information message b_l into another message c_l (hence the term coding), which is more robust to the errors that the channel may introduce. The robustness of the coded message is obtained at the price of some redundancy, which is added on purpose in the form of additional symbols (parity symbols) uniquely determined by the

information message. The transmitted message is therefore one of a set C of possibly transmitted messages, which is a subset of a larger set $\tilde{C}$ of possibly received messages. When the channel corrupts the transmitted message, it will produce a message $\tilde{c}_l$ that is in $\tilde{C}$ but may (with a high probability) not be in C. This allows the receiver to recognize the message as invalid, and in some cases even to correct it, thus recovering the information message [48].

However, our multimedia transmission system is different from traditional radio communication: in our case, the redundancy in channel coding will come from the large number of analyzed features. The idea is to analyze various features and characteristics of our TIScode, then quantize them to create a bitmap for efficient channel encoding and decoding. In this section, we will explore the description and utility of many different features, ranging from common ones in audio tracks, such as key, to more high-level features like genre and instrument.

The study on the features was conducted mainly using Librosa, a pyhton package that provides the building blocks necessary to create music information retrieval system and MIRtoolbox, a Matlab tool that offers an integrated set of functions, dedicated to the extraction from audio files of musical features such as tonality, rhythm and structure [49][50].

3.2.1 Chroma

A chromagram is a visual representation of the pitch content of an audio signal, capturing the intensity of each of the 12 pitch classes (or chroma) in Western music—C, C#, D, D#, E, F, F#, G, G#, A, A#, and B—across time. Unlike a traditional spectrogram, which visualizes frequencies, a chromagram maps the pitch content into a circular representation, where pitches separated by octaves are treated equivalently. This makes chromagrams particularly useful for tasks like music key detection, chord recognition, and harmonic analysis, as they highlight the harmonic structure while disregarding octave differences [51].

Chroma features are an interesting and powerful representation for music audio in which the entire spectrum is projected onto 12 bins representing the 12 distinct semitones (or chroma) of the musical octave. Since, in music, notes exactly one octave apart are perceived as particularly similar, knowing the distribution of chroma even without the absolute frequency (i.e. the original octave) can give useful musical information about the audio – and may even reveal perceived musical similarity that is not apparent in the original spectra [52].

In practice, chromagrams are computed from an audio signal using various time-frequency transforms, such as Short Time Fourier Transform (STFT), Chroma Energy Normalized (CENS), Constant-Q Transform (CQT), Variable-Q Transform (VQT) [53]. Each method has different strengths, with the CQT and VQT being particularly well-suited for musical applications due to their ability to better align with the logarithmic nature of musical pitch [53]. We will utilize all four chroma analysis techniques to establish a voting system that will identify the top two chroma for the evaluated TIScode. The two chroma features that receive the highest number of votes will be incorporated as key attributes in our bitmap representation, enhancing the overall effectiveness of our analysis.

Chroma STFT Chroma STFT is computed by first applying the STFT to the audio signal

(as in 3.2). The STFT takes small, overlapping windows of the signal and computes the Fourier transform on each window, yielding a time-frequency representation.

$$X(t, f) = \sum_{n=0}^{N-1} x[n] \cdot w[n - m] \cdot e^{-j2\pi \frac{f}{N} n} \tag{3.2}$$

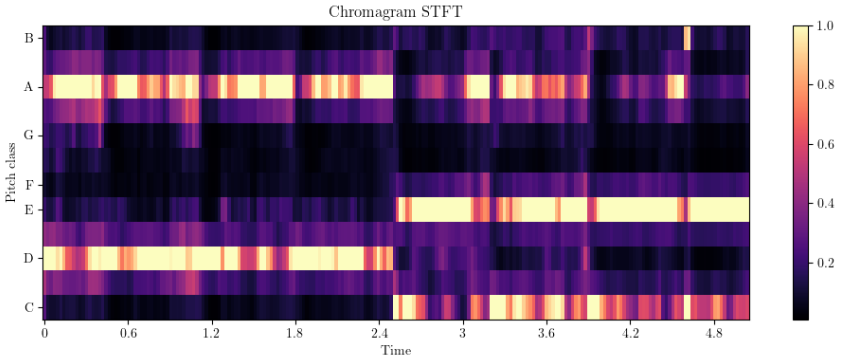


Figure 3.3: Chromagram STFT

This representation shows how the signal's frequency content changes over time. From this time-frequency matrix, only the magnitude is retained, ignoring the phase. The frequency components are then grouped into 12 bins corresponding to the 12 pitch classes (C, C#, D, etc.). Frequencies separated by octaves are summed together, as they correspond to the same pitch class [52]. Finally, the chroma values are normalized so that the energy of each frame is evenly distributed across the pitch classes, providing a harmonic snapshot at each time step. The result can be seen in Figure 3.6.

Chroma CQT Chroma CQT is computed using the Constant-Q Transform[54], which is similar to the Fourier Transform but with logarithmically spaced frequency bins. In the CQT, the width of the frequency bins decreases as the frequency increases, matching the logarithmic nature of musical scales.

$$CQT(f) = \sum_{n=0}^{N-1} x[n] \cdot w[n - m] \cdot e^{-j2\pi ft} \cdot \sum_{k=1}^{12} |X(t, k)|^2 \tag{3.3}$$

This allows for a more accurate representation of musical pitch than the STFT, especially at lower frequencies. The audio signal is first transformed using the CQT, and the resulting magnitudes are then mapped to the 12 pitch classes (see 3.4). Just like in the STFT-based chroma, frequencies separated by octaves are summed into the same pitch class. CQT is especially useful for harmonic and melodic content analysis in music because of its pitch-invariant resolution [55].

Chroma CENS Chroma CENS builds upon the chroma representation (typically using CQT or STFT) but adds an additional step of energy normalization and smoothing [56]. First, a

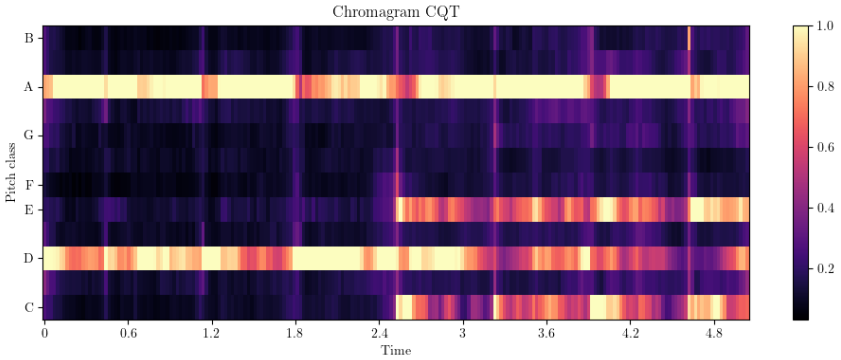


Figure 3.4: Chromagram CQT

standard chroma (CQT or STFT-based) is calculated, and then each chroma vector is normalized so that it captures the relative energy distribution across pitch classes rather than the absolute intensity (as in 3.4

$$\mathrm{CENS}(i,t) = \frac{\sum_{j \in \mathrm{class}(i)} |X(t,j)|^2}{\sum_{k=1}^{12} |X(t,k)|^2} \tag{3.4}$$

After this normalization, the chroma values are further smoothed over time using a moving average, making CENS less sensitive to short-term fluctuations. The result is a more stable representation, useful for higher-level tasks such as chord detection and key recognition. CENS focuses on harmonic information and ignores small variations, such as dynamic changes, which makes it ideal for robust music analysis[56].

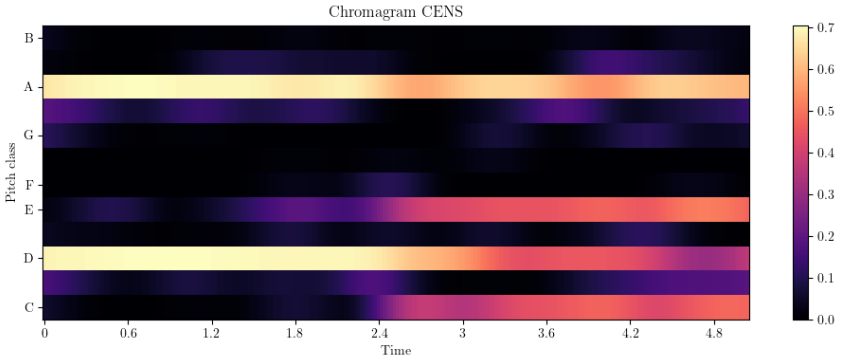


Figure 3.5: Chromagram STFT

Chroma VQT Chroma VQT is calculated using the Variable-Q Transform [57], a variant of the Constant-Q Transform that adjusts the frequency resolution dynamically. While the CQT has fixed bandwidth per octave, the VQT can adapt the resolution based on the frequency range, providing higher resolution at lower frequencies and lower resolution at higher frequencies.

$$VQT(f) = \sum_{n=0}^{N-1} x[n] \cdot w[n-m] \cdot e^{-j2\pi ft} \tag{3.5}$$

This flexibility allows the VQT to capture more detailed harmonic information in lower frequency ranges, which are crucial for musical tones like bass notes. Once the VQT is applied to the audio signal, the magnitudes are grouped into the 12 pitch classes, similarly to how it is done for STFT and CQT. The resulting chroma is then normalized. VQT is particularly useful for detailed harmonic analysis, especially when working with complex musical signals that have a broad range of frequencies [57].

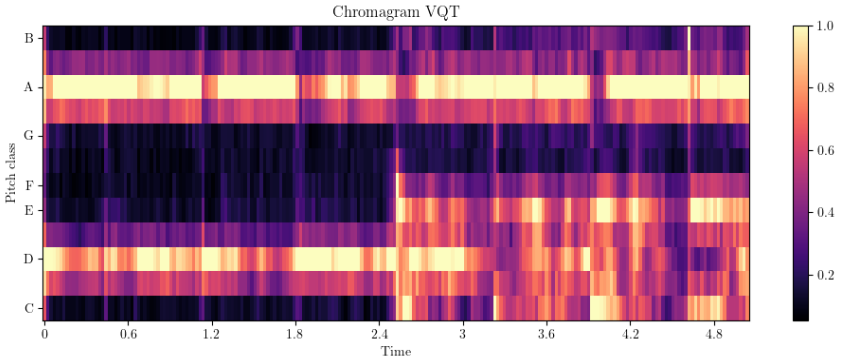


Figure 3.6: Chromagram VQT

3.2.2 Key

The key of a song refers to the central pitch or tonal center around which the harmony and melody are organized. It determines the set of pitches (or scale) that are most prominently used in the song, providing a framework for chord progressions and melodic structure [58]. Each key has a corresponding major or minor scale, and the tonic note (first note of the scale) serves as the most relevant note of the music. The key plays a vital role in shaping the emotional character of the piece. It is also a very important feature that can be used either to characterize and to recognize a sound track, or a TIScode, in our case [58]. It is, however, difficult to recognize correctly the key in presence of high noise, so here the key is computed using a CNN that is more noise resistant than the classic methods: Convolutional Representation for Pitch Estimation (CREPE).

CREPE is a state-of-the-art tool for pitch detection. CREPE uses a CNN to process raw audio samples and estimate the fundamental frequency (f0) every 10 ms. The architecture that can be seen in Figure 3.7 shows the six convolutional layers that operate directly on the time-domain audio signal, producing an output vector that approximates a Gaussian curve, which is then used to derive the exact pitch estimate. CREPE Notes enhances this process by combining the f0 estimates with a ”confidence” measure provided by CREPE and the pitch gradient of the audio signal. First, the method calculates the pitch gradient, accounting for the logarithmic perception of

pitch. Then, it combines CREPE's inverted confidence output with the normalized pitch gradient. This combined signal is then thresholded to detect peaks, which generally correspond to note boundaries. To address false positives, such as repeated notes where the pitch gradient is close to zero, the method employs an onset detection algorithm. Finally, an amplitude threshold is applied to remove silences and spurious instrumental activity, and the note boundaries are further refined through "amplitude trimming."

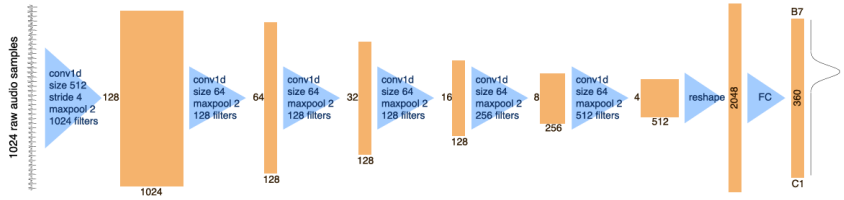


Figure 3.7: The architecture of the CREPE pitch tracker. [59].

Experimental results on two real instrumental music datasets, Filosax and ITM-Flute-99, show that CREPE Notes outperforms other note segmentation systems. Interestingly, the method works well even when using a smaller CREPE model, suggesting that its effectiveness comes from combining confidence information, pitch gradient, and signal processing. Its strengths include simplicity, reliability, and the ability to operate in real-time, making it a promising tool for various applications [59].

3.2.3 Top Tonnetz Class

The tonnetz (short for "tonal network") is a mathematical model used to represent the harmonic relationships between pitches. Specifically, Tonnetz represents the relationships between notes, chords, and keys based on harmonic proximity, primarily through concepts like fifths, thirds, and semitones. It is often visualized as a grid or network where each point corresponds to a pitch class, and connections between points represent harmonically close relationships. In the context of music analysis or feature extraction, the "top Tonnetz class" refers to the most prominent harmonic class derived from the tonnetz representation. When analyzing a piece of music, the tonnetz feature maps the harmonic content, and the "top" or most frequent/important Tonnetz class can represent the dominant harmonic structure, such as the most common intervals or chords present in the music [60].

The projection translates the 12 chroma features into six values (or classes), each encoding specific harmonic intervals. These dimensions are often tied to specific triads and musical intervals.

- **Perfect Fifth (2 Dimensions)**: the perfect fifth interval is one of the most fundamental relationships in harmony. For example, in the Tonnetz, C and G form a perfect fifth. Two dimensions represent this interval, encoding the harmonic movement across fifths. Moving along this axis helps capture modulations or transitions involving perfect fifths, which are critical for chord progressions and key changes.

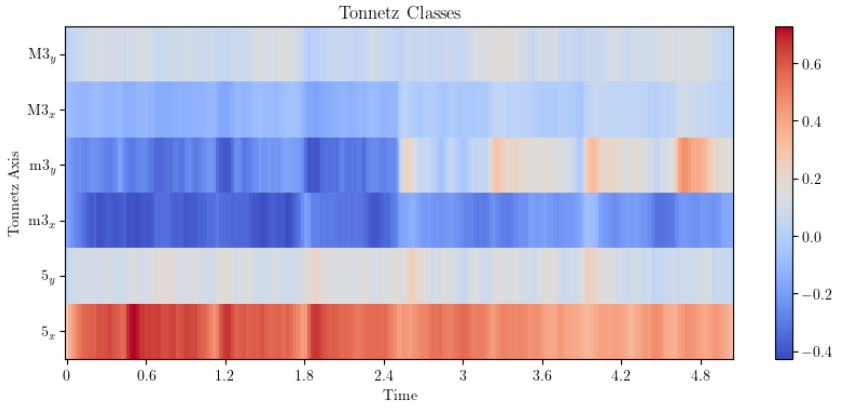


Figure 3.8: Tonnetz Classes

- **Major Third (2 Dimensions)**: the major third interval is important for major triads (e.g., C-E in the C major triad). This interval defines the "color" of major chords. Two dimensions represent the major third relationship, capturing harmonic structures based on major triads and their transitions.
- **Minor Third (2 Dimensions)**: The minor third interval is important for minor triads (e.g., C-E in the C minor triad). It characterizes the harmonic content of minor chords.

