Tafahas

Group Members:
Zahra Rasheed 2113400
Haneen Al-Mabadi 2111251
Hanouf Albarakaty 2110429
Hadeel Almutairi 2117022
Jumana Alhuzali 2111372

Supervisor: Dr. Haneen Himdi

Department of Computer Science and Artificial Intelligence
University of Jeddah
Kingdom of Saudi Arabia

Contents

Chapter 1

1.1 Introduction

In the early 2020s, synthetic voices, known as audio manipulation using artificial intelligence and cloned voices, became prevalent. A study conducted in 2022 by Zaynab Almutairi and Hebah Elgibreen [1] highlights the widespread use of synthetic voices, particularly on mobile devices and computers where the dissemination of misinformation and disinformation are vulnerable to spoofing. Spoofing is a cyber attack aimed at stealing one's identity to access data. This technique uses synthetic voices fueled by an extensive collection of voice recordings transmitted daily over the internet. Given the challenge, attackers use this approach on various domains by fabricating opinions, propaganda, defamation, cybersecurity, and terrorism [1]. Cloning is a significant phenomenon that challenges the authenticity of audio content globally. According to McAfee's Artificial Imposter report, a quarter of adults who surveyed globally experienced voice cloning scams. [2] Synthetic voices enabled by artificial intelligence mask any underlying human trace, posing a grave threat to individuals' identities. AI can create a thin line between natural and synthesized audio, leading to widespread skepticism regarding its credibility. For instance, The American Federal Commission Agency urges the development of multidisciplinary solutions to the rising threat.[3] A detection system is crucial to address the misuse of voice cloning [4]. By understanding the risks associated with voice cloning, we can develop strategies against impersonation or fraudulent activities.

1.2 Problem Definition

The increasing use of AI technology has sparked concerns regarding the manipulation of voices, for purposes. Nowadays people can manipulate voices using AI algorithms, resulting in consequences such as fraud, defamation, and harassment. One major worry revolves around the capability of AI-generated voices to impersonate known personalities or individuals in positions of authority. By taking advantage of the trust and recognition associated with these wrongdoers, gain credibility, and create opportunities for exploitation and harm.

The implications arising from this issue are significant. Scammers can exploit the trust and familiarity people have with celebrity voices to deceive them into revealing information or providing resources, resulting in losses and potential identity theft. Additionally, manipulating voices used for slander and harassment can tarnish the reputation of targeted individuals or organizations. Moreover, this technology can be misused to spread misinformation and manipulate opinion for malicious purposes.
In light of these concerns, our system will provide the ability for people or entities to discover and verify voices, whether the voice is natural or generated.

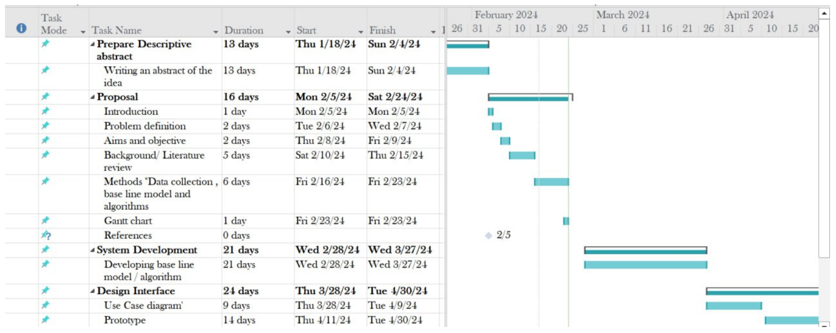
1.3 Aim of the Project

The main aim of this project is to employ an AI model that determines whether the sound is natural or generated using artificial intelligence and enables individuals or entities to discover and authenticate voices. Achieving this is crucial for developing an efficient system.

1.4 Objectives

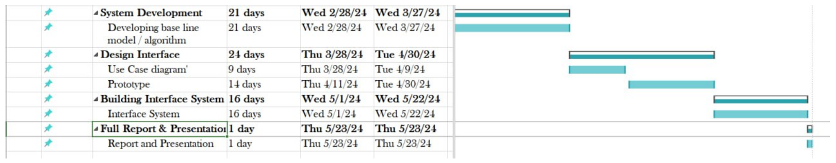
- Improve an AI model that helps authorities differentiate between human and AI-generated voices.
- Employ an efficient and fast application that generates fast and accurate results based on voice only.
- Automate voice recognition using innovative AI methods.
- Utilize the model's classification results to generate a report that alerts authorities when fake audio is detected.
- Utilize comprehensive voice samples that represent human diversity, such as different genders.

1.5 Project Plan



Task Mode	Task Name	Duration	Start	Finish
	Prepare Descriptive abstract	**13 days**	**Thu 1/18/24**	**Sun 2/4/24**
	Writing an abstract of the idea	13 days	Thu 1/18/24	Sun 2/4/24
	Proposal	**16 days**	**Mon 2/5/24**	**Sat 2/24/24**
	Introduction	1 day	Mon 2/5/24	Mon 2/5/24
	Problem definition	2 days	Tue 2/6/24	Wed 2/7/24
	Aims and objective	2 days	Thu 2/8/24	Fri 2/9/24
	Background/ Literature review	5 days	Sat 2/10/24	Thu 2/15/24
	Methods "Data collection , base line model and algorithms	6 days	Fri 2/16/24	Fri 2/23/24
	Gantt chart	1 day	Fri 2/23/24	Fri 2/23/24
?	References	0 days		
	System Development	**21 days**	**Wed 2/28/24**	**Wed 3/27/24**
	Developing base line model / algorithm	21 days	Wed 2/28/24	Wed 3/27/24
	Design Interface	**24 days**	**Thu 3/28/24**	**Tue 4/30/24**
	Use Case diagram'	9 days	Thu 3/28/24	Tue 4/9/24
	Prototype	14 days	Thu 4/11/24	Tue 4/30/24

Figure 1: An illustration of the schedule and timeline of various tasks



System Development	**21 days**	**Wed 2/28/24**	**Wed 3/27/24**
Developing base line model / algorithm	21 days	Wed 2/28/24	Wed 3/27/24
Design Interface	**24 days**	**Thu 3/28/24**	**Tue 4/30/24**
Use Case diagram'	9 days	Thu 3/28/24	Tue 4/9/24
Prototype	14 days	Thu 4/11/24	Tue 4/30/24
Building Interface System	**16 days**	**Wed 5/1/24**	**Wed 5/22/24**
Interface System	16 days	Wed 5/1/24	Wed 5/22/24
Full Report & Presentatio	**1 day**	**Thu 5/23/24**	**Thu 5/23/24**
Report and Presentation	1 day	Thu 5/23/24	Thu 5/23/24

Figure 2: An illustration of the schedule and timeline of various tasks

Chapter 2

2.1 Background

Voice Synthesis is a technique that uses data from human speech, whether text data or audio data, to train models. The trained models can generate realistic speech that mimics human speech based on the analyzed patterns of human speech [5]. Fake Speech Detection is the technology and methods used to recognize audio that is not real and has been artificially generated or modified. The main goal of speech detection is to differentiate between human speech and generated AI speech, which uses techniques such as voice cloning and text-to-speech [6] [7]. Fake speech recognition techniques began to develop around 2010, and the tremendous development in deep learning and fake speech detection has continued until these years. The potential applications of this technology to create deep fakes are indeed breathtaking. For example, doctoring an audio recording of some public figure saying something incendiary in order to create chaos or public outrage [8]. This is one of the many issues looming with deepfakes, and, in general, with audio content created by such means. Nowadays, such a threat to public safety turns even more easily into general activities with the availability of easily accessible open-source programming [8]. Unregulated and potentially illegal uses of deepfake and voice cloning technology are hence made much more easily to the detriment of individuals and society [8]. However, in the past few years, an ever-increasing worrying concern has been noticed that these cloned voices could be used for more unethical and possibly harmful activities [8]. Also, attackers have manipulated the voice-cloning technology for serious cyber security attacks. Impersonation means pretending to be another person, whereby voice cloning technology makes the life of cybercriminals very easy during the impersonation of a person with an aim of perpetuating fraud [8]. This is because it's possible to make a false sound like the targeted victim in phone calls, and one will sound like the victim [8].

2.2 Literature Review

By exploring the research area in greater detail shedding light on relevant studies, theories, and approaches related to the topic under consideration, the literature review expands the scope of the background section. The studies by the authors Weng et al [5], and Qiu et al [9], stated that Speech Recognition and Speech Synthesis are an integral, important, and critical part of modern communication systems, virtual assistants, and voice control devices. The recent development in the field of deep learning has led to a revolution in these fields due to the growth of the misuse of synthetic speech or "fake speech". The study of Qiu et al [9], focused on the combination of two models CNN and TDNNs for building an English speech recognition model. The CNN model was trained separately from the TDNN models. Both models were trained on speech data, and the outputs of the two models were combined. The results of the combined model of Qiu et al [9], prove that the combined model performs better than CNN and TDNN in speech recognition. By integrating different models, researchers could benefit from the strengths of each model and overcome the weaknesses and limitations producing more powerful and accurate models such as the combined model in Qiu et al [9].

On another hand Subramani et al [10] said that every synthesis model is an "attack vector" in the adversarial synthetic voice detection. The study by Subramani et al [10] focus on proposing two models of lightweight convolutional networks, EfficientCNN and RES-EfficientCNN, for synthetic speech detection. The authors Subramani et al [10] prove that EfficientCNN, even for a resnet18 model, exhibits significantly better computational efficiency. These models achieve high detection performance. The transfer learning proposed in Subramani et al [10] makes it possible to learn tasks with a limited amount of data, and often outperforms pure supervised approaches. They study neural-to-neural speech synthesis, neural-to-non-neural voice conversion, and neural-to-non-neural speech synthesis to a unit selection, covering many paradigms.

Fake speech detection is a major challenge. Detecting fake speech has become imperative as the capabilities of artificial intelligence technologies, such as voice cloning and speech synthesis, continue to advance. Fake speech has many potential risks, including misinformation, identity theft, and fraudulent activities. The model is trained to distinguish between real and fake speech in both studies by Scardapane et al [6], and Malik et al [7]. The study by Malik et al [7], depends on the advantages of deep learning techniques, such as (RNNs) recurrent neural networks and (CNNs) convolutional neural networks. Studies by Scardapane et al [6], and Malik et al [7] agreed that RNNs are suitable for modeling sequential data, capturing temporal dependencies, and analyzing audio signals, as they can distinguish between real and fake audio using temporal dynamics. ANN have shown impressive results in capturing high-level features in Scardapane et al work [6]. The results in studies by Qiu et al [9], Scardapane et al [6], and Malik et al [7] shows the power of the deep learning model in detecting fake speech, achieving high performance in terms of accuracy and reliability. The performance evaluation results in Scardapane et al [6] showed the effectiveness and accuracy of RNNs in detecting voice spoofing attacks, as they achieved accurate and robust performance.