  Two dimensions represent the minor third relationship, capturing transitions and modulations in harmonic progressions that involve minor chords.

The idea is to select the most prominent axis and using it as one of the feature that describe the TIScode. In Figure 3.8 we can observe that x-axis of the Perfect Fifth is the most prominent one.

3.2.4 Width of the Maximum Correlation Peak

The width of the maximum correlation peak (Figure 3.9) in sound-related analysis is important because it provides key information about the resolution and accuracy of time alignment, synchronization, or sound source localization. In sound, cross-correlation is often used to align signals or find delays between them. A sharp and narrow peak indicates precise timing, meaning the signals are well-correlated at a specific delay. A wide peak, on the other hand, suggests ambiguity or that the signals are more spread out in time, which can reduce time resolution and make it harder to pinpoint exact time delays or alignments. A wide peak may also indicate that two signals are similar over a broader range of time shifts, which could mean they share common features over time. Conversely, a sharp, narrow peak suggests that the signals only match closely at a very specific delay or time shift. The width of the peak is related to the frequency content of the signals being correlated. Higher frequency signals with more sharp transitions often produce narrow peaks, while lower frequency or more slowly varying signals produce wider peaks. Therefore, the width can indicate the nature of the frequency content of the signal .

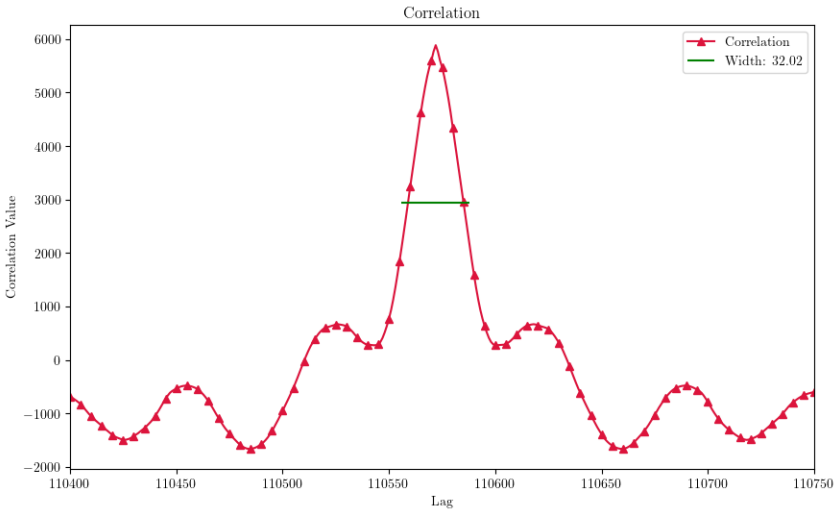


Figure 3.9: Example of Maximum Correlation Peak in a TIScode

In summary, the width of the max correlation peak in sound gives insight into timing precision, localization accuracy, signal similarity, and frequency content, all of which are crucial in various audio processing and analysis tasks [61].

3.2.5 Zero Crossing Rate

The Zero Crossing Rate (ZCR) is a commonly used feature in signal processing, particularly in audio analysis [61]. It refers to the rate at which a signal changes sign from positive to negative (or vice versa) within a specific time frame. In simpler terms, it counts how many times the signal crosses the horizontal axis (the zero point) in a given period.

Mathematically, the ZCR for a signal $x(t)$ can be expressed as:

$$ZCR = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}[x(t) \cdot x(t-1) < 0] \tag{3.6}$$

where:

- T is the total number of samples,
- $\mathbb{1}[\cdot]$ is an indicator function that counts each zero crossing event (i.e., when $x(t)\cdot x(t-1) < 0$, meaning one value is positive and the other is negative).

ZCR can be useful in identifying different genres or distinguishing between harmonic and percussive sounds [62]. ZCR also helps in identifying noisy sections of a signal, since noise often leads to more frequent zero crossings. In summary, the Zero Crossing Rate provides a simple yet powerful method to analyze the frequency characteristics of a signal and is widely used in tasks like speech processing, music analysis, and audio classification.

3.2.6 Sound Moments

The sound moments refer to various statistical characteristics of the audio signal. Below is a list of the different moments:

First Moment (Mean) The first moment represents the mean of the audio signal. It indicates the average level of the sound.

$$mean = \frac{1}{N}\sum_{i=1}^{N} y_i \tag{3.7}$$

Second Moment (Variance) The second moment measures the dispersion of the values around the mean. Variance indicates how much the values of the audio signal vary.

$$variance = \frac{1}{N}\sum_{i=1}^{N} (y_i - M_1)^2 \tag{3.8}$$

Third Moment (Skewness) The third moment measures the asymmetry of the value distribution. A positive value indicates that the distribution is skewed to the right, while a negative value indicates a left skew.

$$skewness = \frac{1}{N}\sum_{i=1}^{N} (y_i - M_1)^3 \tag{3.9}$$

Fourth Moment (Kurtosis) The fourth moment measures the "peakedness" of the distribution. A high value indicates a distribution with higher peaks and heavier tails.

$$kurtosis = \frac{1}{N}\sum_{i=1}^{N} (y_i - M_1)^4 \tag{3.10}$$

3.2.7 Tempo

Tempo refers to the speed or pace of a piece of music, typically measured in Beats Per Minute (BPM). It determines how quickly the beats occur and sets the overall feel and energy of the song. A faster tempo results in a more energetic and lively atmosphere, while a slower tempo can create a relaxed or somber mood. Beat tracking, or the identification of rhythmic moments in a musical piece, is a complex task that requires balancing two needs: on one hand, the beats should correspond to salient moments in the audio signal, such as the onset of a note; on the other hand, the beats should be evenly distributed over time, reflecting the musical tempo.

Librosa function to track the tempo of a song exploit the dynamic programming algorithm described by D. Ellis et al. in [63]. The article describes a system that addresses this problem by formulating it as an optimization problem solvable with dynamic programming. The system relies on a cost function that takes into account both the "onset strength" at a given moment in time (i.e., the probability that there is a beat at that point) and the consistency of the interval between beats relative to a predefined target tempo. The onset strength is calculated from a

spectrogram representation of the audio signal, processed to simulate the human auditory system. The target tempo is estimated through the autocorrelation of the onset strength function, with a greater weight assigned to the most common musical tempos. Dynamic programming allows for efficiently finding the sequence of beats that optimizes the cost function, ensuring that the best solution is found.

3.2.8 Rolloff

Rolloff refers to a frequency feature in audio analysis that indicates the frequency below which a specified percentage of the total energy (or magnitude) of a sound signal is contained. It is often used in music and audio processing to identify the spectral characteristics of a sound.

For example, a common measure, that is also used in this project, is the 90% rolloff, which indicates the frequency below which 90% of the total energy of the audio signal exists. This can help differentiate between different types of sounds and analyze their tonal qualities [64].

3.2.9 Contrast

Contrast in the context of audio refers to the variation in energy levels between different frequency bands in a sound signal. It can be understood as a measure of the dynamic range of the sound, indicating how pronounced or subtle certain frequencies are relative to others. High contrast means that there is a significant difference in energy between frequency bands, leading to a more textured or rich sound, while low contrast indicates a more uniform energy distribution across frequencies [65].

Contrast is particularly useful in sound analysis and music processing, as it can help differentiate between sounds with varying complexities and timbres, making it a relevant feature for tasks such as sound classification and audio effects processing.

3.2.10 Flatness

Flatness is a measure used in audio analysis to indicate how much a sound resembles a flat spectrum, meaning it has a relatively uniform energy distribution across different frequency bands. In other words, a sound with high flatness has a spectrum that is more "noise-like," while a sound with low flatness tends to have pronounced peaks in its frequency spectrum, indicating a more tonal or musical quality [66].

Flatness is often used in various audio applications, such as music genre classification and sound texture analysis, as it can help distinguish between tonal sounds (like musical notes) and non-tonal sounds (like noise).

Flatness is calculated as in 3.11:

$$\text{Flatness}(x) = \frac{10^{\sum_i \log_{10}(x_i)}}{n} \tag{3.11}$$

where x_i represents the spectral magnitude of each frequency bin and n is the number of frequency bins.

3.2.11 Spectral Centroid

Spectral centroid is a feature in audio analysis that represents the "center of mass" of the spectrum of a sound. It indicates where the center of the frequency distribution lies and is often associated with the perceived brightness of a sound. A higher spectral centroid value generally corresponds to a sound that is perceived as brighter or sharper, while a lower value is associated with darker or more muted sounds [67].

The spectral centroid is particularly useful in various applications, including music genre classification, instrument recognition, and timbre analysis, as it provides insight into the harmonic content of an audio signal.

To calculate the spectral centroid, first, a spectrogram of the audio signal is computed using the STFT. This representation shows how the energy of the signal is distributed across different frequencies over time. Then the magnitude of the spectral components is obtained from the STFT, which reflects the amplitude of each frequency bin. Finally the spectral centroid is calculated for each frame using the following formula 3.12:

$$C = \frac{\sum_i f_i \cdot X(f_i)}{\sum_i X(f_i)} \tag{3.12}$$

where:

- C is the spectral centroid.
- f_i represents the frequency of the i-th bin.
- $X(f_i)$ is the magnitude of the i-th frequency bin.

3.2.12 Entropy

Entropy in the context of a sound signal is a measure of the randomness or unpredictability within that signal. It quantifies the amount of information or complexity present in the sound. In audio signal processing, entropy is used as a feature to represent the texture or timbre of a sound. It can be combined with other features like spectral centroid, bandwidth, or pitch to improve the accuracy of sound recognition systems. High entropy might indicate the presence of noise or a high level of disorder in a signal. This can be useful in noise detection and reduction algorithms[68].

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_b p(x_i) \tag{3.13}$$

where:

- X is a random variable representing the set of events,
- $p(x_i)$ is the probability of event x_i,
- $\log_b$ is the logarithm to the base b (typically $b = 2$ for entropy in bits),
- n is the total number of possible events.

3.2.13 Genre

Genre is a crucial characteristic of a song. We classify songs by their genre every day, having some genres we like and others we dislike. Similarly, TIScodes, despite their brief duration, possess their own genre, which can be selected during the generation phase to ensure the musical piece aligns with the intended context for transmission. We can use genre as high-level feature to identify our TIScodes. We will use a CNN genre-classificator built as explained by D. Iakubovskyi in [69].

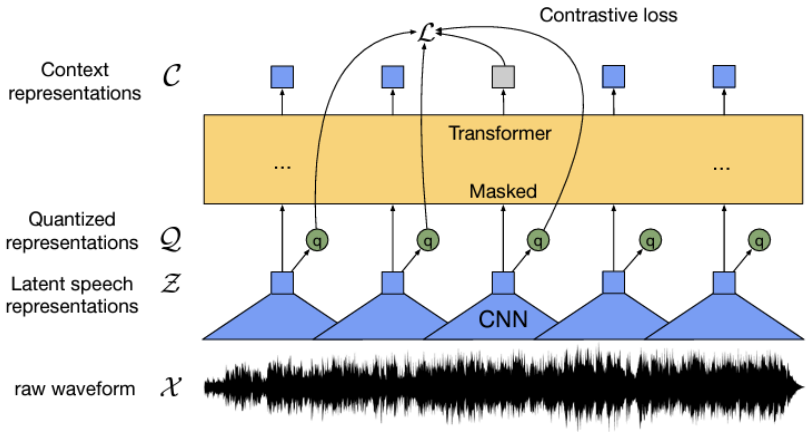


Figure 3.10: Wav2Vec Architecture

This classificator exploit wav2vec 2.0 (which architecture can be seen in Figure 3.10). Wav2Vec is a framework for Self-Supervised Learning of Speech and Sound Representations presents an innovative framework for machine learning of representations from raw audio data, without the need for transcriptions. The framework uses a self-supervised learning approach, meaning the model learns to understand the structure of sound by analyzing large amounts of unlabeled audio data [70]. The model work in three steps:

1. First, the model encodes raw audio into latent representations using a convolutional neural network (CNN).
2. Second, the model randomly masks parts of these latent representations, similar to the masked language modeling used to train language models like BERT.
3. The latent representations, including the masked ones, are then processed by a Transformer network to build contextual representations that capture long-range relationships within the audio sequence.

The dataset for the training of our classificator, GTZAN, contains 1000 sample 30-second audio files evenly split among 10 genres [71]:

1. Blues
2. Classical
3. Country
4. Disco
5. Hip-hop
6. Jazz
7. Metal
8. Pop
9. Reggae
10. Rock

Then we start by loading audio files of different genres, storing them in a DataFrame. We use random oversampling ensuring each genre has an equal number of samples. The audio is then resampled and truncated to a uniform length. This processed data is split into training and testing sets and then all data are passed to Word2Vec that is trained for this specific task. The model reach its best after 25 epochs with 83% of accuracy [69].

3.2.14 Top Instrument

Another important high-level feature is the main instrument of the sound track. Every track is characterized by an instrument that is perceived more than the others, so it's valuable to be able to recognize that instrument. In order to do this wav2vec is utilized again together a Musical Instrument's Sound Dataset[72] that contain data about:

- Guitar Sounds
- Drum Sounds
- Piano Sounds
- Violin Sounds

Despite the only four instrument considered here (that are, anyway, the principal ones) an accuracy of 97% is reached after 10 epochs[73].

3.2.15 Attack Time

The attack time in a song refers to the time it takes for a sound to reach its maximum amplitude or volume after the note is initially played or struck. It is a key aspect of a sound's envelope, which also includes decay, sustain, and release. It can be calculated following these steps:

1. *Identify the Starting Point:* Determine the moment when the sound begins. This is usually when the amplitude starts to rise from near silence.
2. *Identify the Peak Amplitude:* Find the point at which the sound reaches its maximum amplitude. This is the peak of the attack phase.
3. *Measure the Duration:* Calculate the time interval between the starting point (when the sound begins) and the peak amplitude. This time interval is the attack time.

In the context of musical instruments, a short attack time means the sound reaches its peak quickly, as in the case of a snare drum hit, while a long attack time means the sound builds up more gradually, as with a bowed violin note. The attack time can greatly affect the perceived intensity and emotional impact of a sound in a musical composition [50].

3.2.16 Spread

The spread of a musical piece refers to the distribution of frequencies and the variety of sounds present in the mix. In other words, it indicates how evenly the frequencies of a piece are distributed or concentrated in specific areas of the sound spectrum. A greater spread implies a higher diversity of frequencies, while a lower spread indicates a concentration in certain frequencies [61]. Spread can be calculated by collecting the frequency data from a piece. This can include amplitudes of frequencies in specific bands (e.g., lows, mids, highs). Finally, calculate the mean of the frequency amplitudes.