A study by Logan Blue et al was published about Detecting Audio DeepFakes Through Vocal Tract Reconstruction by using techniques from articulatory phonetics [11]. They found a method that addresses the issue of reliably identifying deepfakes by analyzing the biological limitations of human speech. Also, by analyzing the fluid dynamic models' prediction of the human vocal tract's shape during speech production [11]. Deepfakes have vocal-tract designs that are inconsistent with human anatomy, probably because of their lack of biological constraints. Due to the differences between deepfake audio and real human speech, the research has developed an efficient detecting technique. It has been shown a high level of precision (99.9%) and recall (99.5%) is delivered in determining the capability of deepfake audio samples [11]. Previous studies have concentrated on minor variations in the spectrum that are inaudible to the human ear. According to this study [11], speech patterns that are restricted by biology need to be considered when identifying deepfakes.

Another study was published by Zaynab Almutairi and Hebah Elgibreen about A Review of Modern Audio Deepfake Detection Methods [1]. The study showed that there are two audio clip segments usually used in AD detection, the pre-processing segment and the audio clip after the segment has been further transformed to the proper audio features, especially Mel-spectrograms [1]. After extracting these features, the detection model applies them to train itself for the prediction of the uniqueness of the audio or any other. There are associated trade-offs between accuracy and computing complexity. The methods for AD detection have been compared based on the performance of the methods towards some metrics, such as accuracy, precision, recall, and the F1 score. This analysis helps find a better scope for the improvement in diverse approaches and helps analyze the effectiveness.

Another study was published by Hamza et al about Deepfake audio detection via MFCC features using machine learning [12]. Spectrograms provide a representation of audio signals, based on audio signals, it identifies anomalies and inconsistencies in the audio [12]. Some studies have focused on machine learning algorithms that are used to extract features from audio signals. One of the most common features is MFCCs Mel-frequency cepstral coefficients, which capture the spectral characteristics of audio signals [12]. The MFCC features that are extracted from audio signals and uses two models: support vector machines (SVMs) and random forests, and then the audio is classified based on whether it is real or fake [12]. MFCC has proven its effectiveness which makes it suitable in classification processes [12]. Other challenges to the determination of fakes in such media are in the form of strong detection models that may show fakeness in audio with the noises from real-world environments [1]. Guide the domain of future research highlighting the development of methods in support of the enhancement in AD detection, performance, and knowledge gaps bridging [1].

Moreover, another study was published by Mvelo Mcuba et al about The Effect of Deep Learning Methods on Deepfake Audio Detection for Digital Investigation [13]. They found that the characteristics of the audio file are represented through methods like Spectrogram, Mel-spectrum, Chromagram, and Mel-Frequency Cepstral Coefficients (MFCC) [13]. The authors compared these techniques on the grounds of the average accuracy of every one of them in identifying deepfake audio. Based on the results from the experiments, the best-performing architecture is VGG-16 through this work when employed for MFCC image feature, while a custom architecture outperforms it when applied for the Chromagram, Spectrogram, and Me-Spectrum image features. Results show a variation of deep learning strategies that are effective in deepfake audio detection and further contribute to forensic investigation [13].

Another study was published by Robin San Roman et al about the Proactive Detection of Voice Cloning with Localized Watermarking [14]. The fact that such a signal can be embedded within the generated audio is through the technique of imperceptible watermarking [14]. The signal must be in such a way that it is not perceptible to the human ear but can be easily detected by using certain algorithms. Most of the deep learning audio watermarking methods available relate to multi-bit approaches [14]. Current watermarking methods have limitations, especially regarding detection and localization. Given these constraints, a new localized speech watermarking approach named ”AudioSeal” is proposed. AudioSeal introduces a generator/detector architecture co-trained with a localization loss to permit sample precise watermark detection. The training of the detector is to recognize fake speech snippets within longer clips of synthesized speech and is achieved by random masking of the watermark inside different segments [14]. This work has introduced a perceptual loss inspired by aural masking that improves the imperceptibility of the watermark signal. AudioSeal gives a high level of precision detection at the sample level and much better behavior than the watermarking models existing in computing speed. Probably, further research goes toward making AudioSeal functionally and performance-wise implemented in real-life scenarios, thus interfacing it with speech recognition platforms [14].

Chapter 3

3.1 Methodology

To achieve our objectives, we proposed the idea of ”Tafahas”. We plan to enable any user to examine any audio clip to determine whether it is natural or artificially generated using artificial intelligence. Subsequently, a report will be generated and submitted to the relevant authorities.

3.2 Data Collection

To build a very large dataset, we will construct and gather diverse data, including a variety of voices (natural and synthetic, where synthetic refers to voices generated using artificial intelligence and cloned voices), dialects, and linguistic variations. We ensure the inclusivity of speakers of both genders and various age groups to avoid biases.

We have found some software for generating voices using artificial intelligence as well as some voice cloning software, all falling under synthetic voices, such as Tacotron [15], and WaveNet by Google DeepMind [16], and Microsoft Azure Speech. We explored two public audio datasets from DeepFake WaveFake [17] and ASVspoof 2019 [18]. We will perform labeling, assigning distinct labels to audio samples to denote their nature as either natural or AI-generated. Preprocessing will be performed as needed.

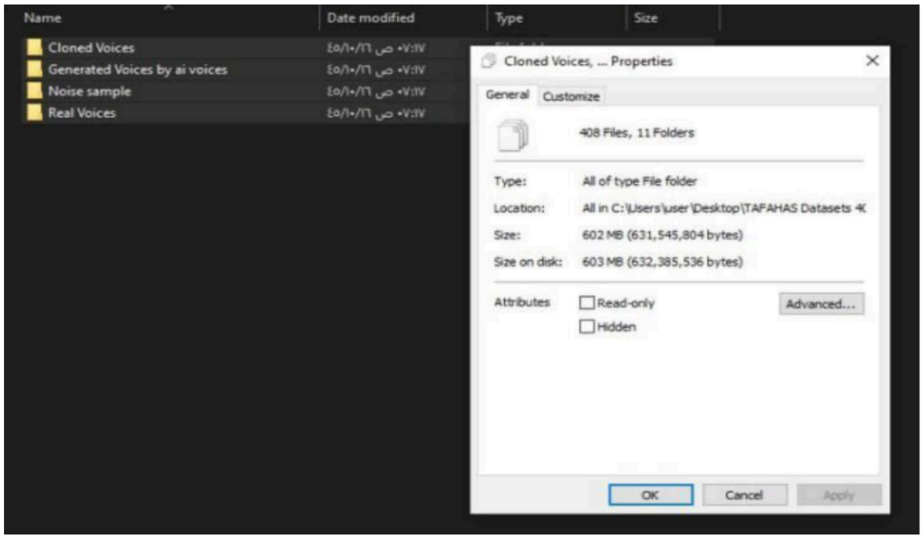
The data collection phase represents a pivotal juncture in our project methodology, characterized by meticulous organization and the creation of a tailored dataset designed to distinguish between genuine voices and those manufactured through deepfake technology. This dataset was constructed by us through rigorous organization and assembly. We designed our dataset from the outset to capture the nuanced diversity of human voices.

3.3 Classification Framework

At the outset of the project, as we aim to classify genuine voices from artificial ones using deepfake techniques, such as those generated by text-to-speech (TTS) systems based on artificial intelligence or cloned voices created through voice cloning technology. In the context of your project, it is important to consider both composite voices, including those generated by artificial intelligence (TTS), and cloned voices as potential examples of deepfakes. These composite voices cannot be distinguished from genuine human voices, posing a significant challenge for voice recognition and authentication systems. We have defined a classification framework comprising three distinct categories: genuine voices, cloned voices, and generated voices (text-to-speech conversion) as a foundational framework. This classification serves as a cornerstone for our subsequent analysis efforts and model development.

3.4 Creation of the Dataset and its Specifications

Creation of the Dataset and its Specifications With steadfast commitment to quality and relevance, we embarked on creating a customized dataset from the outset. Adhering to meticulously designed specifications for each category, we formulated a diverse set of voices, encompassing both genders with 65 voices each, totaling 130 voices per category. Each persona within the dataset was endowed with 13 distinct voices, imbued with seven emotional states and situated within six varied scenarios as neutral, calm, happy, sad, angry, fearful, disgust, and surprised speeches.[19] It is noteworthy that the dialects were diverse, including American and British accents. This comprehensive endeavor culminated in the creation of 390 voices, complemented by an additional 18 voices enriched with specific noise attributes, such as Background, Distortion, and Hard Clipping noises. Resulting in a total of 408 meticulously coordinated audio samples As shown in Figure 3. We work on three categories, each differing slightly in its methodology.



Name	Date modified	Type	Size
Cloned Voices	٤٥/١٠/٢٦ ص ٠٧:١٧		
Generated Voices by ai voices	٤٥/١٠/٢٦ ص ٠٧:١٧		
Noise sample	٤٥/١٠/٢٦ ص ٠٧:١٧		
Real Voices	٤٥/١٠/٢٦ ص ٠٧:١٧		

Cloned Voices, ... Properties

408 Files, 11 Folders

Type: All of type File folder
Location: All in C:\Users\user\Desktop\TAFAHAS Datasets 4(
Size: 602 MB (631,545,804 bytes)
Size on disk: 603 MB (632,385,536 bytes)
Attributes ☐ Read-only ☐ Hidden

Figure 3: Tafahas Dataset

In the Category of Real Voices: Publicly available audio data free of specific publication restrictions was utilized. The data comprised recordings of audiobooks in various contexts sourced from digital libraries [20], as well as excerpts from radio broadcasts [21]. It is worth noting that not all audio samples were pristine; we intentionally included audio samples imbued with specific noise characteristics. It is noteworthy that the audio clips were collected in .mp3 format.

Playht Software: In this study, synthetic voices(generated by AI and cloned voices) were generated using Playht Software, a user-friendly web-based platform incorporating AI-generated voice and cloned voice technologies.[22] Playht stands out as an innovative and advanced tool, boasting a diverse selection of languages, accents, and emotions for voice synthesis. Its robust features and capabilities render it invaluable in the realm of speech synthesis. With an intuitive interface and streamlined workflow, Playht is particularly accessible to novices in this domain.[22] While exploring alternative software options such as Elvenlabs[23], MurfAI[24], and ResambleAI[25], Playht[22] emerged as the most suitable for our project due to its provision of emotions, which is a crucial component for our dataset.