3.2.17 Roughness

Roughness is a perceptual property of sound that describes the sensation of irregularity or discontinuity in the texture of sound. It is commonly associated with complex sounds and noise, such as those generated by musical instruments, and can influence the emotional interpretation of music. The greater the roughness, the more the sound is perceived as ”rough” or ”gritty” [74] Roughness can be calculated using various methods, but one of the most common approaches is based on frequencies and their interactions. Usually the fundamental frequency is calculated and set in this formula:

$$R = \sum_{i=1}^{N} \frac{f_i^2}{(f_j - f_i)^2} \tag{3.14}$$

where:

- f_j is the reference frequency
- f_i is a frequency present in the signal..
- N is the total number of frequencies considered

3.2.18 Pulse Curve

A pulse curve represents the amplitude variation of a sound wave over time, often visualized as a graph that shows how the sound signal evolves. In music and audio processing, a pulse curve typically refers to the waveform of a musical note, capturing the changes in loudness and dynamics throughout its duration. Pulse curves help in understanding the dynamics of a sound, illustrating how loud or soft a sound is at different moments. This is essential for musicians and sound engineers in composing and mixing music. From the Pulse Curve the mean and the maximum

peak are extracted (see Figure 3.11). In fact, the mean amplitude can be compared to the peak amplitude (the highest point in the pulse curve) to assess the dynamic range of the sound. A larger difference between the peak and mean values indicates a more dynamic audio signal, which can be critical for expressive performances.

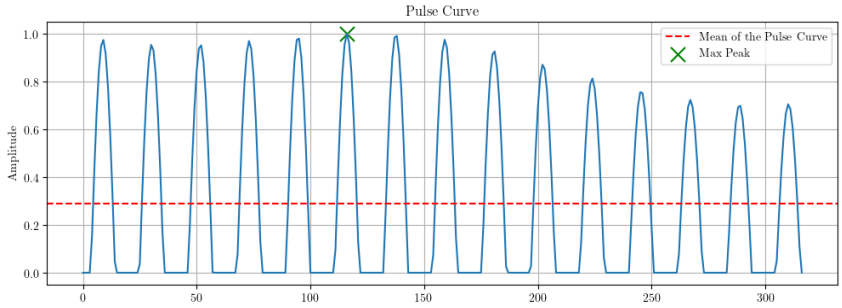


Figure 3.11: Plot of a TIScode Pulse Curve

While the mean does not directly correlate to perceived loudness (which is affected by the human ear's response to different frequencies), it can be a useful indicator when combined with other measures, like the root mean square Root Mean Square (RMS) level, which better reflects perceived loudness [50].

3.2.19 Root Mean Square (RMS)

RMS is a statistical measure of the magnitude of a varying quantity. It is commonly used in signal processing to quantify the average power of an audio signal, but it can be applied to any set of numbers or signal data. RMS provides a single value that represents the overall level or intensity of a signal [50].

$$\text{RMS} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \tag{3.15}$$

where:

- n is the number of values in the set
- f_i represents each individual value

This feature is selected together with the **lower energy ratio** that indicates the percentage of frames with RMS below mean RMS. While, another adopted feature, is **low energy** that indicates the minimum value of energy of the signal.

3.3 Frequency and Peaks Analysis

The analysis of the frequency of an audio signal and its peaks to construct a characterization began to be used after the publication of the Wang et al. paper in 2003 [75]. The idea of

this work is therefore to analyze the TIScodes not only from the perspective of the previously considered features but also from the frequency point of view. The goal is to create an identifier, a fingerprint, that utilizes two different types of analysis. The better and more accurately our sounds are characterized, the better, simpler, and more correct the decoding and recognition phase will be. In this section, it will be explained how the TIScodes will be analyzed from this perspective, detailing the various steps necessary to construct an identifier based on frequency peaks and their temporal spacing. The following explanation is inspired by the work of M. Strauss, which can be found in [76].

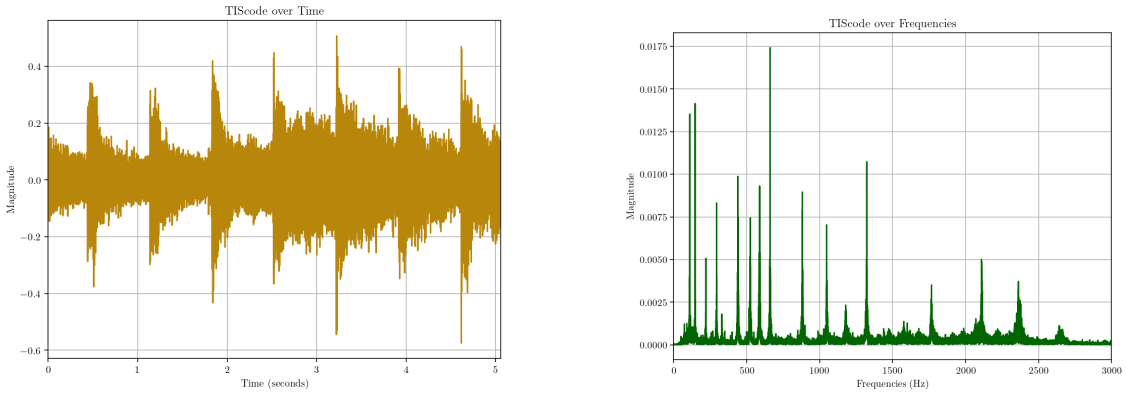
3.3.1 Constellation Map

As mentioned, we want to start by analyzing the signal in the frequency domain to understand what might be useful for its characterization. To visualize a signal in the frequency domain, we need to perform a Fourier transform.

The Fourier transform is a mathematical method that converts a signal from the time domain to the frequency domain, allowing us to analyze the frequency components of the signal. It is calculated by integrating the signal function multiplied by a complex exponential function [77]. The formula for the Fourier transform F(ω) of a function f(t) is given by:

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t}\, dt \tag{3.16}$$

We can thus see how a TIScode changes as a function of time (Figure 3.12a), and then apply the Fourier transform to observe its frequency behavior (3.12b).



(a) TIScode over time

(b) TIScode over frequency

Figure 3.12: TIScode representation over time and frequency

Typically, the microphone capsules in phones, the main devices intended to receive TIScodes, are of low quality and are primarily capable of effectively capturing only low frequencies. Therefore, this study will focus mainly on frequencies below 3000 Hz. This approach helps us avoid working

with peaks that may not be recognized due to the frequency response of the devices involved or that could be easily distorted by external noise.

As can be seen in the frequency representation of the signal, there are several peaks. The idea is to track these peaks and their temporal location. However, it is not feasible to select all the peaks; we need to focus only on the most influential ones. To achieve this, we use the *"argpartition"* function from NumPy. *"argpartition"* is useful for identifying the main peaks in an audio signal because it allows for the rapid identification of points of maximum interest without the need to fully sort the entire signal. It works by finding the indices of the first k highest or lowest values in an array, placing them in the first k positions, while the order of the other elements remains undefined. In the context of audio signal analysis, it can be applied to an array of amplitudes to quickly isolate the main peaks without performing a complete sort. This allows for the efficient identification of salient characteristics of the signal, aiding also further analyses such as sound event recognition or classification of its timbral properties [78].

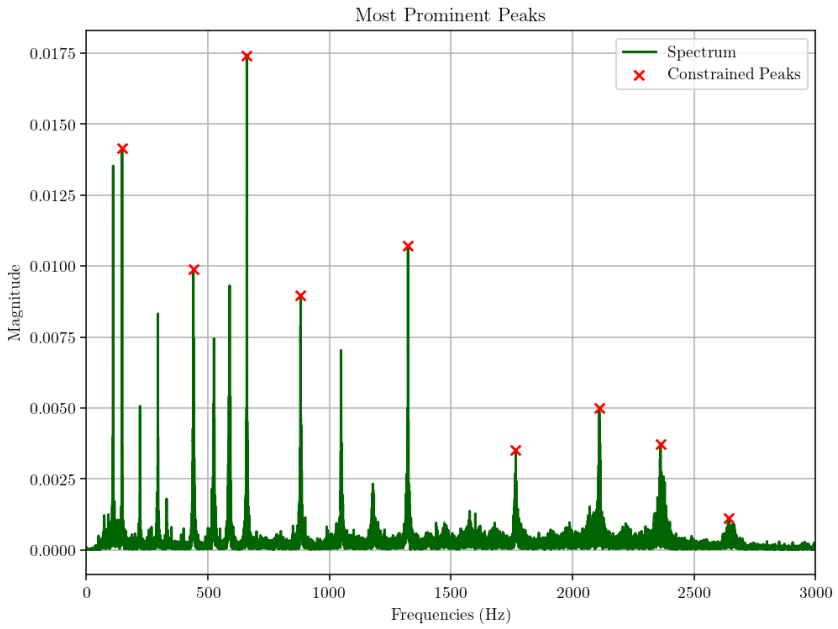


Figure 3.13: Tracked Peaks of a TIScode

In Figure 3.13, we see all the identified peaks that are most relevant (the maximum number of peaks to consider can be set according to the needs). This analysis was performed on the entire TIScode; however, we want to avoid requiring the entire audio track because, despite the precautions already discussed, the recording might start late, or significant noise might heavily corrupt part of the short sound message. Therefore, it is useful in this case not to apply the Fourier transform to the entire signal but rather to divide it into snippets of a selected length and apply the STFT.

STFT (Formula 3.17) is a variant of the Fourier transform that analyzes non-stationary signals

by dividing them into temporal segments (windowing) and applying the Fourier transform to each segment.

$$\text{STFT}\{x(t)\}(m,\omega) = \sum_{n=-\infty}^{\infty} x[n] \cdot w[n-m] \cdot e^{-j\omega n} \tag{3.17}$$

Now we apply a STFT to our TIScode example with a window length of 1 second and a maximum peak search of 10 peaks per snippet. By taking the peaks and representing them in a scatter plot where the x-axis represents the time progression and the y-axis represents the frequency, we can create the so-called "constellation map," which can be seen in Figure 3.14. The individual points represent all the selected peaks, and they are all centered on integer seconds due to the selected window length.

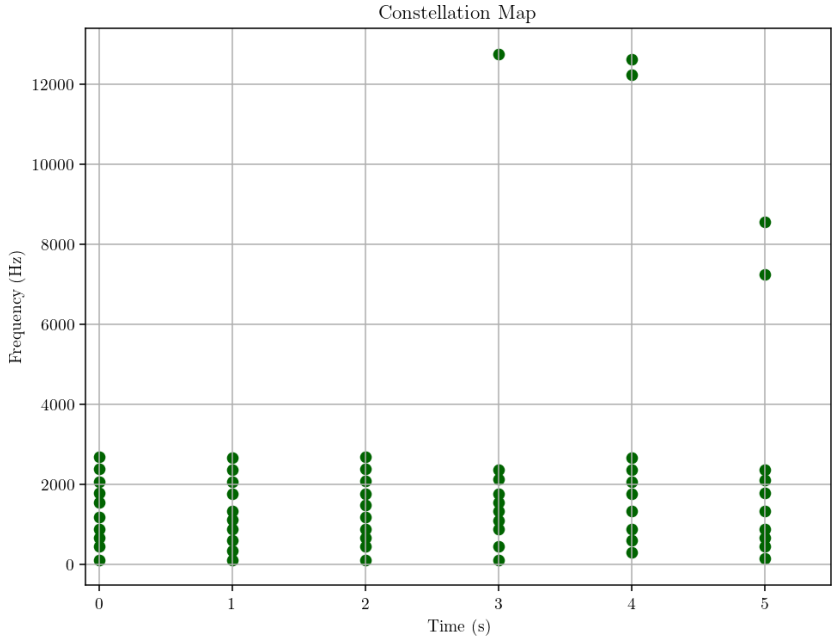


Figure 3.14: Constellation Map of a TIScode

The main idea is that each TIScode has its own unique constellation map that identifies it uniquely. However, due to noise or recording issues, it is not easy to recognize the exact peaks of the song, so the approach will now also track the distance between them. The points in the constellation map will be combinatorially associated. Each point will be paired with several other points to form pairs of frequencies, stored with the difference in time between them [75].

3.3.2 Combinatorial Hashing

We now have a useful set of data: numerous peaks for each TIScode snippet (with temporal duration depending on the STFT window length) and the distance between them. We need to

take one more step to figure out how to store them in a database.

We choose to divide the frequencies into 1024 bins so that we can use 10 bits to represent a single frequency. Two peaks will then be described with 20 bits. We then decide to map their temporal difference using 12 bits. The distance is not recorded only between a peak and its immediate successor; instead, a maximum number of peaks is chosen to track differences, discarding peaks that are too close or too far apart. This approach creates an identifier that is robust yet easy to calculate and quick to retrieve.

Now we want to exploit all these data to create a storable hash. An hash is a function that takes an input and produces a fixed-size output, known as a hash value. It is deterministic, meaning the same input will always yield the same output, and it is designed to be efficient and secure, making it difficult to find two different inputs that produce the same hash.

Now consider two frequencies, 1100 Hz and 1200 Hz, which are binned into 10-bit integers:

$$1100 \text{ Hz} \rightarrow \text{bin } 1100 \rightarrow 1100_{10} = 10001001100_2$$

$$1200 \text{ Hz} \rightarrow \text{bin } 1200 \rightarrow 1200_{10} = 10010110000_2$$

Assume the time difference between these peaks is 1 second, which is stored as a 12-bit integer:

$$1 \text{ second} \rightarrow 000000000001_2$$

The resulting 32-bit hash is created by concatenating the binary representations:

$$\text{Hash} = \underbrace{10001001100}_{\text{1100 Hz, 10 bits}}\ \underbrace{10010110000}_{\text{1200 Hz, 10 bits}}\ \underbrace{000000000001}_{\text{1 sec, 12 bits}}$$

$$\text{Hash} = 10001001100\,10010110000\,000000000001_2$$

In decimal, this hash is:

$$\text{Hash} = 112722145_{10}$$

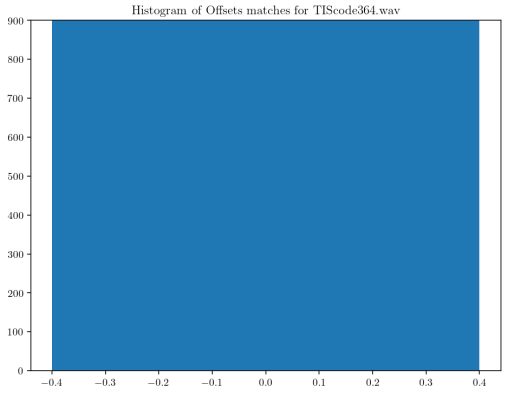
This 32-bit hash uniquely identifies the combination of these two frequency peaks and their time difference.

This is just one hash related to the distance between two peaks of our TIScode; we can produce and save many more by adjusting the settings of our system. For example if we want to track 10 peaks for each 1-second STFT snippet of a 5-second audio, and we want to calculate the distance between each of these peaks and only the next set of 10 peaks, we can reach 100 hash for each second, so about 500 hashes.

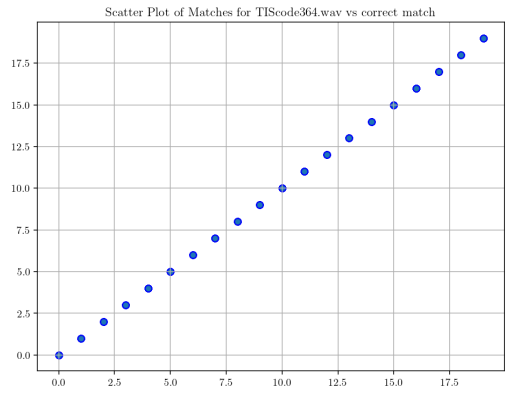
Once our hash database for the TIScodes has been created, we will now analyze how to utilize the hash structure to identify TIScodes recorded by our phone.

We can calculate a histogram of the number of matches per offset. The height of the tallest peak in the histogram is the best score for that match. In a situation as in Figure 3.15a, where

there is no noise at all we can see the result of a full correct match. Figure 3.15b shows a scatter plot that show the perfect matches of the different hashes over time.