In the Category of Cloned Voices: The dataset was meticulously crafted from publicly available audio data sourced primarily from YouTube[26].This data encompassed a diverse range of speech patterns and accents, ensuring diversity and representativeness in our cloned voices. It is worth noting that not all audio samples were pristine; we intentionally included audio samples imbued with specific noise characteristics. Each audio clip was carefully cataloged and stored in .mp3 format to maintain consistency and accessibility across various platforms.

In the Category of Generated by AI Voices: The software we offer is paid and designed for personal use, allowing you to effortlessly convert text into speech. Moreover, we have ensured that the formatted sentences within this dataset express a wide range of emotions, accompanied by random stories to provide a comprehensive representation of emotional states. It is worth noting that not all audio samples were pristine. We intentionally included samples with specific noise characteristics, providing a realistic and immersive experience. All audio clips have been collected in mp3 format for easy access and use.

3.5 Data Sources and Acquisition Methods

For the Category of Real voices: By leveraging a multitude of sources, including LibriVox [20], Archive.org [21], and proprietary repositories, we retrieved audio samples that meticulously cover various types and scenarios. Despite exploring platforms such as FreeSound.org [27] and Voices.com [28], they were not utilized in the data collection phase. Following the gathering of voices, the audio samples underwent meticulous enhancement and editing processes using software such as WavePad Audio Editor[29] and Audacity[30], ensuring their compliance with predefined specifications.

For the Category of Cloned Voices: Our process began with sourcing audio from YouTube[26], a readily available platform offering diverse content. We then used Audacity[30] to pinpoint and extract specific segments from these audio files. Once we isolated the desired portions, we employed Playht software[22] to transform the original audio into cloned voices. This straightforward approach ensured that our dataset of cloned voices was derived from varied sources on YouTube[26] and meticulously processed using Audacity [30] and Playht software [22], resulting in a versatile resource for voice cloning. Passing both genders with 65 voices each, totaling 130 voices per Cloned Voices category.

For the Category of Generated by AI Voices: Our investigation involved employing various software tools and utilizing multiple sources for analysis, making the most of their unique strengths. Notably, integrating ElevenLabs [23] was crucial for selecting diverse age demographics, strengthening our data selection process. We decided to use PlayHt [22], which covers our specification. Actively participating in the program helped us generate extensive data, allowing us to organize it efficiently and ensure comprehensive representation across different demographics, including age, gender, and accent. Passing both genders with 65 voices each, totaling 130 voices per generated voices category.

After building our data and organizing it into three classes, we proceed to prepare our data for training machine learning (ML) and deep learning (DL) models. This involves following these steps in sequence: Preprocessing and Feature Extraction.

3.6 Preprocessing Step

- Convert Audio Files:

Convert our audio clips from MP3 to a lossless format like WAV. This conversion helps preserve audio quality and ensures consistency across all files

- Standardize Audio Properties:

Ensure all audio clips have consistent properties, including sample rate, bit depth, and duration

Sample Rate: Adjust the sample rate to a standard value, such as 44.1 kHz, to maintain uniformity.

Bit Depth: Optimize the bit depth to ensure all clips have the same quality and compatibility such as 16-bit.

Duration: Standardize the duration of each audio clip for consistency. We set a minimum duration of 18 seconds after reviewing similar datasets to maintain compatibility and preserve original characteristics.

Channels: Consistent 2-channel audio ensures immersive sound and clear directionality.

3.7 Feature Extraction

"In order to scrutinize the distinctive attributes of our voice collection, it's imperative to extract pertinent features from the audio dataset. Feature extraction holds significant importance in contemporary AI and deep learning endeavors. Within this research, we've diligently extracted key features from the dataset to facilitate subsequent feature engineering processes. Notably, we've isolated essential features such as Mel-Frequency Cepstral Coefficients (MFCCs) and Spectral Centroid for future analyses. Additionally, Spectral Contrast has been extracted to explore the dataset, serving as an investigative tool rather than a component for model training. It's noteworthy that in future investigations, we may explore the integration of Spectral Contrast and other extracted features into our modeling framework as our comprehension of the dataset evolves"[31].

Mel-Frequency Cepstral Coefficients (MFCCs): derived through a non-linear mel-scale transformation emphasizing lower frequency components over higher frequency components [32], are highly adept at capturing the distinctive features of diverse audio signals by computing coefficients that reflect the short-term power spectrum of sound [33].

This transformation, inspired by the response of the human auditory system to sound, encapsulates both frequency and temporal traits of audio signals. Analogous to how X-rays unveil the internal structure of an object, MFCCs offer valuable insights into the inherent acoustic properties of audio signals, serving as a faithful representation of the vocal tract responsible for sound production [34].

Additionally, MFCCs encapsulate pertinent vocal characteristics, such as phonetic attributes like timbre and pitch, facilitating the discrimination between different voice types. They excel in discerning varied speech sounds amidst background noise and fluctuations in speaking styles [35]. To discriminate among distinct voice types, encompassing real, cloned, and artificially generated voices as depicted in Figures 4, 5, and 6 respectively, we leveraged the librosa library in Python, a widely utilized toolkit for audio analysis and feature extraction [36].

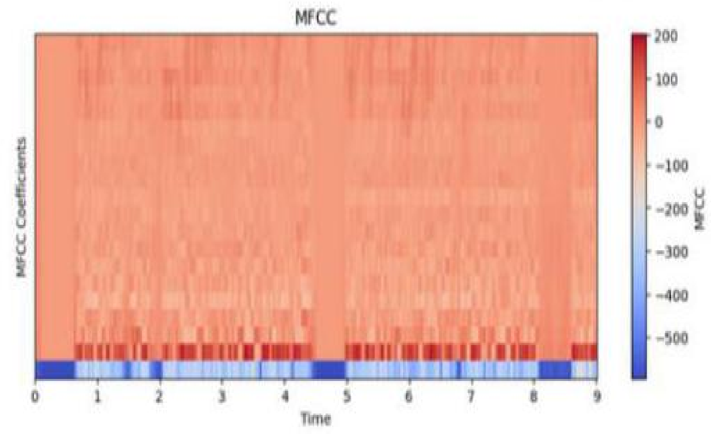


Figure 4: Real Voice

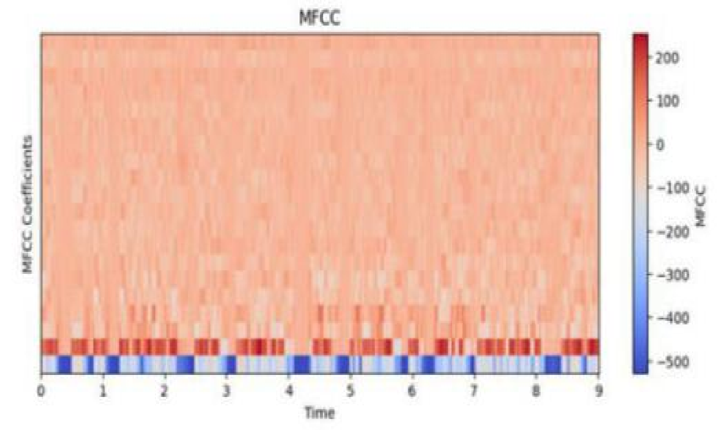


Figure 5: Cloned Voice

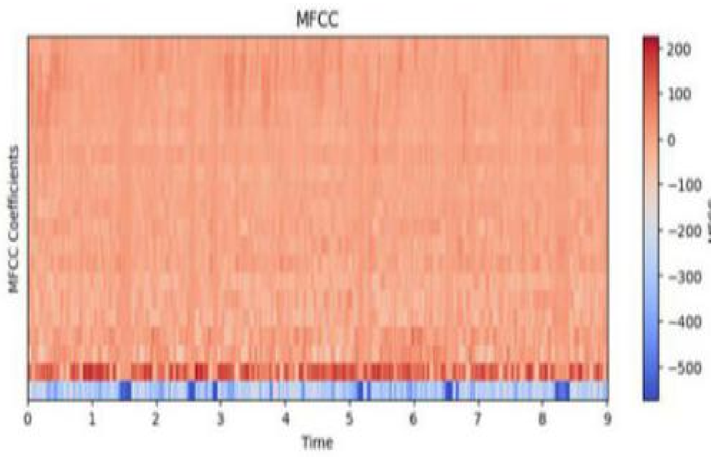


Figure 6: Generated by AI Voice

Spectral Centroid: Is a feature extracted from audio signals that represents the center of mass of the power spectrum, providing valuable insights into the frequency distribution of the signal. It is calculated as the weighted mean of the frequencies present in the signal, with each frequency component weighted by its magnitude. Spectral Centroid is a useful descriptor for analyzing the spectral characteristics of audio signals, as it indicates where the majority of the signal's energy lies along the frequency axis[37]. The Spectral Centroid feature is particularly adept at capturing the overall spectral shape of an audio signal, providing information about its tonal characteristics and timbral qualities. Higher values of Spectral Centroid indicate a greater concentration of energy towards higher frequencies, while lower values suggest a dominance of lower frequencies. Spectral Centroid can be employed in various audio processing tasks, including sound classification, music genre recognition, and speech analysis[38].

Spectral Centroid serves as a fundamental component in the realm of audio signal processing, providing indispensable insights into the spectral attributes of audio signals and fostering a deeper grasp of their acoustic characteristics As shown in Figure[7], Figure[8] and Figure[9].

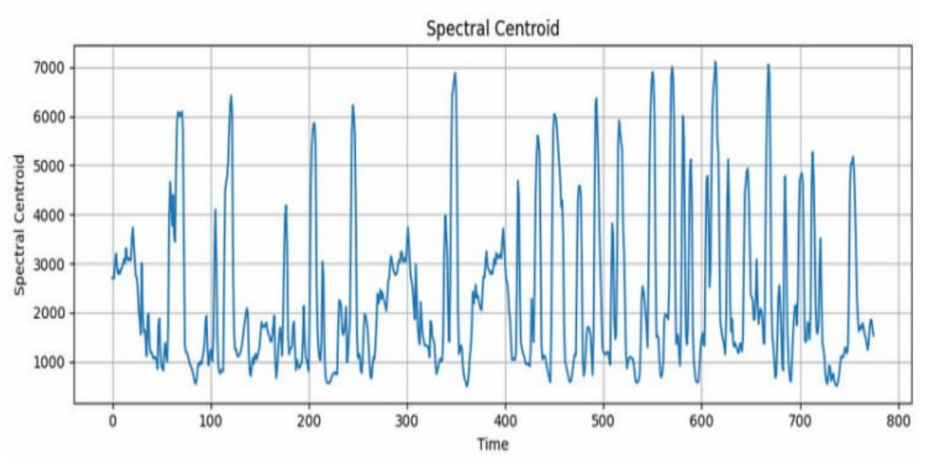


Figure 7: Real Voice

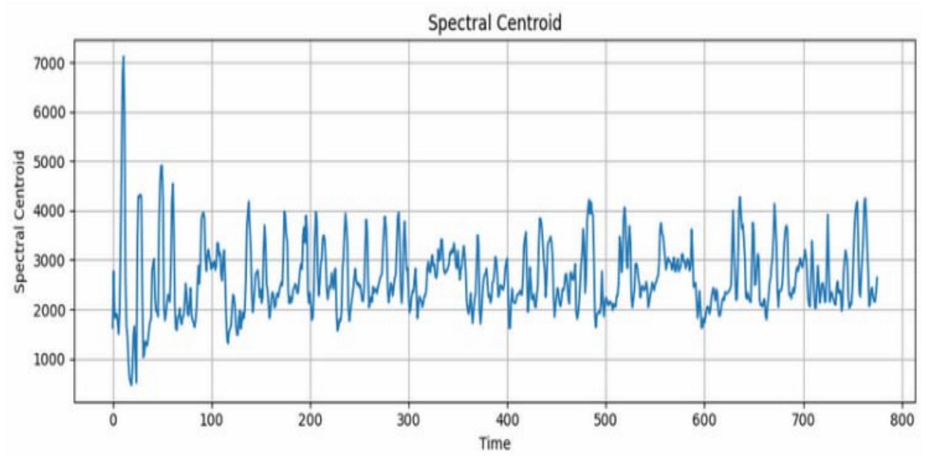


Figure 8: Cloned Voice

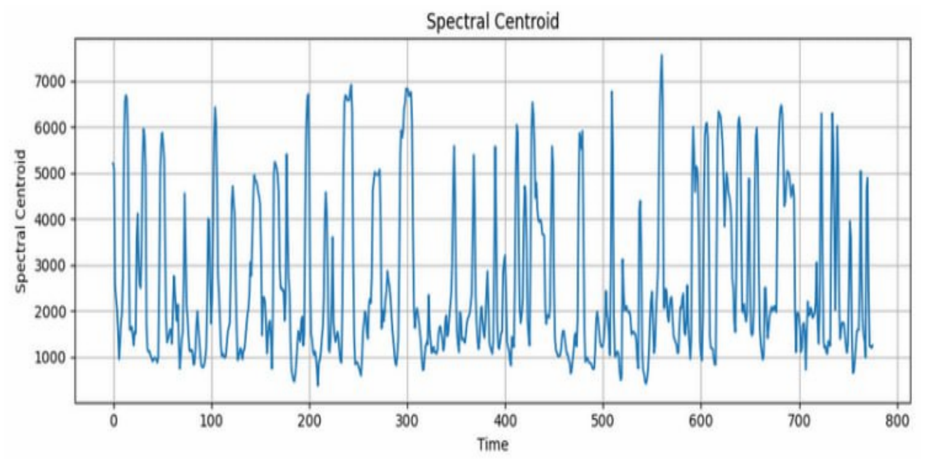


Figure 9: Generated by AI Voice

Spectral Contrast: Analysis is a technique used to extract acoustic features from audio signals, capturing the differences in spectral content across different frequency bands[39]. It provides valuable information about the energy distribution in an audio signal and is effective for distinguishing between different sounds. In order to distinguish between different types of voices, including real voices in Figure 10, cloned voices in Figure 11, and artificially generated voices in Figure 12, we utilized the librosa library in Python, one of the most commonly used libraries for audio analysis and feature extraction techniques[40].

This combination of techniques enhances the ability to differentiate and classify various types of voices based on their acoustic characteristics. By employing spectral contrast analysis and utilizing the librosa library, nuanced variations in energy levels across the spectrum can be measured, enabling the differentiation of speech signals from background noise.Visualizing the spectral contrast offers insights into the energy distribution across frequency bands over time, facilitating tasks such as speech analysis, classification, and enhancement [39]. Spectral Contrast, Emphasize Harmonic Differences Spectral contrast highlights the difference in energy between the peaks and valleys in different frequency bands of the audio signal, aiding in distinguishing between different voice types. This information can be leveraged to improve speech processing algorithms, enhance speech recognition accuracy, and support the development of robust speech-based applications. Analyzing and interpreting the spectral contrast, alongside the use of the librosa library, allows researchers and practitioners to gain a deeper understanding of the acoustic properties of the audio and make informed decisions in various speech-related domains.

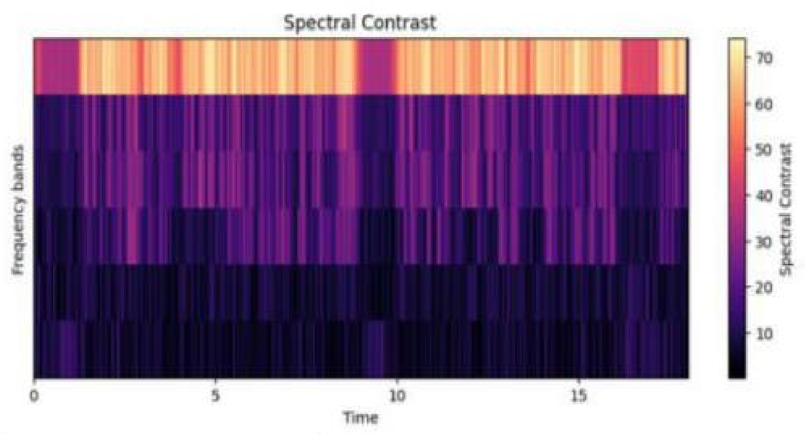


Figure 10: Real Voice

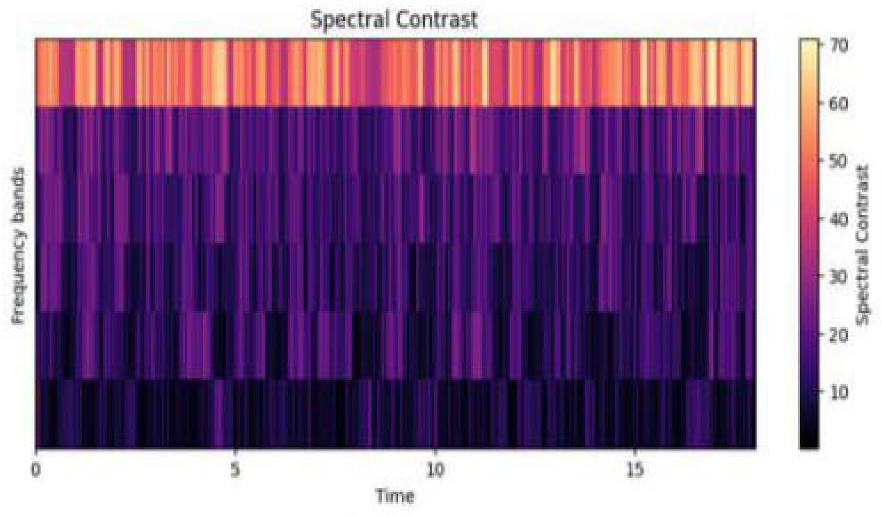


Figure 11: Cloned Voice

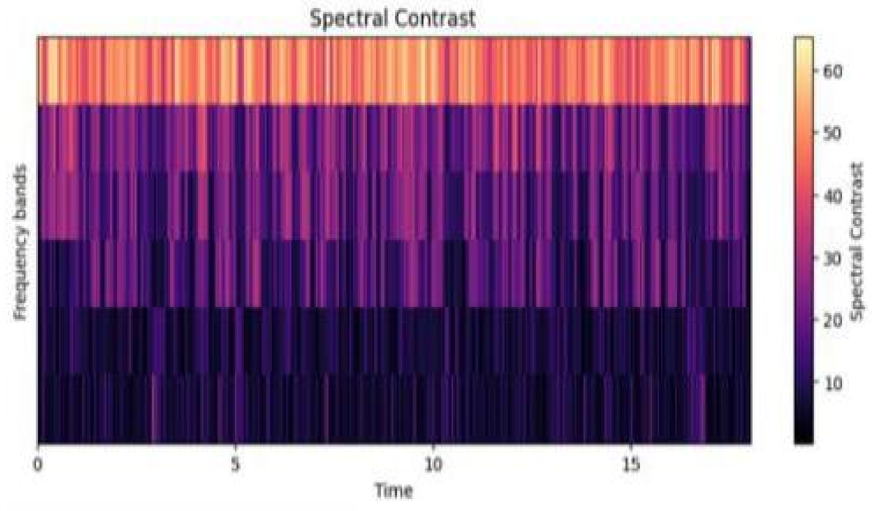


Figure 12: Generated by AI Voice

3.8 Challenges and Mitigation Strategies

Parameter Optimization:
During the audio clips normalization, one of the key challenges encountered was optimizing parameters such as the sample rate, bit depth, and channels. These parameters significantly impact audio quality and can affect the effectiveness of subsequent processing steps.

Sample Rate Optimization:

The sample rate plays a crucial role in accurately capturing sound waveforms. A higher sample rate results in better sound quality by capturing more data points per second. Typical values range from 8 kHz to 48 kHz or higher. For our purposes, we experimented with three different sample rate values: the maximum value of 48 kHz, the minimum value of 8 kHz, and the standard CD quality of 44.1 kHz. After comprehensive testing, the CD quality standard of 44.1 kHz was determined to be the most effective for our project.