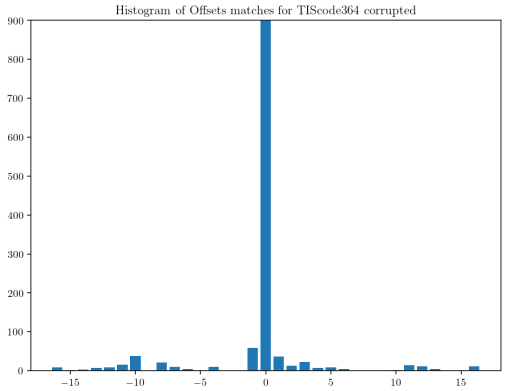
(a) Number of matches per offset



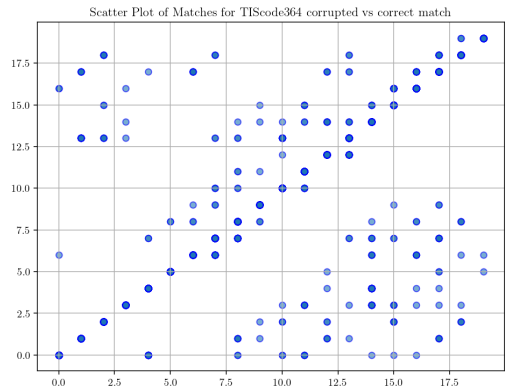
(b) Scatter plot of the matching hashes

Figure 3.15: TIScode match without noise

However, an ideal situation will never exist, so we must check what happens during recognition in the presence of noise. The noise can also be introduced by the frequency response of microphone capsules, which often tend to distort the signal significantly. In this case, we wouldn't have a perfect match as before, but we will have a very clear peak that indicates a perfect match (Figure 3.16a). If we observe Figure 3.16b, we can see that there is not only a perfect line with a coefficient of 1, but also a clear match despite the noise and some mismatches.



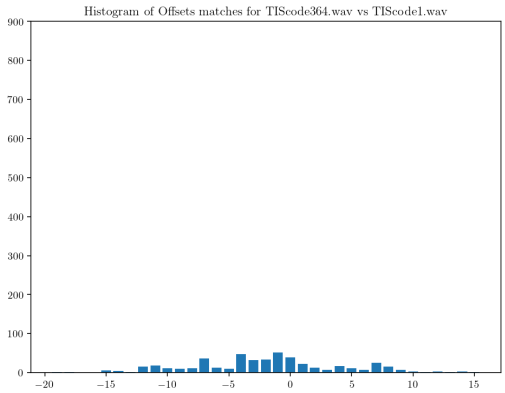
(a) Number of matches per offset



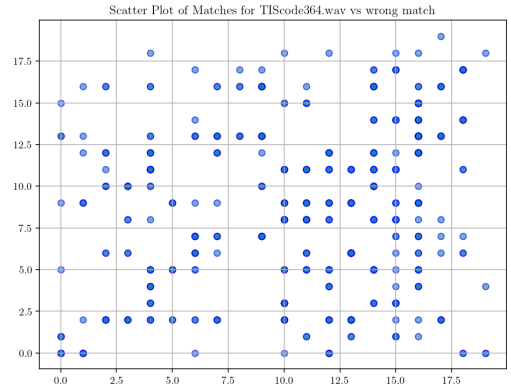
(b) Scatter plot of the matching hashes

Figure 3.16: TIScode match corrupted with noise noise

In the worst-case scenario, when there is a wrong match, we will observe either a flat histogram with no distinct peaks (Figure 3.17a) or a scatter plot with random points, showing no correlation at all (Figure 3.17b).



(a) Number of matches per offset



(b) Scatter plot of the matching hashes

Figure 3.17: TIScode wrong match

4

Analysis and Results

We are now able to generate, compute features, and analyze the frequency peaks of our TIScodes. This allows us to analyze the complete TIScode system pipeline (Figure 1.1), from its creation, through transmission via the channel, to its recognition.

In Section 4.1, we will explore how to construct the bitmap that identifies each individual TIScode and how it can assist in error detection or correction.

In Section 4.2, we will examine how TIScodes are recognized once received by a phone, implementing a recognition process that combines both feature analysis and peak detection.

Finally, in Section 4.3, we will validate our hypotheses in the field by reviewing recognition results through real-world testing.

4.1 Feature Bitmap for Channel Coding

The first part of the TIScode system pipeline involves generating the jingle through a text prompt, to which the data to be transmitted is assigned. Once our short sound message has been paired with the digital infomration, its features, both those computed via CNN and others, are extracted, a bitmap is created that uniquely identifies the TIScode, serving as its representation in the transmission channel (Figure 4.1).

4.1.1 Features Quantization

We now discuss how to manage our features. Starting with a dataset of generated sounds, as described in Table 3.1, we perform the following operations:

1. Calculate and save the features for all TIScodes

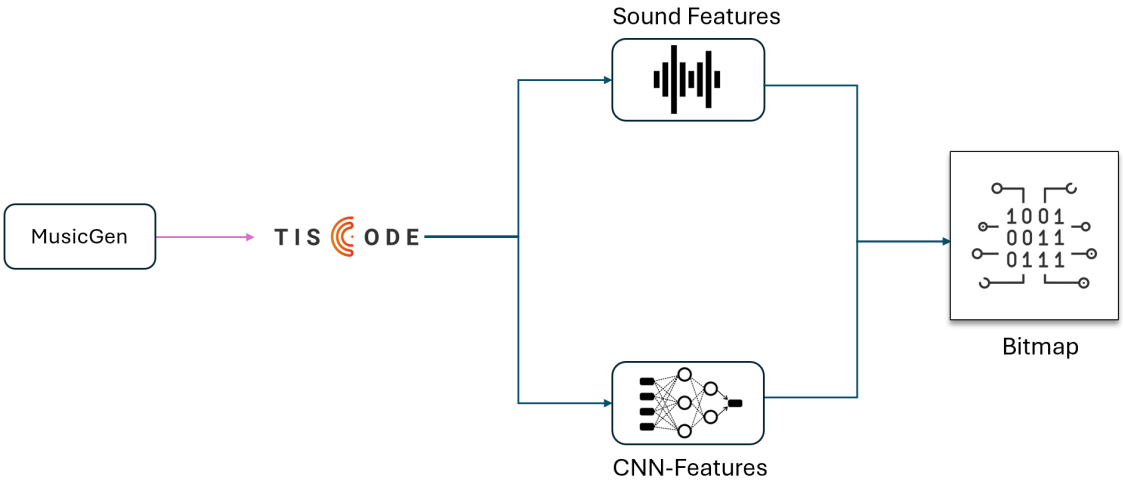


Figure 4.1: Bitmap creation for channel coding

2. Create histograms representing the distribution of feature values, expressed as numerical values within our dataset (Appendix Figure 4.15)
3. Quantize each numerical feature into 256 intervals, obtaining an 8-bit value for each feature
4. Calculate the required bits to represent non-numerical features
5. Map all the features of individual TIScodes according to their intervals
6. Create the corresponding bitmap for each TIScode
7. Save the bitmap

Regarding features not represented by numerical values, they were mapped as follows:

Top 1 and 2 Chromas: As discussed in Subsection 3.2.1, this feature analyzes the two most prominent notes within a 5-second sound clip. To represent the Chromas, we need:

- 3 bits to represent the seven notes [A, B, C, D, E, F, G]
- 1 bit to indicate the presence or absence of the sharp (#)

Considering that B# and E# do not exist, we have a total of 12 values, so we still need 4 bits. *Total Bit:* 4 bit

Key: Key is an important feature that is calculated with a CNN solution (see Subsec 3.2.2. To represent the key we need:

- 3 bits to represent the seven notes [A, B, C, D, E, F, G]
- 3 bits to indicate the octave [1, 2, 3, 4, 5, 6, 7, 8]

- 1 bit to indicate the presence or absence of the sharp (#)

- 1 bit do indicate if it is in major or minor scale

Total Bit: 8 bit

Genre: The Genre is the second CNN-feature and we are able to identify a song with 10 different genres (see Subsection 3.2.13). Here we need 4 bit.

Total Bit: 4 bit

Top Instrument: As we said in Subsection 3.2.14 we are able to identify the four main instrument: guitar, piano, drum and violin.

Total Bit: 4 bit

Top Tonnetz Class: We have to represent now the six projection that encode the different harmonic intervals (see Subsection 3.2.3).

Total Bit: 3 bit

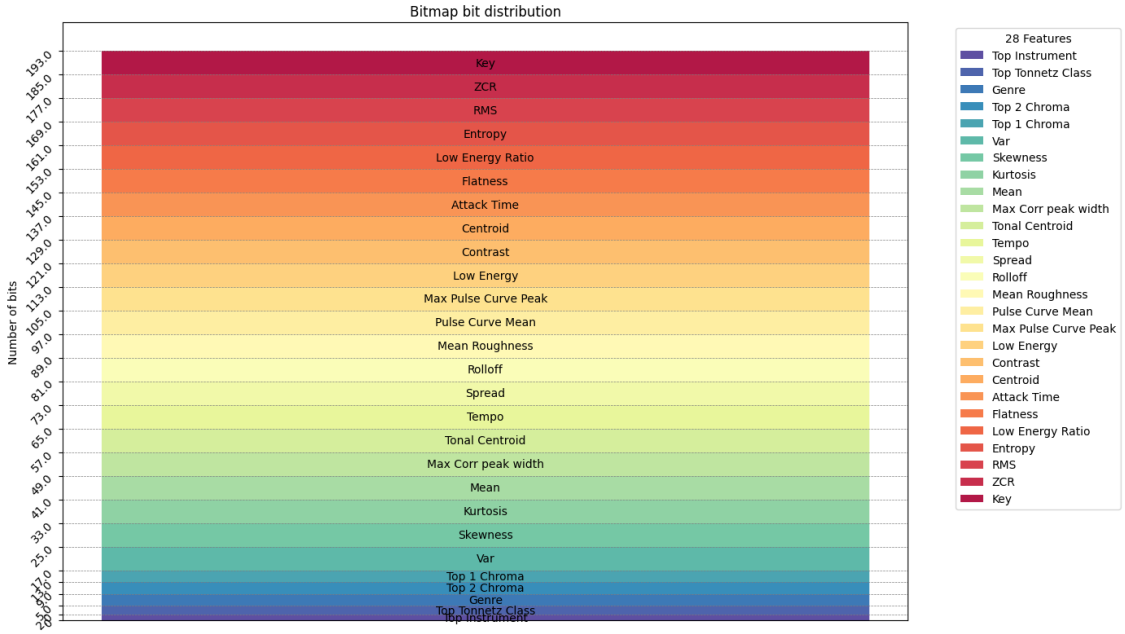


Figure 4.2: Feature Bit Allocation (Ordered by number of bits)

In this way, we can map all the calculated features into bit strings composed of blocks, each representing different characteristics of our short sound message, thereby creating our 193-bits bitmap (Figure 4.2 and Table 4.1).

Feature	Bits
Attack Time	8
Centroid	8
Contrast	8
Entropy	8
Flatness	8
Genre	4
Key	8
Kurtosis	8
Low Energy	8
Low Energy Ratio	8
Max Corr peak width	8
Max Pulse Curve Peak	8
Mean	8
Mean Roughness	8
Pulse Curve Mean	8
RMS	8
Rolloff	8
Skewness	8
Spread	8
Tempo	8
Tonal Centroid	8
Top 1 Chroma	4
Top 2 Chroma	4
Top Instrument	2
Top Tonnetz Class	3
Var	8
ZCR	8

Table 4.1: Feature Bit Allocation (Alphabetical Order)

4.1.2 Hamming Distances Evaluations

Having established the method for constructing a bitmap and created our bitmap dataset, we now introduce one of the most important parameters in coding theory: the *minimum Hamming distance*.

We first remind that the concept of *Hamming distance* between two binary words $\mathbf{c} = [c_1, \ldots, c_U]$ and $\hat{\mathbf{c}} = [\hat{c}_1, \ldots, \hat{c}_U]$ is the number of binary digits in which $\mathbf{c}$ and $\hat{\mathbf{c}}$ differ.

Now we can state that *the minimum Hamming distance* of a code $\mathcal{C}$ is defined as:

$$d_{\min} = \min_{\substack{\gamma_1, \gamma_2 \in \mathcal{C} \\ \gamma_1 \neq \gamma_2}} d_H(\gamma_1, \gamma_2) \tag{4.1}$$

where γ_1 and γ_2 are the two bitmaps that give the $d_{\min}$ when compared.

In this work, we exploit block coding that is a method used in data transmission where inform-

ation is divided into fixed-size blocks, and redundancy is added to each block to detect and correct errors. The original data block (information word) is transformed into a longer code block (code word) by adding extra bits. However, in our system setup, the redundancy is currently given by the large number of features used, in order to trying to decode correctly the TIScode. These features mapped into bits can help identify and fix errors caused by noise or interference in the communication channel.

With block coding, the possibility of detecting and correcting errors introduced by the channel lies in the fact that not all possible received words $\tilde{c}_m$ are codewords. Indeed, an error will be detected if and only if the corrupted word $\tilde{c}_m$ does not lie in the code $\mathcal{C}$.

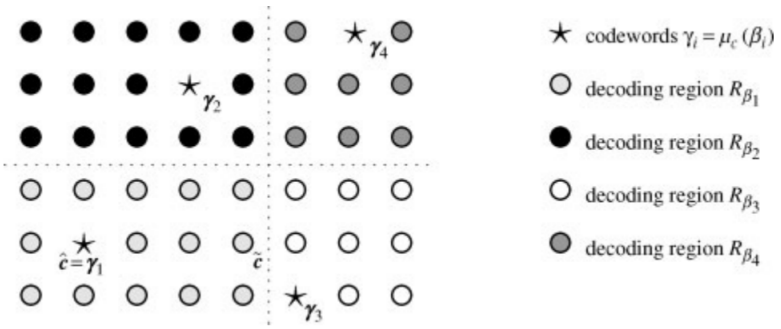


Figure 4.3: Minimum Distance Decoding Rule [48]

As a general decoding criterion we outline the rule based on the decoding regions. Consider the space $\mathcal{A}^n$ of possible received words $\tilde{c}_m = \xi \in \mathcal{A}^n$. We partition it into 2^k disjoint sets $\{R_{\beta_i}, i = 1, \ldots, 2^k\}$, called *decoding regions*, one for each information word $b_m = \beta_i$, as illustrated in Figure 4.3 [48].
Hence the decoding rule is:

$$\text{if} \quad \tilde{c}_m \in R_{\beta_i} \quad \text{then} \quad \hat{b}_m = \mu_d(\tilde{c}_m) = \beta_i \tag{4.2}$$

and so, we have the following:

Proposition 1 *A binary block code with minimum Hamming distance d_{min} can detect all patterns of $(d_{min} - 1)$ or fewer errors* [48].

Furthermore, the following proposition states how many errors can be corrected in a single word with minimum Hamming distance decoding.

Proposition 2 *Let $\mathcal{C}$ be a binary block code with minimum Hamming distance d_{min}. Let $c_m \in \mathcal{C}$ and $\tilde{c}_m \in \mathbb{A}^n$ be the transmitted codeword and received word, respectively. If the number of errors, $t = d_H(\tilde{c}_m, c_m)$, introduced by the binary channel in the word $\tilde{c}_m$, satisfies*

$$t < \frac{d_{\min}}{2} \tag{4.3}$$

then, regardless of the error positions, the received word will be correctly detected as $\hat{c}_m = c_m$ [48].