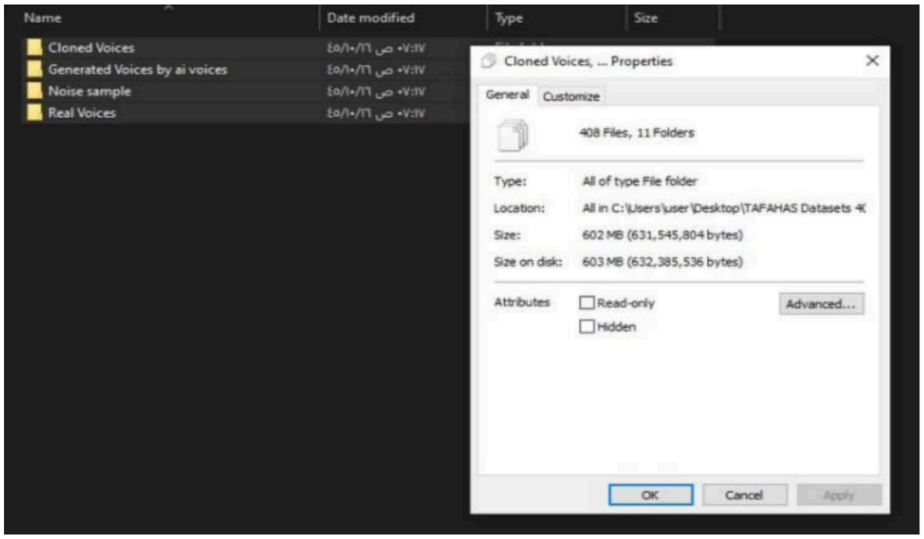
Bit Depth Optimization:
Bit depth determines the level of detail captured in each audio sample, directly affecting sound quality. Higher bit depths lead to more accurate and precise audio representation. Common bit depths include 8-bit, 16-bit (standard for CDs), and 24-bit. To optimize bit depth, we tested three values: the maximum value of 24-bit, the minimum value of 8-bit, and the standard CD quality of 16-bit. After thorough analysis, the CD quality standard of 16-bit was identified as the optimal choice for our requirements.

Channel Optimization:
The number of audio tracks or channels also affects sound quality and immer-

sion. Mono sound (one channel) provides a single audio track, while stereo sound (two channels) offers a richer sound experience with separate tracks for left and right. To optimize channels, we considered the immersive quality of stereo sound and its ability to provide a better sense of direction in the audio.

To address these challenges, our team conducted comprehensive experiments with different sets of sample rates, bit depths, and channels. The goal was to identify the optimal settings that strike a balance between audio quality and data storage and processing efficiency, ensuring the effectiveness of subsequent audio processing tasks such as speech recognition, music playback, or sound classification.

Datasets Size:
The data collection process was not without its challenges, with the most significant being the sheer volume of the dataset, which reached a weight of 602 megabytes. This posed some difficulty in handling the data and proved to be cumbersome on our devices during downloading and training due to the relatively limited specifications of our equipment. To alleviate computational burdens and streamline processing, a strategic decision was made to reduce the dataset size while ensuring preliminary results and verifying that the model operates acceptably for a dataset size of 74.7 megabytes, comprising 138 audio clips. Additionally, deliberate focus was placed on collecting audio samples with specific noise characteristics, strategically selected to support the classification task, including background noise and distortion hard-clipping noises. As shown in Figure 13.



Name	Date modified
Cloned Voices	٤٥/١٠/٢٦ ص ٠٧:١٧
Generated Voices by ai voices	٤٥/١٠/٢٦ ص ٠٧:١٧
Noise sample	٤٥/١٠/٢٦ ص ٠٧:١٧
Real Voices	٤٥/١٠/٢٦ ص ٠٧:١٧

408 Files, 11 Folders

Type:	All of type File folder
Location:	All in C:\Users\user\Desktop\TAFAHAS Datasets 4(
Size:	602 MB (631,545,804 bytes)
Size on disk:	603 MB (632,385,536 bytes)
Attributes	☐ Read-only ☐ Hidden

Figure 13: Size of the Dataset

Cloned Voice Software:
Navigating between various software platforms presented a formidable challenge, particularly when it came to selecting the one best suited for our needs. Specifically, In the domain of voice cloning for our dataset, our objective of ensuring diversity in emotions presented a significant hurdle. Many of the software solutions we explored, including Elvenlabs[23], ResembleAI[25], and MurfAI[24], prioritized voice cloning over the integration of emotional nuances. Incorporating emotions into the cloned voices often required manually adding emotional cues within sentences, such as using phrases like "oh my god, I am happy to announce." However, we encountered accuracy issues with this approach when using these three software types. Fortunately, our search led us to Playht Software[22], which effectively addressed all of these challenges.

3.9 Baseline Model and Algorithms

We will explore certain characteristics that distinguish natural voices from generated ones, such as features related to pitch, rhythm, or spectral content, Time-domain Features, and Frequency-domain Features. Then, we will investigate and extract features from the audio data capturing the desired differences that the model should learn.Currently, we will rely mainly on features extracted from Mel-Frequency Cepstral Coefficients (MFCCs), spectral features[32] [40]. The key is to extract relevant information that helps the model distinguish between the classes. The extracted features will be used as inputs for training the model. Since we are working on a sound classification task, we will employ some machine learning models along with deep learning algorithms.

3.10 Machine Learning Models

The K-nearest neighbor (KNN): is a supervised machine learning algorithm. It's a non-parametric method used for classification tasks[41]. It classifies audio to three different classes: real audio, cloned audio, and artificial intelligence-generated audio. When you want to classify data, the k-nearest neighbor algorithm looks at the k nearest neighbors of any data points in the training set and assigns the majority class among those neighbors to the new data point. If there are three neighbors of class cloned voice, and one neighbor of class real voice, the knn algorithm will classify the sample as cloned voice since it is the majority class as shown in Figure 14[42].

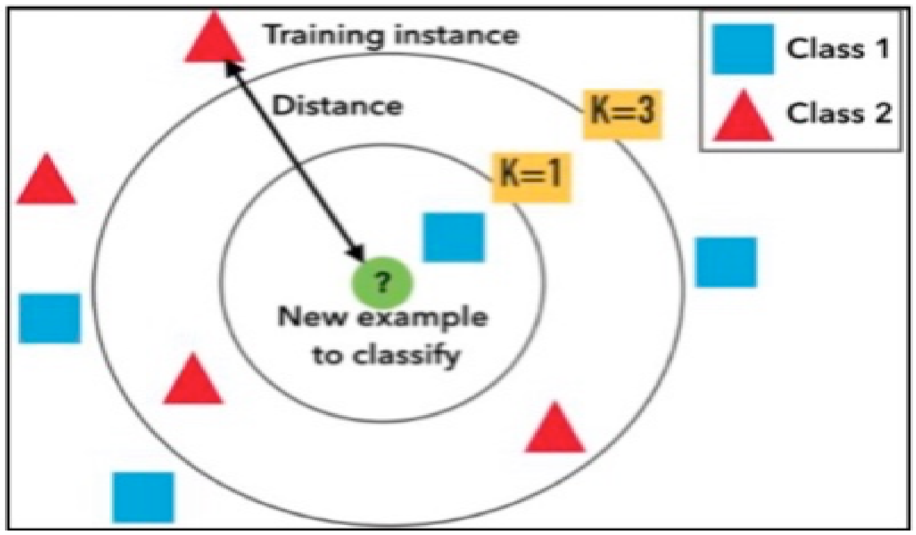


Figure 14: KNN Algorithm

Support Vector Machine (SVM): is one of the supervised learning algorithms. SVM is used for classification tasks, to understand pattern and decide to which class a pattern belongs to. SVM works well in many cases, even if the size of the training dataset is small. It constructs a hyperplane that separate data, to be in one of the two sides of the hyperplane. It attempts to create a hyperplane that ends up, by maximizing the width of the partition between the data points of non-Identical classes, resulting in a better classifier. The separation of data may be linear or non-linear[42][43].

Logistic Regression Model: is based on the probability of an outcome occurring, which can be defined as the ratio of the probability of an event occurring to the probability of it not occurring. Logistic regression shows the relationship between a dependent variable and an independent variable[44]. Logistic regression models are used to investigate the influence of the predictor variables (x) on categorical outcomes as shown in Figure 12.[44]. One of its main features is the possibility of including continuous explanatory variables, and managing multiple explanatory variables in the same time. This is important when investigating the impact of various explanatory variables on the response variable because it allows for the examination of covariance among variables and helps prevent the impact of confounding.In logistic regression, the model predicts the probability of an outcome based on individual features. Since probability is a ratio, then the model in fact predicts the logarithm of the probabilities, which is given by the equation below, where π. represents the probability of an event occurring, and ß are the regression coefficients related to the explanatory variables xi as shown in Figure 15[45].

$$\log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots \beta_m x_m$$

Figure 15: Logistic Regression

3.11 Deep Learning Models

Convolutional neural network (CNN): Efficiency in audio classification using CNN is a result of adaptation to recognizing hierarchical representations and local patterns. We extracted relevant features from the dataset that would be fed to the input of the CNN model, such as the extracted features of MFCC and Spectral centorid. The CNN model constructed will comprise convolutional layers, pooling layers, a fully connected layer, and an output layer[46].

Artificial Neural Networks (ANN): ANN models find applicability in accomplishing a large number of applications dealing with audio classification. Designing an artificial neural network for audio classification consists of an input, hidden, and output layer. We had extracted relevant features from the dataset, which would be fed as input into the ANN model; for example, the MFCC and Spectral centorid. In this network, the target classes or labels are given as the output layer, and the input layer represents the audio features. Basically, it is the hidden layers responsible for the number and size in the overall attempt to utilize input information in learning complex patterns and representations[47].

3.12 Design Basic Use Case diagram

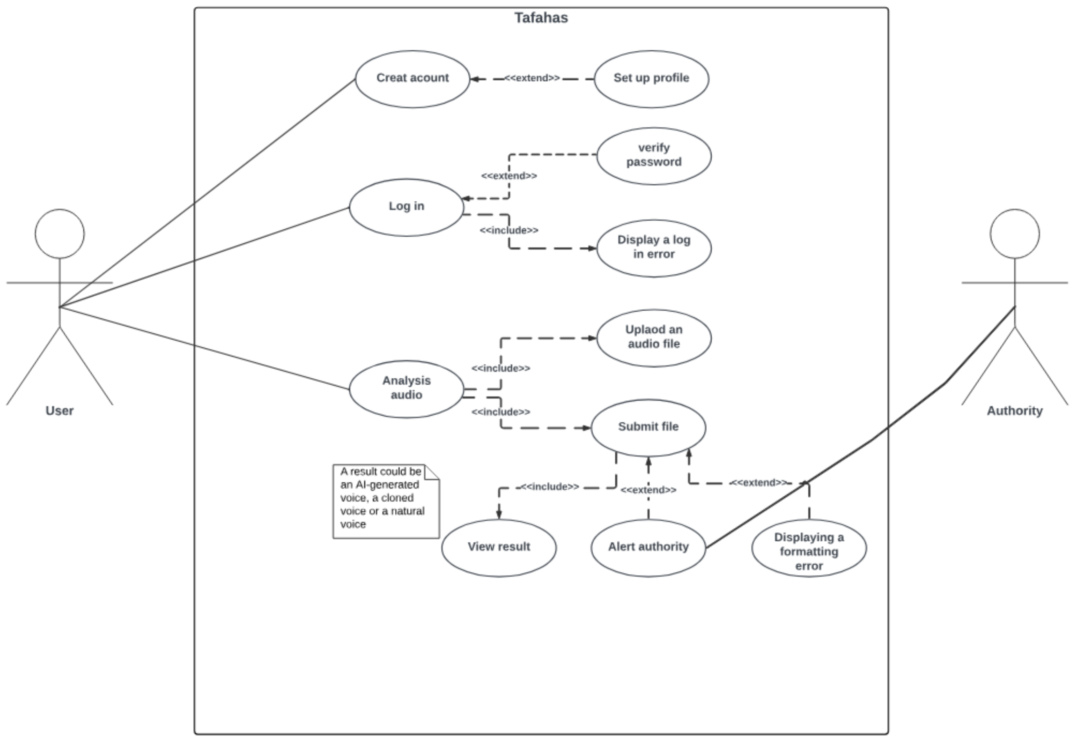


Figure 16: Tafahas Use-Case Diagram

Use-Case Diagram Content:

1-Use Case: Create Account
Brief description: In order to use the system, the system will allow the user to an create account
Pre-condition: The user must install the application
Post-condition: The user can log into the system
Actor: User
Flow of control:

- The user will download the application
- The system will ask the user to create account
- The user will fill the information needed Email,username, password
- The system will save user information

Constrains:

- The system must ensure the confidentiality of user's data
- User must follow password policy

Outcomes: The user will have a personal account in TAFAHAS

2-Use Case: Log in
Brief description: The user will be able to log in with his-her personal data username, password.
Pre-condition:

- The user must install the application
- The user must have an account

Post-condition: The user can log into the system
Actor: user
Flow of control:

- The user will download the application
- The user will fill the information needed username, password and it must be correct
- If the user has an account, then the user can log in successfully, otherwise, the system will ask the user to create an account

Constrains:

- User must provide correct username and password

- The system shall provide 3 attempts for the user to insert the password
- If all the 3 attempts failed, the system will delete the user information

Outcomes: The logging process to TAFAHAS is successful and complete

3-Use Case: Audio analysis

Brief description: The user will upload an audio to analyze it and classify it to real voice, cloned voice , synthesized voice.

Pre-condition:

- The user must have an account
- The user must log in to the system
- The system shall provide a user-friendly interface
- The user must have an audio file to upload
- Audio must be in a supported format

Post-condition: The audio analysis process will not begin unless the user clicks the submit button.

Actor: user

Flow of control:

- After the user logging to the system the user will be able to analyze an audio file
- The user will upload a file in a supported format, or an error message will appear
- After uploading a file, the user shall submit the audio file to be analyzed
- After the analysis process complete,the system will classify the audio to 3 categories real voice, cloned voice, synthesized voice
- The user will have an option to alert the authority if the voice was a cloned voice

Constrains:

- Audio must be in supported format
- Audio size must be limited
- The system will handle error that may occur during the uploading process

Outcomes: The user gets the classification result for the uploaded audio

4-Use Case: Upload an audio file
Brief description: To analyze the audio, the user must upload it to the system
Pre-condition:

- The user must have an audio file to upload
- Audio must be in a supported format

Post-condition: The system is ready to analyze the audio
Actor: user
Flow of control:

- After the user logging to the system the user will be able to analyze an audio file
- The user will upload a file in a supported format, or an error message will appear

Constrains:

- Audio must be in supported format
- Audio size must be limited

Outcomes: The user will successfully upload an audio to be analyzed in TAFAHAS

5-Use Case: Submit file
Brief description: After uploading an audio, the user must submit it to start the analysis process
Pre-condition: Audio must be in a supported format
Post-condition: The system will analyze the audio file and complete the process
Actor: user
Flow of control:

- After uploading the audio file correctly, the user will submit the audio to start the analysis process
- After the analysis process complete, a result will be shown

Constrains:

- Audio must be in supported format
- Audio size must be limited

Outcomes: The user will successfully upload and submit an audio in TAFAHAS

6-Use Case: View result

Brief description: After the completion of the analysis process, a result of the audio class will be shown to the user

Pre-condition: Audio must be submitted

Post-condition: The system will classify the audio class after the completion of the analysis process

Actor: user

Flow of control:

- After uploading the audio file correctly, the user will submit the audio to start the analysis process
- After the analysis process complete, a result will be shown
- If the result is a cloned voice, the user will have an option to report the authority

Constrains: None

Outcomes: The system shall provide the result of the audio class.

7-Use Case: Alert authority

Brief description: After viewing the result, the user can alert the authority if the audio is cloned.

Pre-condition:

- Audio must be submitted and analyzed
- Audio must be cloned

Post-condition: The user will have the option to alert the authority

Actor: User, authority

Flow of control:

- After the analysis process complete, a result will be shown
- If the result is a cloned voice, the user will have an option to alert the authority

Constrains: Audio must be cloned

Outcomes: The user will report the authority when the audio is a cloned voice

8-Use Case: Display formatting error
Brief description: The system will show a formatting error message if the audio file not in a supported format
Pre-condition: Audio must be in a supported format
Post-condition: The system will show an error message if the audio isn't in a supported format.
Actor: User
Flow of control:

- The user will upload an audio
- Error message will be shown if the file isn't supported
- The user must reupload the audio in supported format

Constrains: Audio must be in a supported format
Outcomes: The system will show an error message if the audio file format isn't supported.

9-Use Case: Setup profile
Brief description: The users will be able to set their profile information such as name , age,etc
Pre-condition: The user must have an account
Post-condition: The system will allow the user to set the profile
Actor: User
Flow of control:

- The user creates an account
- Option to set profile is shown to the user

Constrains: User must create an account
Outcomes: The user creates an account and sets their profile

10-Use Case: Verify password
Brief description: The system must ensure that the user has the correct password
Pre-condition: The user must have an account
Post-condition: The system will show an error message when the user tries to log in to an existing account with a wrong password **Actor:** User
Flow of control:

- The user tries to log in with username,password
- Error will be shown if the password is wrong
- The user will have more 3 attempts to enter the password correctly

Constrains: User must create an account
Outcomes: The system will show an error message if the password is wrong and ask the user to enter the password again

11-Use Case: Display a log in error message
Brief description: The system will display a log in error if the account is not found e.g. username is not found
Pre-condition: The user must download the application
Post-condition: The system will show a log in error the user information isn't correct
Actor: User
Flow of control:

- The user downloads the application
- The user tries to log in
- Log in information isn't correct

Constrains: User must download the application
Outcomes: The system show an error message if the user tries to log in with in correct account information

3.13 Design Prototype

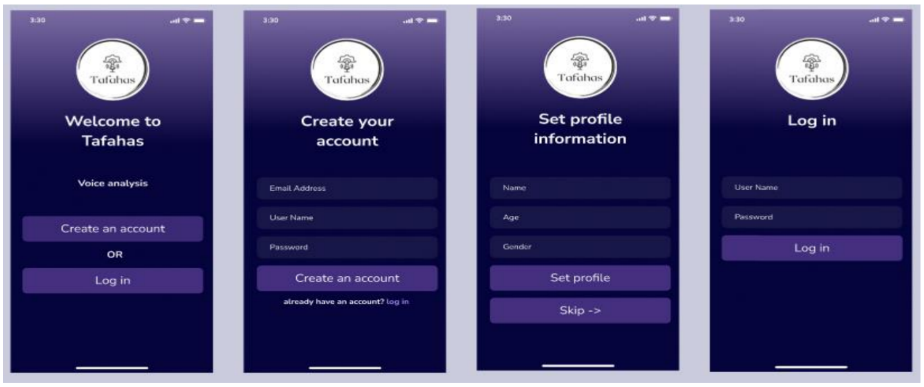


Figure 17: Create account / Log in /Set profile interfaces

As you can see in Figure 17, to access our application, the users must create an account, otherwise they can't use the app. Once the users have created their account, they can proceed to set up their profile information or skip this step. In the set-up process, a user might enter their actual name, age, and gender.If the user already has an account,they can simply login using their username and password.

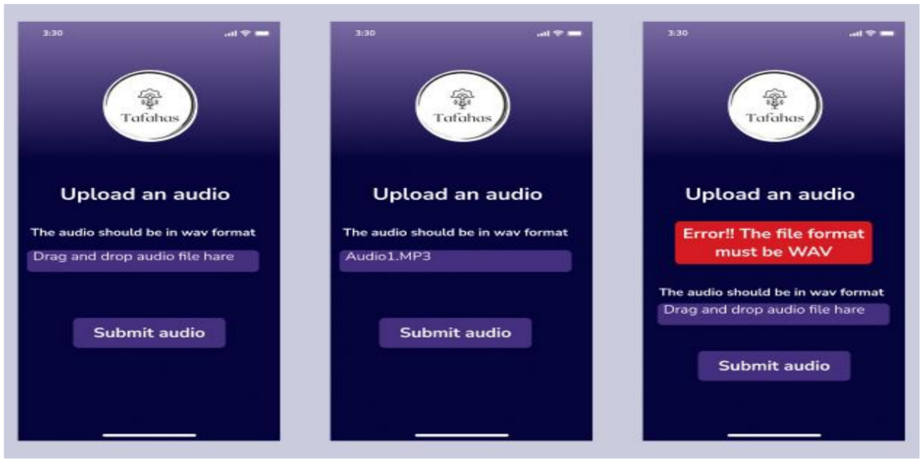

3:30
Tafahas
Upload an audio
The audio should be in wav format
Drag and drop audio file hare
Submit audio
Upload an audio
The audio should be in wav format
Audio1.MP3
Submit audio
Upload an audio
Error!! The file format must be WAV
The audio should be in wav format
Drag and drop audio file hare
Submit audio

Figure 18: Upload audio/ generating error if the format is invalid interfaces

When a user wants to check audio if it's real, cloned, or generated by the AI, the user needs to drag the audio from the device and drop it in the (drag and drop audio here) box. The audio must be in a WAV format, otherwise, the system won't accept it and it will display an error message as you can see in Figure 18. The audio analysis process won't begin unless the user clicks on the submit button.