At this point, the next step is to calculate and analyze how the Hamming distances are distributed within our dataset. We compute the Hamming distance between each bitmap and all the others, then plot a histogram that represents the frequencies of the various Hamming distances observed, as shown in Figure 4.4.

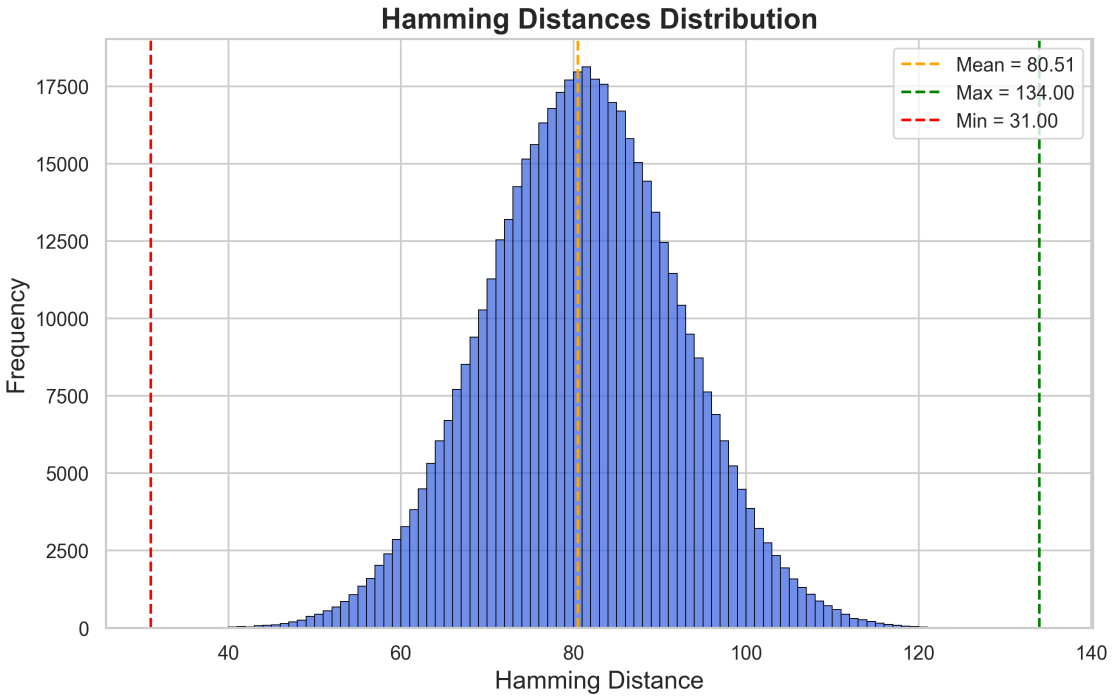


Figure 4.4: Hamming distances distribution within the TIScode dataset

Here we can observe that, with the generated mapping, we achieve an average distance of 80 bits, a maximum distance of 134, and, most importantly, we obtain our key result, which is 4.4:

$$d_{\mathbf{min}} = 31 \tag{4.4}$$

This result is very important because, thanks to *Proposition 1* and *Proposition 2*, we can understand how many errors we are able to detect or correct. In our case:

$$\textbf{Number of detectable errors} = d_{\mathbf{min}} - 1 = 31 - 1 = \mathbf{30}\ \boldsymbol{errors} \tag{4.5}$$

$$\textbf{Number of correctable errors} = \left\lfloor \frac{d_{\mathbf{min}}}{2} \right\rfloor = \left\lfloor \frac{31}{2} \right\rfloor = \mathbf{15}\ \boldsymbol{errors} \tag{4.6}$$

These two results show us that, during channel decoding, we can correct 15 errors, which essentially correspond to two entire features being detected incorrectly due to noise or distortion. We

are able to fully correct these errors by getting as close as possible to the original codeword, thereby obtaining the TIScode identifier and creating the pointer to the digital information database with which that sound was recorded.

We can also observe that not all minimum distances are clustered near the Hamming distance of 31. In fact, 31 is a minimum value that occurs only in a few cases. As shown in the CDF plot in Figure 4.5, the probability of generating a TIScode with a Hamming distance below 50 is nearly 0%, and the probability of obtaining one with a distance below 60 bits is less than 10%.

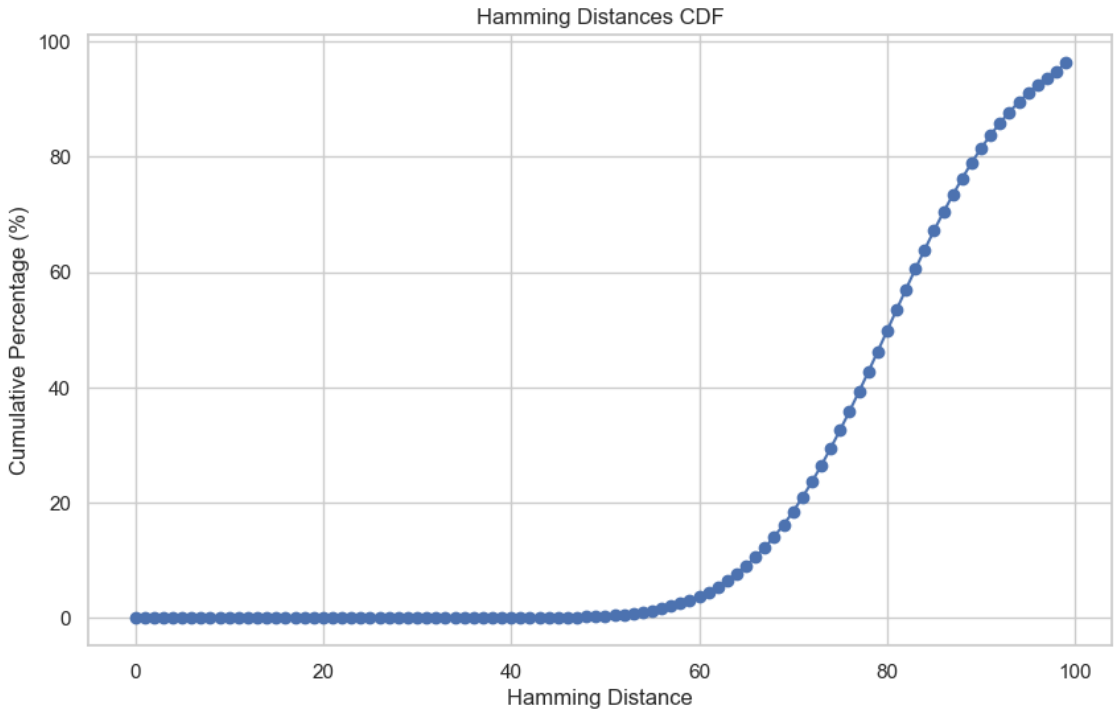


Figure 4.5: CDF of Hamming Distances

While under our current conditions these observations may not directly impact our results, since what ultimately matters is the minimum distance, we can choose to intervene at the source, during the generation or post-generation phase.

By analyzing the minimum Hamming distances during the generation of the various parts of our dataset, it was observed that the minimum Hamming distance found when dividing our dataset into clusters of 100 elements each is determined by the 100 TIScodes unconditionally generated as shown in Figure 4.6.

From this graph, we can draw three main observations:

1. This represents the worst-case minimum distance among the various blocks of 100 generated sounds in the dataset, with a minimum distance of 35 bits. This highlights that, when increasing the order of magnitude from 10^2 to 10^3, we see a decrease of only 4 bits in the minimum distance, indicating that the encoding is robust to changes in dataset size.

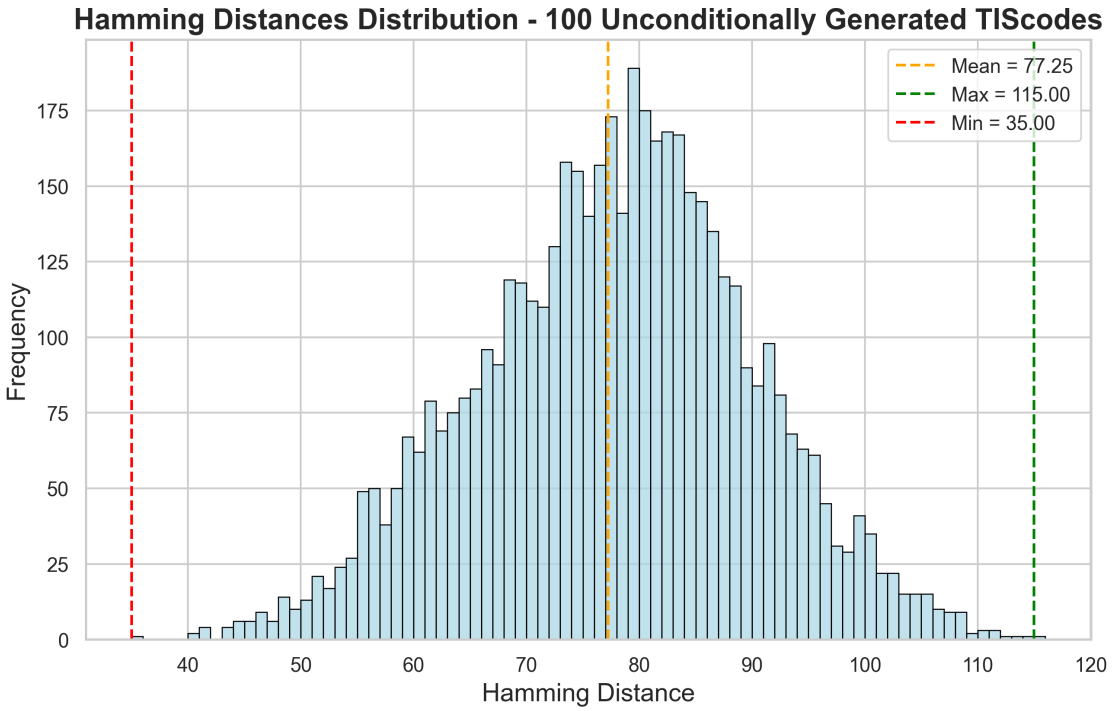


Figure 4.6: Hamming Distances distribution for the 100 unconditionally generated TIScode

2. The average and maximum distances are lower here compared to the entire dataset of 1000 signals, a consequence of the fact that the bitmaps of the signals generated through prompts are significantly larger than these.

3. The difference in the slope of the graph in Figure 4.4 compared to Figure 4.6 is notable. In the latter, the frequency of Hamming distances increases much more slowly than in the former, indicating a trend of greater Hamming distances among the bitmaps of the prompt-generated sounds.

From these graphs and this analysis, we can draw the following conclusions:

- The unconditionally generated signals tend to exhibit more similarities at the feature level. This can also be confirmed by listening to them; they are more likely to be signals with very low sounds, resembling noisy signals rather than musical tracks. Therefore, unconditional generation is not an effective solution.

- The signals that cause a sharp decrease in the minimum Hamming distance are very few, as seen in Figure 4.5. This leads us to consider a brute-force approach during generation. We could assess the minimum distance of a newly generated signal against the available database and establish a minimum distance threshold below which the signal is discarded and regenerated. It may occur that, even during prompt-based generation, the result is

unsatisfactory, both in terms of auditory pleasantness and differentiation from the dataset. Although MusicGen is an excellent sound generator, it is still a developing and improvable tool.

In conclusion, with the current minimum Hamming distance, we can still correct or detect a considerable number of errors. However, to refine the system further, a control approach during generation could be introduced to enhance the possibilities of correction during decoding.

4.2 Recognition System

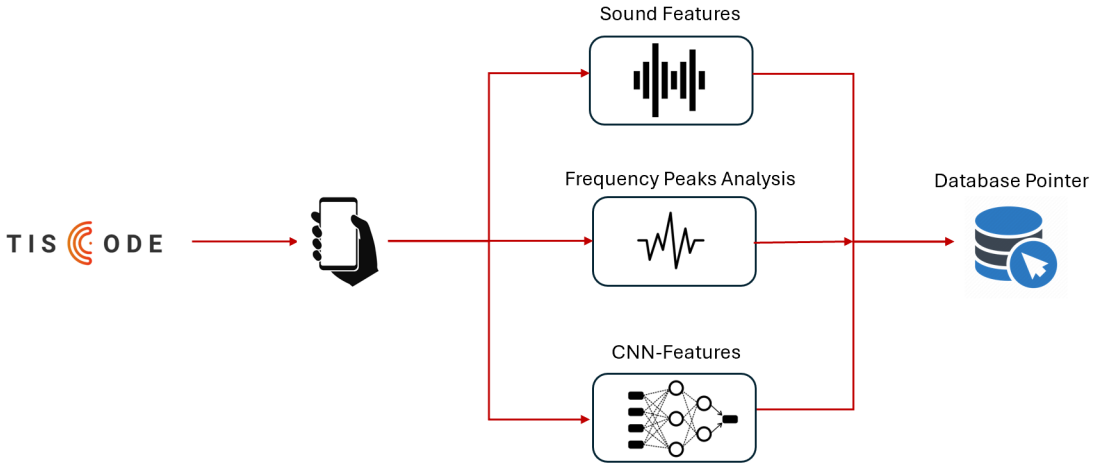


Figure 4.7: TIScode recognition system

Now that we have analyzed and explained the first part of the TIScode pipeline concerning generation and channel coding, we will move on to verifying the validity of our studies through recognition tests of TIScodes corrupted by noise and distorted by the frequency response of phone microphones.

In the Subsection 4.2.1, we will explain the algorithm used for recognition, detailing the steps required to match the received TIScode with the one stored in the database in order to retrieve the digital information it contains.

In Subsection 4.2.2, we will examine and discuss the results of the simulations.

4.2.1 Recognition Algorithm

The process typically followed for the reception and decoding of the TIScode is as follows:

1. The TIScode is emitted by a source
2. The application on the mobile device detects the opening marker and starts recording
3. The recorded TIScode is then analyzed and decoded

4. If a match is found with a TIScode in the database, a pointer to the database is created

5. The corresponding digital information is retrived and displayed on the smartphone

This process takes into account various factors, such as distortions caused by the frequency responses of the source, receiver, and the room environment, as well as background noise (as shown in Figure 4.7) To run computer simulations, we need to simplify the process to allow for the introduction of different types of noise. While in the next section, real tests will be conducted and presented. However, to perform a large number of simulations under varying conditions, we introduce our simplified process, illustrated in Figure 4.8.

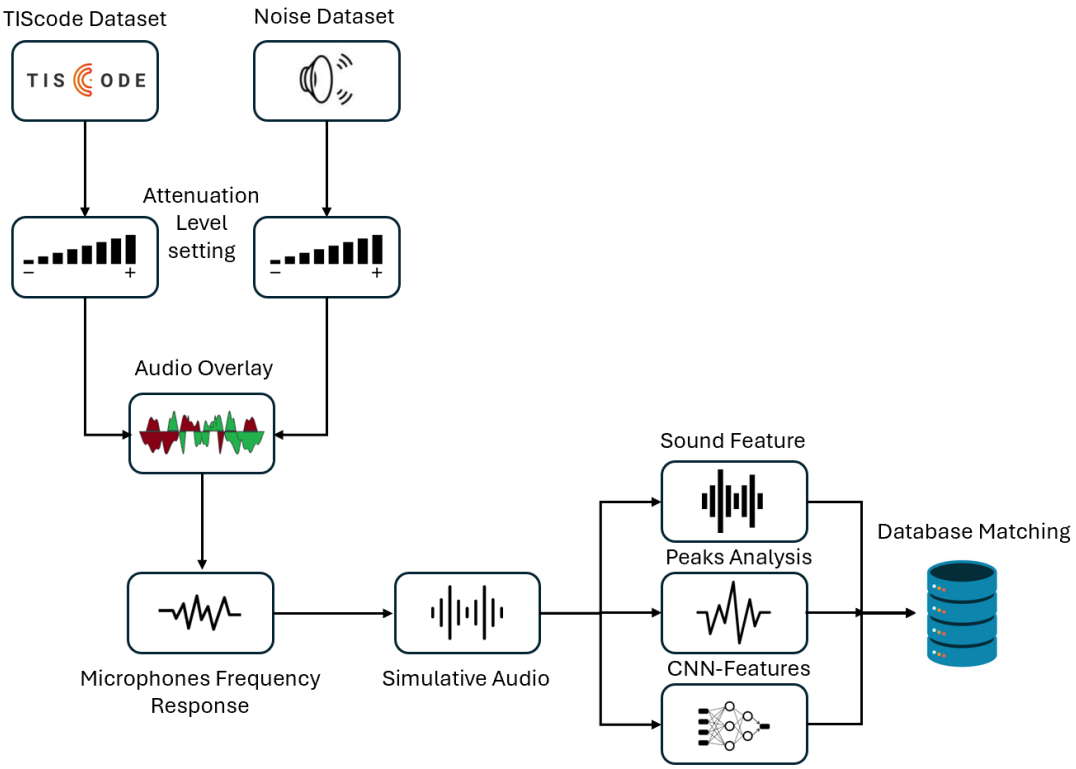


Figure 4.8: TIScode simulation performance system

The simulations in this section follow the algorithm outlined below (and shown in Figure 4.8):

Remark 1. Frequency Peaks analysis is used only to decode the signal, it's not part of the channel coding scheme, as shown in the TIScode pipeline in Figure 1.1.

Remark 2. H represents a general characterization of the frequency response of the smartphone microphone. Typically, in most analysis works related to recognition systems, filters specifically designed to simulate the frequency response of a microphone capsule are employed. In this case, the frequency responses were calculated through measurements conducted in an anechoic chamber, where a 10-second sweep from 20 Hz to 20 kHz on a Mel scale was recorded for each available phone

Algorithm 4.1 Simulations Recognition Algorithm

```
Select a number N of simulations
for each simulation
    Choose a random TIScode from the dataset
    Choose a random noise sample from the noise dataset
    Set an attenuation level in dB for the TIScode
    Set an attenuation level in dB for the noise sample
    Overlay the TIScode and noise
    if H is selected
        Apply the frequency response of the phone's microphone to the overlaid sound
    end if
    Extract features and perform frequency peaks analysis
    Match the audio with the highest-scoring TIScode in the database
end for
Compute the total number of matches
```

[79]. With access to the original audio $x(t)$ and the recorded audio $y(t)$ the Fourier transform was applied to both signals to obtain $X(f)$ and $Y(f)$, allowing for the calculation of the frequency response of the phone as follows:

$$\mathbf{H(f)} = \frac{Y(f)}{X(f)} \tag{4.7}$$

All the data and instrumentation used to calculate these frequency response can be summarized in:

Instrument	**Description**
Anechoic Room	CSC, Via Gradenigo 6/A (PD)
Sound Generation Software	Isotope Rx
Generated Sounds	Sweep in Linear, Log and Mel scale
Sound Length	10 seconds
Frequencies Band	20Hz-20KHz
Speaker	KRK Classic 5
Speaker Distance	1 m
Height	80 cm

Table 4.2: Experimental kit for the Anechoic Chamber Misurations [79]

Phone	**Model**
Smartphone 1	IPhone 13
Smartphone 2	OPPO A54
Smartphone 3	Honor 9X

Table 4.3: Smartphones List

Remark 3. Once the frequency response of the desired phone (selected from those listed in

Table 4.3) is chosen, we transform the "Audio Overlay" file into the frequency domain, obtaining $X(f)$. At this point, we perform the following calculation:

$$Y(f) = H(f) \cdot X(f) \tag{4.8}$$

Subsequently, to obtain the filtered signal in the time domain, we apply the Inverse Fourier Transform to $Y(f)$:

$$\mathcal{F}^{-1}\{Y(f)\} = y(t) = \int_{-\infty}^{\infty} Y(f) e^{j2\pi ft} df \tag{4.9}$$

Remark 4. In the final phase, which involves feature and peak matching, the score obtained from feature matching is added to the score from frequency peak matching. This creates a temporary list of all TIScodes in the database, each with its respective score for both feature and frequency peaks. The TIScode with the highest total score, resulting from the sum of the two, is selected as the best match candidate.

4.2.2 Simulations Results

In this section, we present the various results obtained from the simulations. On average, each simulation takes 15-20 seconds using CUDA, while it takes 40 seconds to a minute when using only the CPU. For each simulation, which involves analyzing 100 different combinations of TIScode and noise, the required time ranges from 25 minutes with CUDA to up to 1 hour and 40 minutes using only the CPU. This is mainly due to the use of CNN features, which take longer to compute compared to traditional features. These calculations will eventually be performed remotely in the cloud at the application level, so the recognition duration will not depend on the capabilities of an individual smartphone.

For the simulations, two different datasets of ambient noises were used:

- *Ambient dataset*: This includes a series of environmental sounds such as rain, thunderstorms, wind, etc. The average SNR of the dataset relative to the TIScode is about -3.75 dB [80].
- *Hospital dataset*: This includes ambient noises from hospital environments, such as people speaking loudly, crying children, the sound of objects being moved, etc. The average SNR of the dataset relative to the TIScode is about -6.5 dB [81].

In Figure 4.9, we can observe the results of a standard simulation, meaning without applying the frequency response of smartphones. On the x-axis, we chose to apply the delta sound level, which represents the difference in attenuation between the TIScode and the noisy audio. At 0 dB, the attenuation of the two audio tracks is not set to zero, while as we move towards negative values, the intensity of the TIScode becomes progressively more attenuated, up to -9 dB, meaning the TIScode's intensity has been reduced by 9 dB while the noisy track remains unchanged. Conversely, as we move towards positive values, there is attenuation of the noisy track, resulting in the TIScode's sound level being higher than the noise.

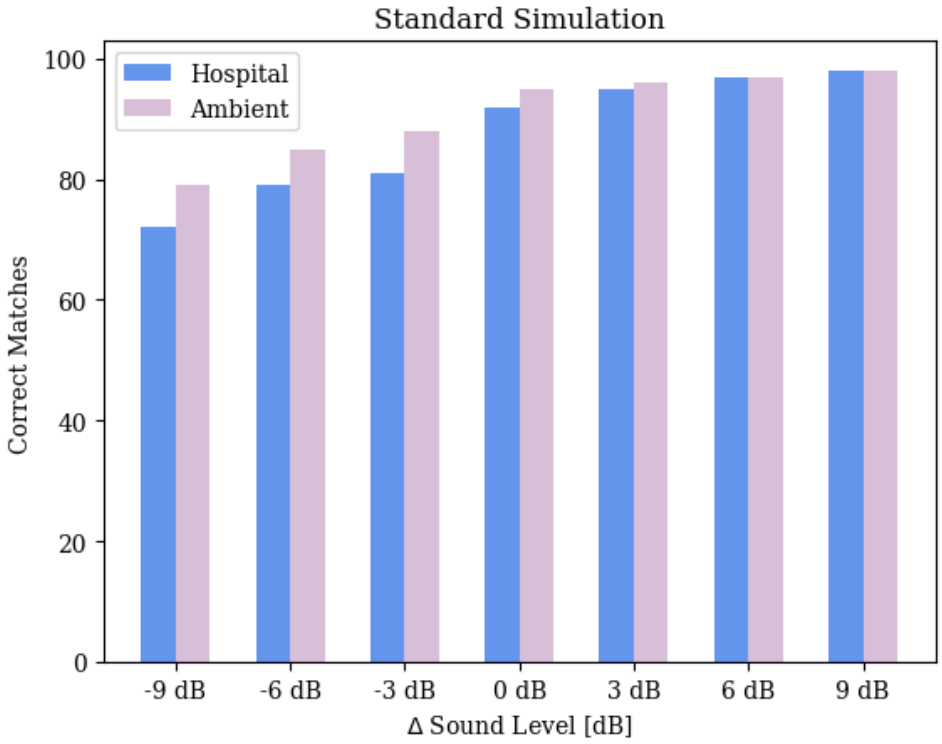


Figure 4.9: Results of standard simulation without the use of smartphones frequency response

We can observe that the impact of the noise in the Hospital dataset is indeed greater compared to that of the Ambient dataset. Nonetheless, we consistently achieve a correct match rate above 50%. Moreover, as the intensity of the TIScode increases relative to the noise, the influence of the selected dataset decreases, and at lower noise levels, the recognition rates become comparable. However, this is a simulation without any distortion applied, and should therefore be considered as a baseline and the maximum achievable limit. In the following graphs, we will analyze scenarios where the frequency responses of different smartphones are applied.

In Figures 4.10, 4.11, and 4.12, we can observe the simulations carried out using the frequency response of the three selected phones for this recognition test: iPhone 13, OPPO A54, and Honor 9X. As shown (and corroborated by Tables 4.4 and 4.5), in the case of the Ambient dataset, the success rate never drops below 50%, and without sound attenuation, it remains above 75% for correct matching. The noisier Hospital dataset, on the other hand, yields worse results, although it never falls below 50% at 0 dB. The results also highlight a dependence on the quality and condition of the microphone capsule. The Honor 9X proves to be the best in terms of correct match percentage (it can be seen that it has the best frequency response in Figure 4.14), while the OPPO A54, a low-end smartphone, performs the worst. These two phones are rarely used and are primarily employed for testing the TIScode, whereas the iPhone 13, although not producing excellent results, outperforms the OPPO, likely due to more frequent use and wear compared to the

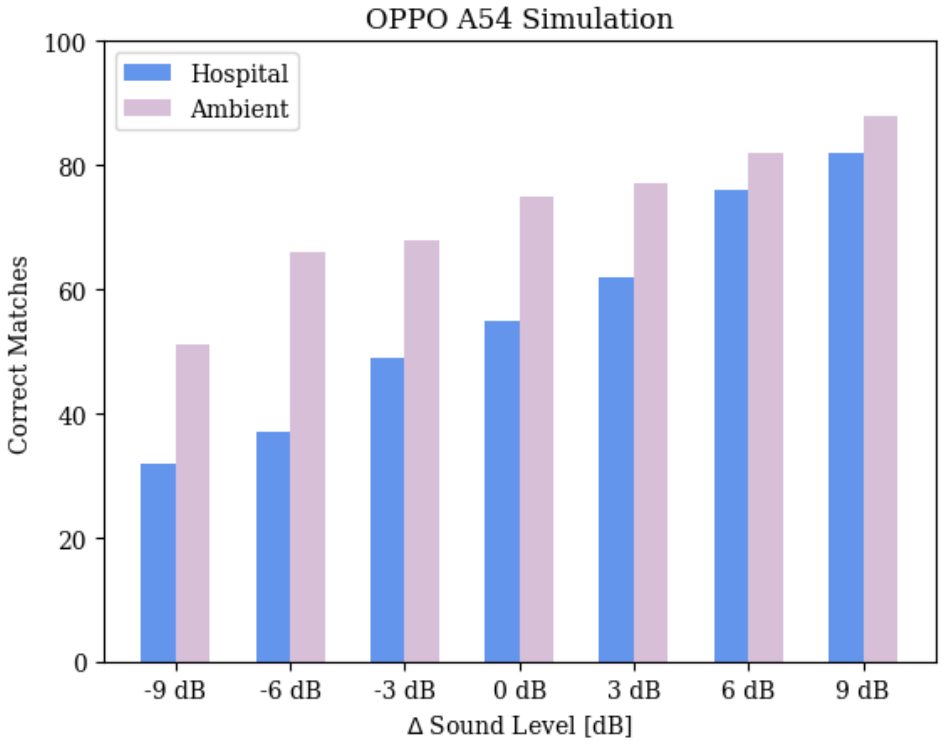


Figure 4.10: Results of simulation with the application of Oppo A54 frequency response

other two devices. Overall, the system delivers solid results, but there is room for improvement. It will likely be necessary to apply noise reduction systems in all cases, although the system remains robust as it stands.

	Ambient Dataset						
	-9 dB	**-6 dB**	**-3 dB**	**0 dB**	**3 dB**	**6 dB**	**9 dB**
Without H	79%	85%	88%	95%	96%	97%	98%
OPPO A54	51%	66%	68%	75%	77%	82%	88%
iPhone 13	50%	52%	63%	76%	85%	92%	95%
Honor 9X	68%	85%	87%	88%	91%	94%	95%

Table 4.4: Correct Matches Percentages for Smartphones in Ambient Datasets

	Hospital Dataset						
	-9 dB	**-6 dB**	**-3 dB**	**0 dB**	**3 dB**	**6 dB**	**9 dB**
Without H	72%	79%	81%	92%	95%	97%	98%
OPPO A54	32%	37%	49%	55%	62%	76%	82%
iPhone 13	42%	49%	52%	56%	69%	81%	83%
Honor 9X	62%	75%	85%	87%	89%	91%	92%

Table 4.5: Correct Matches Percentages for Smartphones in Hospital Datasets

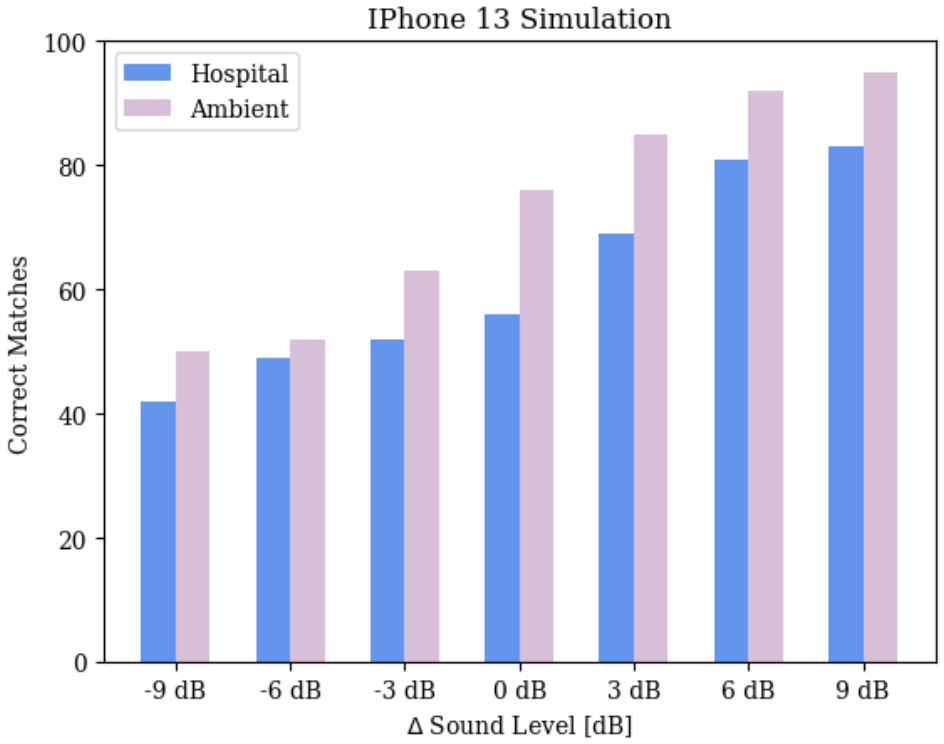


Figure 4.11: Results of simulation with the application of IPhone 13 frequency response

4.3 Field Tests

In this section, we will present a series of real tests conducted to evaluate the recognition system, not only from a simulated perspective but also to demonstrate its effectiveness through concrete results.