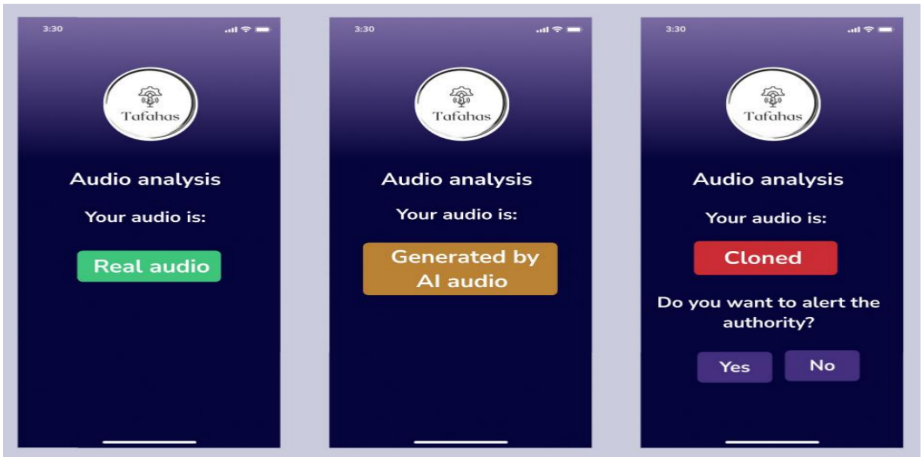


Figure 19: Results interfaces

After submitting the audio, the system will start analyzing it. When the system is done analyzing the audio, it will display the audio result as you can see in Figure 19 . A result could be real audio, generated by AI audio, or cloned audio. When the result is cloned audio, the system will ask the user if they want to alert the authority or not. If the user chooses “Yes”, the system will alert the authority. If the user clicks “No” the system won't alert the authority. It's the user's choice to alert the authority or not.

Chapter 4

Result and Experiment

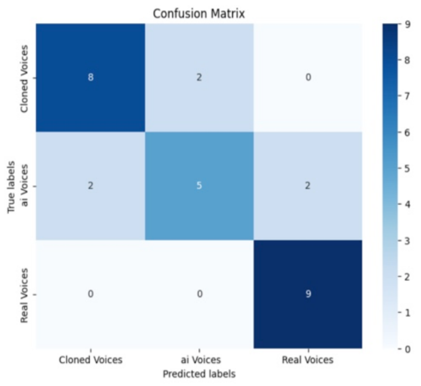
4.1 The Parameters and Results for Machine Learning Models

KNN Model:

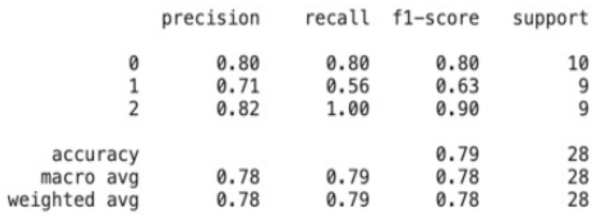
- The features extracted from speech are used as input.
- The number of neighbors (k) must be determined.
- The distance between the training data, and the input data is calculated.
- The k training samples closest to the input data are selected.[41] For example, [k = 4 means that the four training data points closest to the point that we want to classify will be selected].
- The point classification is assigned based on the majority class among the samples closest to the point.[42]
- The classification accuracy of the system is evaluated based on how well the predicted class matches the actual class.[41][42]

Model	Training accuracy	Validation accuracy	Testing accuracy	Training loss	Validation Loss	Accuracy
KNeighborsClassifier(n_neighbors=5, weights='uniform',algorithm='auto', metric='minkowski')	0.81	0.64	0.79	0.28	0.35	79%
KNeighborsClassifier(n_neighbors=5, weights='uniform',algorithm='auto', metric='cityblock'	0.80	0.71	0.79	0.32	0.28	79%
KNeighborsClassifier(n_neighbors=5, weights='uniform',algorithm='auto', metric='cosine')	0.80	0.71	0.86	0.26	0.28	86%
KNeighborsClassifier(n_neighbors=3, weights='uniform',algorithm='auto', metric='minkowski')	0.89	0.64	0.89	0.20	0.35	89%
KNeighborsClassifier(n_neighbors=3, weights='uniform',algorithm='auto', metric='cityblock')	0.86	0.64	0.79	0.26	0.35	79%
KNeighborsClassifier(n_neighbors=4, weights='uniform',algorithm='auto', metric='cosine')	0.82	0.71	0.79	0.27	0.28	82%
KNeighborsClassifier(n_neighbors=4, weights='uniform',algorithm='auto', metric='cityblock')	0.82	0.71	0.86	0.27	0.28	86%

Table 1: KNN Result



(a) KNN Confusion Matrix



```
              precision    recall  f1-score   support

           0       0.80      0.80      0.80        10
           1       0.71      0.56      0.63         9
           2       0.82      1.00      0.90         9

    accuracy                           0.79        28
   macro avg       0.78      0.79      0.78        28
weighted avg       0.78      0.79      0.78        28
```

(b) Classification Report

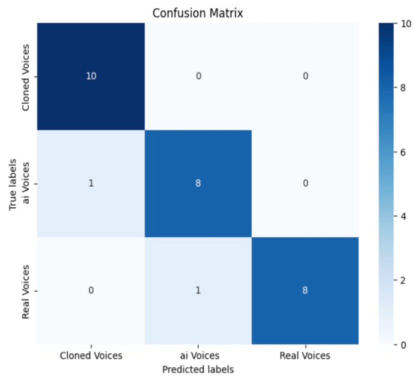
Figure 20: KNN

As shown in Table 1 and in Figure 20. The knn model tends to overfit the data as number of k neighbors decrease. We noticed that the metric parameter affects the performance in a good way. We decided to choose the k to be 5 and the metric to be cityblock which is a Manhattan distances metric. The weight is uniform so that all points have the same weight means that it all equally contribute to the classification of the point. Algorithm will be auto which means the knn algorithm will choose the best algorithm in our situation depending on the data.

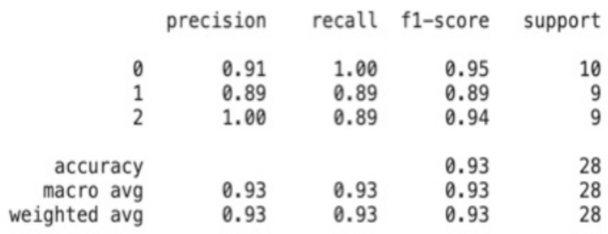
Support Vector Machine Model:

Model	Training accuracy	Validation accuracy	Testing accuracy	Training loss	Validation Loss	Accuracy
SVC(C=1,kernel='linear', probability=True)	0.95	0.93	0.93	0.083	0.071	93%
SVC(C=0.7,kernel='linear', probability=True)	0.92	0.93	0.93	0.114	0.071	93%
SVC(C=0.1,kernel='linear', probability=True)	0.85	0.79	0.89	0.208	0.214	89%
SVC(C=1,kernel='rbf', probability=True)	0.95	0.64	0.82	0.083	0.357	82%
SVC(C=0.7,kernel='rbf', probability=True)	0.94	0.64	0.82	0.093	0.357	79%
SVC(C=0.1,kernel='rbf', probability=True)	0.69	0.57	0.64	0.375	0.4285	64%
SVC(C=0.7,kernel='sigmoid' ,probability=True,gamma='auto')	0.81	0.79	0.86	0.28	0.21	86%

Table 2: Support Vector Machine Result



(a) SVM Confusion Matrix



```
              precision    recall  f1-score   support

           0       0.91      1.00      0.95        10
           1       0.89      0.89      0.89         9
           2       1.00      0.89      0.94         9

    accuracy                           0.93        28
   macro avg       0.93      0.93      0.93        28
weighted avg       0.93      0.93      0.93        28
```

(b) Classification Report

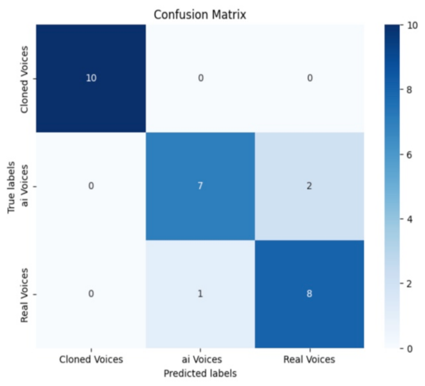
Figure 21: SVM

For support vector machine model, the model shows impressive results without overfitting as shown in the Table 2. Our chosen parameters result in 0.114 training loss and 0.071 validation loss. Since the training loss is much higher than the validation loss, we believe that it's not overfitting. We decided to choose C= 0.7 which is a Regularization parameter, and between the kernel sigmoid which indeed shows a good performance and the kernel linear we decided to choose the kernel linear because it's more efficient and mostly work well on high dimensional data like audio. The confusion matrix and classification report shown in Figure 21.

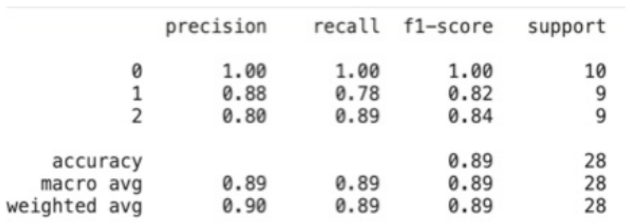
Logestic Regression Model:

Model	Training accuracy	Validation accuracy	Testing accuracy	Training loss	Validation Loss	Accuracy
LogisticRegression(penalty='l2', C=0.7, solver='lbfgs')	0.90	0.86	0.96	0.13	0.14	96%
LogisticRegression(penalty='l2', C=1, solver='lbfgs')	0.92	0.86	0.96	0.11	0.14	96%
LogisticRegression(penalty='l1', C=1, solver='liblinear')	0.88	0.86	0.93	0.21	0.14	93%
LogisticRegression(penalty='l1', C=0.7, solver='liblinear')	0.84	0.79	0.89	0.25	0.21	89%
LogisticRegression(penalty='l2', C=0.7, solver='liblinear')	0.86	0.79	1.00	0.19	0.21	100%
LogisticRegression(penalty='l2', C=1, solver='liblinear')	0.86	0.79	1.00	0.19	0.21	100%

Table 3: Logestic Regression Result



(a) Logestic Regression Confusion Matrix



```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       0.88      0.78      0.82         9
           2       0.80      0.89      0.84         9

    accuracy                           0.89        28
   macro avg       0.89      0.89      0.89        28
weighted avg       0.90      0.89      0.89        28
```

(b) Classification Report

Figure 22: Logistic Regression

Regarding the results shown above in Table 3, it seems like Logistic Regression model is facing the overfitting problem in most cases especially when the penalty = 'l2' and the solver is liblinear. l2 penalty have no mechanism for feature selection and not robust to outliers unlike l1 regularization that select features in penalty term and robust to outliers. The Table 3 shows the results when we tried different parameters. We decided to choose (penalty='l1', C=0.7, solver='liblinear') with 89% accuracy, since training loss is greater than validation loss and the results is more reasonable for us although we cannot depend on it because of our data size. The results of confusion matrix and classification reports are shown In Figure 22.