The procedure followed is illustrated in Figure 4.7. In this context, a source emits a TIScode, which is then recorded by a smartphone device and analyzed using the methodologies previously outlined.

The instrumentation used is:

- **Source:** Laptop DELL Inspiron 16Plus 7620
- **Receiver:** OnePlus 8T

Scenario 1:

- *Location:* Room in a house
- *PC Volume:* Between 30% and 50%
- *Distance:* 40 cm

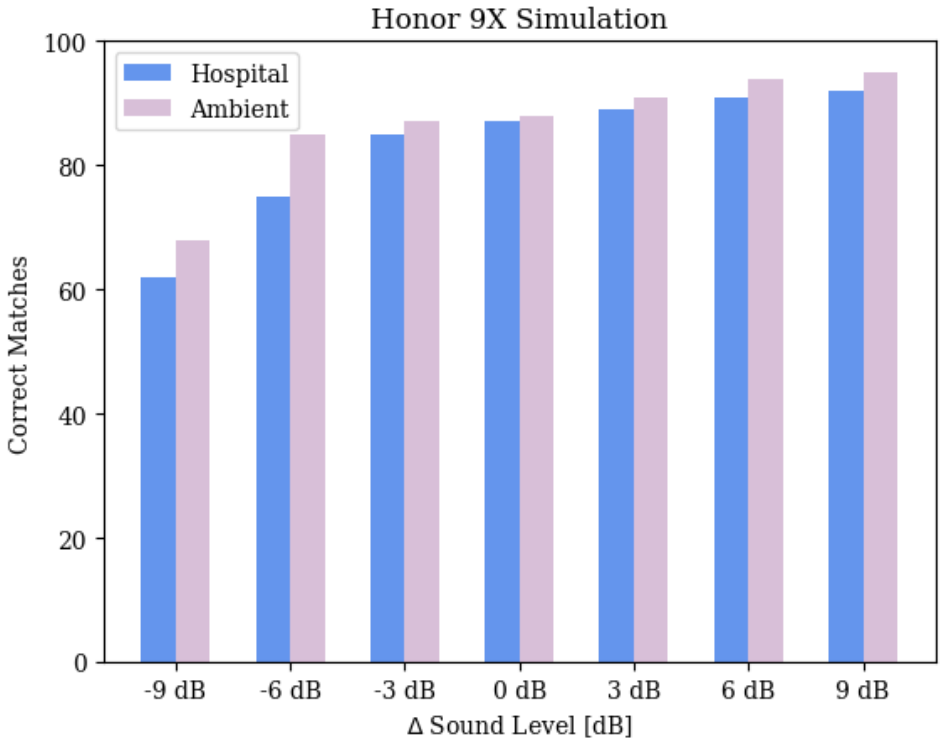


Figure 4.12: Results of simulation with the application of Honor 9X frequency response

- *Background Noise:* None

Scenario 2:

- *Location:* Room in a house
- *PC Volume:* 30%
- *Distance:* 40 cm
- *Background Noise:* Speech occurring 15-20 cm from the smartphone, music played from the PC at half the volume of the TIScode

Scenario 3:

- *Location:* Outdoor area with tables around and people speaking in the background (PALADEI)
- *PC Volume:* 50%
- *Distance:* 40 cm

- *Background Noise:* Continuous background voices at least 2 meters away, with some intermittent voices 30-40 cm from the smartphone

We conducted 50 tests for each of the three scenarios, and the results are presented in Figure 4.13.

In these scenarios, the noise level remained consistent, partly due to the frequency response of both the smartphone and the source, as well as that of the room. However, it was not as high as in the simulations, due to time constraints of finding the correct noise scenario and the inherent challenges of conducting such tests rigorously. Nevertheless, the recognition proved to be reliable in everyday tests, which can also generalize to similar situations where TIScodes might be transmitted. The noise level in such contexts, including radio transmission in a car or TIScode transmission from a computer or television, such as during videos, advertisements, or shows, may be comparable or even lower.

We can observe that the results presented in this section, despite a smaller number of tests, are nonetheless better than the simulations in the previous section. The fact that the phone was not too far from the computer likely contributed to maintaining a certain level of performance. However, this further emphasizes the need for real-world tests in different scenarios to confirm the validity of the results obtained.

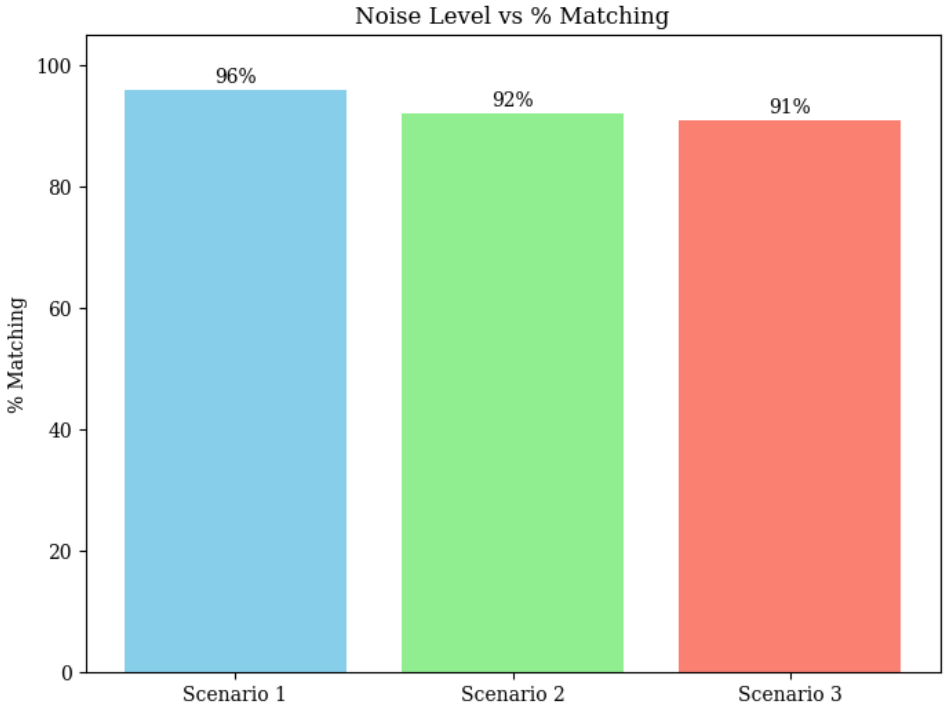
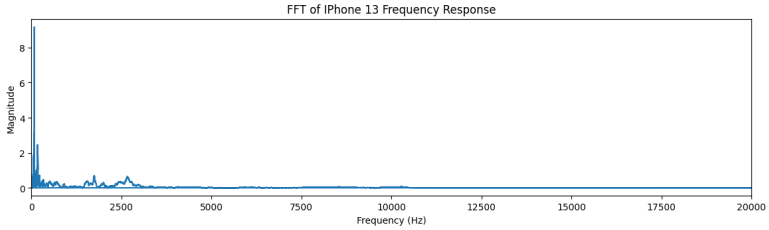
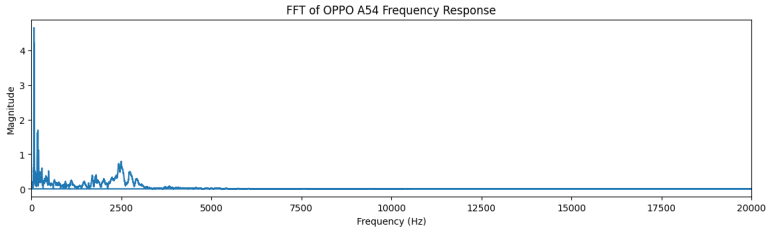


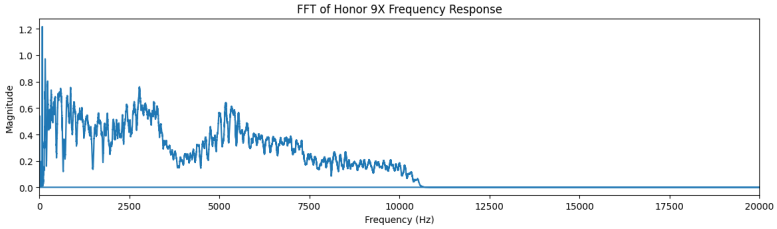
Figure 4.13: Results of the real tests



(a) IPhone 13 Frequency Response



(b) OPPO A54 Frequency Response



(c) Honor 9X Frequency Response

Figure 4.14: Selected Frequency Responses

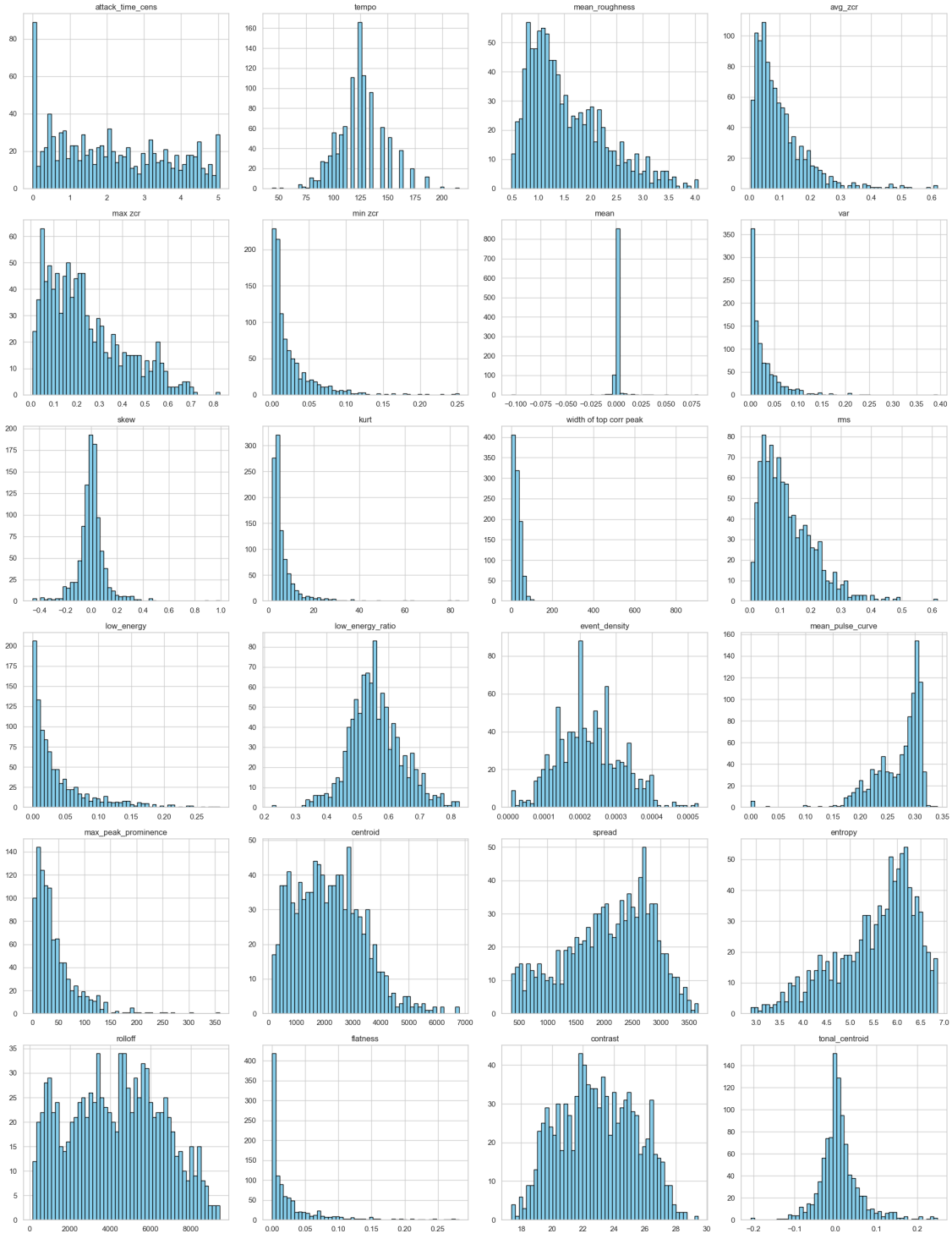


Figure 4.15: Histograms of the analyzed features

5

Conclusions

The objective of this thesis was to study different methodologies that could be suitable for each part of the TIScode pipeline (Figure 1.1).

The first step was to find an automated method for generating short sound messages, and the choice fell on MusicGen, an autoregressive transformer model based on codebook interleaving schemes [45], which, using text prompts, allows us to generate musical tracks with at least 20 bits of degrees of freedom, ensuring the possibility of up to one billion unique generations.

Once the audio tracks were generated, it was crucial to devise a system for channel encoding and decoding. The chosen approach involved using 28 different sound features, including 3 extracted using neural network models (genre, key, top instrument).

These sound features were analyzed and quantized into bit strings. It was verified that, when using a decoding method based on Hamming distance, we obtained a minimum Hamming distance of 31 bits on a dataset of 1000 TIScodes. This result allows us to detect up to 30 errors or correct 15. Combined with peak detection in the frequency domain and their temporal distances, this ensures a reliable outcome.

The validity of the channel coding and recognition system can be confirmed through the various simulations conducted. It is highlighted how performance is highly dependent on the quality and condition of the microphone used for recording. However, the results remain consistent, ensuring that we never fall below a 55% success rate in conditions without attenuation of either the TIScode or the noise. Significantly more promising results can be observed when the intensity of the TIScode exceeds that of the noise, reaching up to 95% accuracy without the use of any noise attenuation system.

5.1 Future Works

There are still many improvements and techniques that can be implemented in the TIScode system. Given the limited available research on the Internet of Audio Things, there are still numerous scenarios we can explore and various techniques we can experiment with.

First and foremost, this thesis does not implement any noise cancellation or filtering methods during the audio track recognition phase to reduce noise in any way. Therefore, the results are purely evaluations of the system as it has been described. To enhance performance, introducing a system capable of attenuating noise or eliminating potential distortions would certainly be a valuable improvement.

M. Dogra et al. present in [82] a CNN model for performing a noise detection and removal task, using features such as Chromagram, STFT, CQT, and CENS as inputs to the model. These features are among those we have already computed for our short sound messages. This could be beneficial for our system, as we could leverage data already available within the system.

The extraction of statistical features from datasets for Machine Learning is becoming increasingly valuable in various contexts, and these methodologies are also widely applied in the fields of image processing and computer vision [83]. This leads us to consider the possibility of utilizing additional features to characterize our TIScodes, this time by analyzing features and patterns within the spectrogram image, as done in [84]. Z. Mushtaq et al. present a CNN that classifies environmental sounds by transforming audio clips into spectral images, specifically Mel-spectrograms, which visually represent the frequency spectrum of an audio signal over time. The convolutional network processes these images to identify patterns or features that distinguish different classes of sounds.

The Channel coding is still in its early stages and can certainly be improved to make it more efficient. This can be achieved, for instance, by using algorithms like Reed-Solomon, an error-correcting code, which introduce additional redundancy, bits useful for error correction, and it has many applications such as CDs, DVDs, Blu-ray discs, QR codes, Data Matrix and data transmission[85]. Redundancy can be added, for example, by leveraging systems such as audio watermarking, where we embed distinctive sound patterns, undetectable to the human ear, which can later be recognized and used for more efficient and secure decoding [86] [87].

Additionally, more real-world testing on various smartphones is necessary. These tests require significant time and resources, considering the need to assess performance in environments like shopping malls, restaurants, and live events, where conducting this type of testing is currently challenging. However, such testing will be essential to verify the system's functionality in more specific situations compared to those analyzed in Section 4.3 of this thesis.

References

[1] L. Turchet, G. Fazekas, M. Lagrange, H. S. Ghadikolaei and C. Fischione, "The internet of audio things: State of the art, vision, and challenges," *IEEE internet of things journal*, vol. 7, no. 10, pp. 10 233–10 249, 2020.

[2] Y. Xiao, S. Yu, K. Wu, Q. Ni, C. Janecek and J. Nordstad, "Radio frequency identification: Technologies, applications, and research issues," *Wireless Communications and Mobile Computing*, vol. 7, no. 4, pp. 457–472, 2007.

[3] M. Stojanovic, "Acoustic (underwater) communications," *Wiley Encyclopedia of Telecommunications*, 2003.

[4] L. Qian, X. Chi, L. Zhao and A. Chaaban, "Secure visible light communications via intelligent reflecting surfaces," in *ICC 2021-IEEE International Conference on Communications*, IEEE, 2021, pp. 1–6.

[5] L. Badia, P. Casari, M. Levorato and M. Zorzi, "Analysis of an automatic repeat request scheme addressing long delay channels," in *2009 International Conference on Advanced Information Networking and Applications Workshops*, IEEE, 2009, pp. 1142–1147.

[6] C. M. Gussen, P. S. Diniz, M. L. Campos, W. A. Martins, F. M. Costa and J. N. Gois, "A survey of underwater wireless communication technologies," *J. Commun. Inf. Sys*, vol. 31, no. 1, pp. 242–255, 2016.

[7] B. Tomasi, P. Casari, L. Badia and M. Zorzi, "Cross-layer analysis via markov models of incremental redundancy hybrid arq over underwater acoustic channels," *Ad Hoc Networks*, vol. 34, pp. 62–74, 2015.

[8] L. Badia, "Analysis of age of information under sr arq," *IEEE Communications Letters*, 2023.

[9] A. Buratto, B. Yivli and L. Badia, "Machine learning misclassification within status update optimization," in *2023 IEEE International Conference on Communication, Networks and Satellite (COMNETSAT)*, IEEE, 2023, pp. 640–645.

[10] J. M. Jornet, M. Stojanovic and M. Zorzi, "Focused beam routing protocol for underwater acoustic networks," in *Proceedings of the 3rd International Workshop on Underwater Networks*, 2008, pp. 75–82.

[11] L. Nkenyereye, L. Nkenyereye and B. Ndibanje, "Internet of underwater things: A survey on simulation tools and 5g-based underwater networks," *Electronics*, vol. 13, no. 3, p. 474, 2024.

[12] L. Yu, H. Sun, S. Su, H. Tang, H. Sun and X. Zhang, "Review of crucial problems of underwater wireless power transmission," *Electronics*, vol. 12, no. 1, p. 163, 2022.

[13] C.-C. Kao, Y.-S. Lin, G.-D. Wu and C.-J. Huang, "A comprehensive study on the internet of underwater things: Applications, challenges, and channel models," *Sensors*, vol. 17, no. 7, p. 1477, 2017.

[14] M. C. Domingo, "An overview of the internet of underwater things," *Journal of Network and Computer Applications*, vol. 35, no. 6, pp. 1879–1890, 2012.

[15] J. Heidemann, M. Stojanovic and M. Zorzi, "Underwater sensor networks: Applications, advances and challenges," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 370, no. 1958, pp. 158–175, 2012.

[16] O. Kundacina, M. Petkovic, A. Munari, D. Vukobratovic and L. Badia, "Move away from me! user repulsion under proximity-induced interference in owc systems," in *European Wireless 2023; 28th European Wireless Conference*, VDE, 2023, pp. 308–313.

[17] D. R. K. M *et al.*, "Underwater network management system in internet of underwater things: Open challenges, benefits, and feasible solution," *Electronics*, vol. 9, no. 7, p. 1142, 2020.

[18] ATTRACT. "Visible light communication for indoor monitoring. "[Online]. Available: . https://iotpoint.wordpress.com/lifi-tutorial/.

[19] D. C. O'brien, L. Zeng, H. Le-Minh, G. Faulkner, J. W. Walewski and S. Randel, "Visible light communications: Challenges and possibilities," in *2008 IEEE 19th international symposium on personal, indoor and mobile radio communications*, IEEE, 2008, pp. 1–5.

[20] L. U. Khan, "Visible light communication: Applications, architecture, standardization and research challenges," *Digital Communications and Networks*, vol. 3, no. 2, pp. 78–88, 2017.

[21] A. Memedi and F. Dressler, "Vehicular visible light communications: A survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 161–181, 2020.

[22] A. Memedi and F. Dressler, "Vehicular visible light communications: A survey," *IEEE Communications Surveys Tutorials*, vol. 23, no. 1, pp. 161–181, 2021.

[23] H. Haas, E. Sarbazi, H. Marshoud and J. Fakidis, "Visible-light communications and light fidelity," in *Optical fiber telecommunications VII*, Elsevier, 2020, pp. 443–493.

[24] A. Singh. "Li-fi tutorial. "[Online]. Available: . https://iotpoint.wordpress.com/lifi-tutorial/.

[25] A. Petrosino, D. Striccoli, O. Romanov, G. Boggia and L. A. Grieco, "Light fidelity for internet of things: A survey," *Optical Switching and Networking*, vol. 48, p. 100 732, 2023.

[26] L. Turchet *et al.*, "The internet of sounds: Convergent trends, insights, and future directions," *IEEE Internet of Things Journal*, vol. 10, no. 13, pp. 11 264–11 292, 2023.

[27] L. Turchet and C. Fischione, "Elk audio os: An open source operating system for the internet of musical things," *ACM Transactions on Internet of Things*, vol. 2, no. 2, pp. 1–18, 2021.

[28] L. Turchet, "Musical metaverse: Vision, opportunities, and challenges," *Personal and Ubiquitous Computing*, vol. 27, no. 5, pp. 1811–1827, 2023.

[29] M. Favero, C. Schiavo, L. Verzotto, A. Buratto, T. Marchioro and L. Badia, "Strategic cooperation in the metaverse: A game theory analysis with age of information," in *2024 International Wireless Communications and Mobile Computing (IWCMC)*, 2024, pp. 1466–1471.

[30] L. A. Hiller and L. M. Isaacson, *Experimental Music; Composition with an electronic computer*. Greenwood Publishing Group Inc., 1979.

[31] D. Conklin and I. H. Witten, "Multiple viewpoint systems for music prediction," *Journal of New Music Research*, vol. 24, no. 1, pp. 51–73, 1995.

[32] C. S. Quintana, F. M. Arcas, D. A. Molina, J. D. F. Rodríguez and F. J. Vico, "Melomics: A case-study of ai in spain," *AI Magazine*, vol. 34, no. 3, pp. 99–103, 2013.

[33] D. Cope, *Experiments in musical intelligence*. AR editions Madison, WI, 1996, vol. 12.

[34] F. Pachet, "The continuator: Musical interaction with style," *Journal of New Music Research*, vol. 32, no. 3, pp. 333–341, 2003.

[35] A. Roberts, J. Engel, C. Raffel, C. Hawthorne and D. Eck, "A hierarchical latent vector model for learning long-term structure in music," in *International conference on machine learning*, PMLR, 2018, pp. 4364–4373.

[36] A. Van Den Oord *et al.*, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, vol. 12, 2016.

[37] C. Payne, "Musenet," *https://openai.com/index/musenet/*, 2019, OpenAI Blog.

[38] Z. Li, F. Liu, W. Yang, S. Peng and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 12, pp. 6999–7019, 2021.

[39] S. Grossberg, "Recurrent neural networks," *Scholarpedia*, vol. 8, no. 2, p. 1888, 2013.

[40] A. Graves and A. Graves, "Long short-term memory," *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.

[41] I. Lezhenin, N. Bogach and E. Pyshkin, "Urban sound classification using long short-term memory neural network," in *2019 federated conference on computer science and information systems (FedCSIS)*, IEEE, 2019, pp. 57–60.

[42] Y. Song, S. Dixon and M. Pearce, "A survey of music recommendation systems and future perspectives," in *9th international symposium on computer music modeling and retrieval*, Citeseer, vol. 4, 2012, pp. 395–410.

[43] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[44] L. Murrone, "Tis-code project: Focus on innovation," *Ogenus S.R.L.*, 2024, https://www.tis-code.com/.

[45] J. Copet *et al.*, "Simple and controllable music generation," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[46] A. Défossez, J. Copet, G. Synnaeve and Y. Adi, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.

[47] J. Du, W. Li, Y. He, R. Xu, L. Bing and X. Wang, "Variational autoregressive decoder for neural response generation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 3154–3163.

[48] N. Benvenuto and M. Zorzi, *Principles of communications Networks and Systems*. John Wiley & Sons, 2011.

[49] B. McFee *et al.*, "Librosa: Audio and music signal analysis in python.," in *SciPy*, 2015, pp. 18–24.

[50] O. Lartillot, "Mirtoolbox," *https://www. jyu. fi/hum/laitokset/musiikki/en/research/coe-/materials/mirtoolbox*, 2008.

[51] C. Harte and M. Sandler, "Automatic chord identifcation using a quantised chromagram," in *Audio Engineering Society Convention 118*, Audio Engineering Society, 2005.

[52] D. P. Ellis. "Chroma feature analysis and synthesis. "[Online]. Available: . 2007/04/21.

[53] G. K. Birajdar and M. D. Patil, "Speech/music classification using visual and spectral chromagram features," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 1, pp. 329–347, 2020.

[54] C. Schörkhuber and A. Klapuri, "Constant-q transform toolbox for music processing," in *7th sound and music computing conference, Barcelona, Spain*, SMC, 2010, pp. 3–64.

[55] A. Manzo-Martinez and A. Camarena-Ibarrola, "A robust characterization of audio signals using the level of information content per chroma," in *2011 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, IEEE, 2011, pp. 212–217.

[56] M. Muller, F. Kurth and M. Clausen, "Chroma-based statistical audio features for audio matching," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2005.*, IEEE, 2005, pp. 275–278.

[57] J. Li, H. Wang, P. He, S. M. Abdullahi and B. Li, "Long-term variable q transform: A novel time-frequency transform algorithm for synthetic speech detection," *Digital Signal Processing*, vol. 120, p. 103 256, 2022.

[58] B. Berlin, M. Sclater and K. Sinclair, *Keys to music rudiments: Textbook*. Alfred Music, 2006, vol. 1.

[59] X. Riley and S. Dixon, "Crepe notes: A new method for segmenting pitch contours into discrete notes," *arXiv preprint arXiv:2311.08884*, 2023.

[60] C. Harte, M. Sandler and M. Gasser, "Detecting harmonic change in musical audio," in *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, 2006, pp. 21–26.

[61] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*. John Wiley & Sons, 1996.

[62] Drishti, *Analysis of zero crossing rates of different music genre tracks*.

[63] D. P. Ellis, "Beat tracking by dynamic programming," *Journal of New Music Research*, vol. 36, no. 1, pp. 51–60, 2007.

[64] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on speech and audio processing*, vol. 10, no. 5, pp. 293–302, 2002.

[65] D.-N. Jiang, L. Lu, H.-J. Zhang, J.-H. Tao and L.-H. Cai, "Music type classification by spectral contrast feature," in *Proceedings. IEEE international conference on multimedia and expo*, IEEE, vol. 1, 2002, pp. 113–116.

[66] S. Dubnov, "Generalization of spectral flatness measure for non-gaussian linear processes," *IEEE Signal Processing Letters*, vol. 11, no. 8, pp. 698–701, 2004.

[67] A. Klapuri and M. Davy, "Signal processing methods for music transcription," 2007.

[68] J. S. Downie, "Music information retrieval," *Annual review of information science and technology*, vol. 37, no. 1, pp. 295–340, 2003.

[69] D. Iakubovskyi, *Building a free advanced music genre classification pipeline using machine learning*.

[70] S. Schneider, A. Baevski, R. Collobert and M. Auli, "Wav2vec: Unsupervised pre-training for speech recognition," *arXiv preprint arXiv:1904.05862*, 2019.

[71] B. L. Sturm, "An analysis of the gtzan music genre dataset," in *Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies*, 2012, pp. 7–12.

[72] S. P. Mohanty, "Musical instrument's sound dataset," *https://www.kaggle.com/datasets/soumendraprasad/musical-instruments-sound-dataset/data*,

[73] D. Iakubovskyi, "Harmony in ai: Building a sample musical instrument classifier with meta's audio transformer," *https://levelup.gitconnected.com/harmony-in-ai-building-a-sample-musical-instrument-classifier-with-metas-audio-transformer-cb1b066f19f8*,

[74] R. Plomp and W. J. Levelt, "Tonal consonance and critical bandwidth," *Journal of the Acoustical Society of America*, vol. 38, pp. 548–560, 1965.

[75] A. Wang, "An industrial strength audio search algorithm.," Jan. 2003.

[76] M. Strauss, "How shazam works," *https://michaelstrauss.dev/shazam-in-python*, Accessed: 2024-09-27.

[77] R. N. Bracewell, "The fourier transform," *Scientific American*, vol. 260, no. 6, pp. 86–95, 1989.

[78] "NumPy Documentation," *https://numpy.org/doc/stable/reference/generated/numpy.argsort.html*, 2024, Accessed: 2024-09-27.

[79] M. Favero and L. Verzotto, "Short sound recognition with a simple audio fingerprinting system," *https://github.com/manuelefavero/DSP*, 2024.

[80] J. H. Solorozano, "Hospital ambient noise dataset," *https://www.kaggle.com/datasets/solorzano/ambient-noise*, 2022.

[81] S. Shuvo, "Hospital ambient noise dataset," *https://www.kaggle.com/datasets/nafin59/hospital-ambient-noise*, 2021.

[82] M. Dogra, S. Borwankar and J. Domala, "Noise removal from audio using cnn and denoiser," in *Advances in Speech and Music Technology: Proceedings of FRSM 2020*, Springer, 2021, pp. 37–48.

[83] R. Hazra, M. Banerjee and L. Badia, "Machine learning for breast cancer classification with ann and decision tree," in *2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, IEEE, 2020, pp. 0522–0527.

[84] Z. Mushtaq, S.-F. Su and Q.-V. Tran, "Spectral images based environmental sound classification using cnn with meaningful data augmentation," *Applied Acoustics*, vol. 172, p. 107 581, 2021.

[85] S. B. Wicker and V. K. Bhargava, *Reed-Solomon codes and their applications*. John Wiley & Sons, 1999.

[86] H.-C. Chen, Y.-W. Chang and R.-C. Hwang, "The modulation method based on reed-solomon code for watermarking," in *2012 9th International Conference on Ubiquitous Intelligence and Computing and 9th International Conference on Autonomic and Trusted Computing*, 2012, pp. 633–637.

[87] Y.-Y. Tai and M. Mansour, "Audio watermarking over the air with modulated self-correlation," in *ICASSP 2019*, 2019.