4.2 An overall Result of the Machine Learning Models

The data was tested on the decision tree model, but it showed poor perform ance on the test data. The model learned perfectly from the training data but failed to perform well on the test data. Results showed that the model's performance on the training data ranged from 1 to 0.96 which is very good, but the performance on the test data ranged between 0.72 to 0.54 Which is considered bad compared to the performance of the training data and indicates that the model is overfitting. We were unable to hyper tune the parameters to suit our task in a good way, so we can say that the decision tree model failed to perform well on the test data

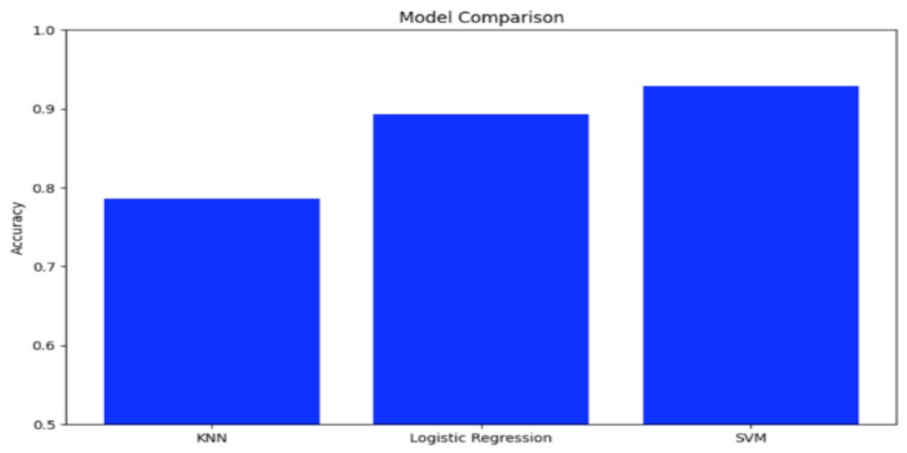


Figure 23: Models Comparison

Model	Accuracy
KNN	79%
Logistic Regression	89%
SVM	93%

Table 4: Model Accuracy

As shown in Table 4 and Figure 23, the machine learning models showed very high performance in classifying voices especially when we compare it to our deep learning models. We were surprised from the performance, but we believe based on the results we obtained, that the models were not overfitting the data. Since the training loss was greater than the validation loss in all models, this is one of the signs that the model is not overfitting. We also believe that models need more data to train so we can decide whether they are performing well or not. We believe that the size of the data was one of the reasons that affected performance significantly, and we will work to solve all problems in the future when we train the model on a large amount of data, as it can learn and generate well for unseen data

4.3 The Parameters and Result for Deep Learning Models

CNN and ANN Experiments
16 batch size was used for the training CNN and ANN over a dataset that comprised of 138 audio samples. 16 batch size works well for many reasons:

Batch Size:

- Memory efficiency: Batches smaller than this are inefficient in memory usage, making them suitable for devices with low memory, for example GPUs.
- Generalization: With a batch size of 16, diversity is already guaranteed in every training process, which will be useful for model adaptability to new samples. and prevent overfitting.
- Computational efficiency: 16 batch sizes ensure that parallel processing is achieved, so that maintain high convergence qualities while being computational Effectivity.
- Dataset size: The model can update weights frequently and compute efficiently.Analyse the dataset with a batch size of 16, which is slightly more than 11% of the total dataset size of 138.

Optimizer: High-performance optimizers, adaptive learning rates effective convergence, Adam, and RMSprop will be selected as the initial optimizer preference. They are a great base to develop neural networks to do so many audio-based classification tasks. We started with these two optimizers Because of various reasons:

- Stable Performance: The Adam and RMSprop optimizers have shown stable performance over a wide range of applications and architectures. They work very well for problems in the area of audio classification for both CNNs and ANNs.
- Adaptive learning rates: Both optimizers can handle the noise, modify the learning rates for the data and parameter gradient scales to each parameter.
- Convergence Speed: Compared to the typical optimizers, Adam and RMSprop oftentimes achieves the minimum quickly, and hence reducing training time.

The best way to choose the values for the hyperparameter in both optimizers is to experiment with different values and observe their output. In CNN and ANN models, setting the random seed is the factor that can make them reproducible. This means that every time we run the code, it will produce the same results, including the splitting of data.

CNN Model Experiments: The code of the CNN was taken from [48] but we modified it to meet our requirements. Our CNN model is not given as high accuracy as it is expected due to our small dataset size.

parameters	Training accuracy	Testing accuracy	Training loss	Testing loss
learning rate=0.00001,decay=1e-6, epochs=100, batch_size=16	0.49	0.54	1.04	1.05
learning rate=0.00001,decay=1e-6, epochs= 80, batch_size=16	0.44	0.26	1.08	1.10
learning rate=0.00001,decay=1e-6, epochs= 50, batch_size=16	0.37	0.23	1.09	1.10

Table 5: CNN Epochs Numbers

In Table 5 we started with 100 epochs and then we decreased to 80 and then 50, so we can determine the suitable number for the epochs. As observed from Table 5, the accuracy decreases when we decrease the epochs number, so based on these results, we won't keep investigating the epochs number and we will keep it =100 epochs

parameters	Training accuracy	Testing accuracy	Training loss	Testing loss
learning rate= 0.1, decay=1e-6 , epochs=100, batch_size=16	0.35	0.25	1.10	1.11
learning rate= 0.01, decay=1e-6 , epochs=100, batch_size=16	0.87	0.40	0.34	19.04
learning rate= 0.001, decay= 1e-6, epochs=100, batch_size=16	0.90	0.40	0.28	5.89
learning rate= 0.0001, decay= 1e-6, epochs=100, batch_size=16	0.84	0.40	0.39	2.27
learning rate=0.00001, decay=1e-6 epochs=100, batch_size=16	0.49	0.54	1.04	1.05

Table 6: CNN Adam Optimizer

In Table 6 we experimented with the Adam optimizer with different learning rates and decay values, and with fixed epoch numbers and batch size. As we observed from the results in Table 6, the best value for the learning rate that gave us the highest accuracy is 0.00001 with decay=1e-6. Moreover, we tried to change the deacy's value whenever we changed the learning rate but changing the decay value didn't affect the accuracy result.

parameters	Training accuracy	Testing accuracy	Training loss	Testing loss
learning rate= 0.1, decay=1e-6, epochs=100, batch_size=16	0.33	0.25	1.10	1.12
learning rate= 0.01, decay=1e-6, epochs=100, batch_size=16	0.85	0.40	0.44	2.28
learning rate= 0.001, decay= 1e-6, epochs=100, batch_size=16	0.85	0.57	0.37	7.78
learning rate= 0.0001, decay= 1e-6, epochs=100, batch_size=16	0.80	0.40	0.45	1.57
learning rate=0.00001, decay=1e-6 epochs=100, batch_size=16	0.46	0.54	1.04	1.04

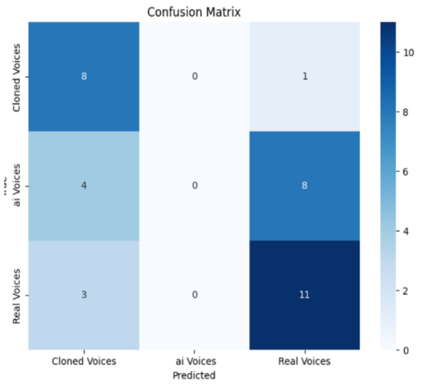
Table 7: CNN RMSprop Optimizer

In Table 7, as we can observe from the results, that with Learning rate=0.00001 the model performs better in the regard of overfitting. It has a balanced testing accuracy and comparable training and testing losses, suggesting a better generalization capability compared to Learning rate=0.001. So, the best value for the learning rate is 0.00001 with decay=1e-6. Moreover, we tried to change the deacy's value whenever we changed the learning rate, but it didn't affect the accuracy result.

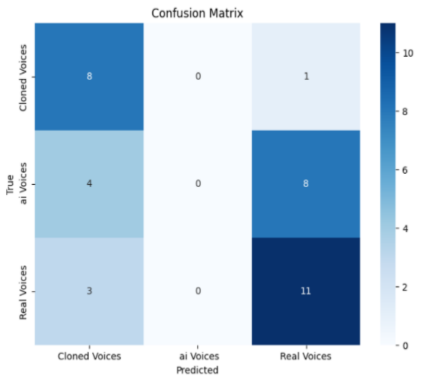
Optimizer name	RMSprop	Adam
Parameters	Learning rate= 0.00001, decay= 1e-6	Learning rate= 0.00001, decay= 1e-6
Final Training Accuracy	0.46	0.49
Final Testing Accuracy	0.54	0.54
Accuracy for class 0	0.88	0.88
Accuracy for class 1	0.0	0.0
Accuracy for class 2	0.78	0.78

Table 8: Comparing the two Optimizers for CNN

Cloned Voices = 0, Generated by AI Voices = 1, Real Voices = 2.



(a) Adam Optimizer



(b) RMS Optimizer

Figure 24: CNN Optimizers

In Table 8, we compared the two optimizers, both of them showed comparable performance in terms of overall accuracy and generalization. As it appears in the confusion matrix, in Figures 24, both optimizers classified and misclassified the classes equally. They classified 19 samples correctly out of 35 samples. However, we will choose Adam optimizer considering it is an adaptive optimization algorithm that combines the benefits of both AdaGrad and RMSprop.

ANN Model Experiments: The code was taken from [49] but we modified it to meet our requirements.

Parameters	Training accuracy	Testing accuracy	Training loss	Testing loss
Learning rate=le-4, epochs=100, batch_size=16	0.72	0.69	1.63	1.64
Learning rate=le-4, epochs=80, batch_size=16	0.66	0.60	1.95	1.92
Learning rate=le-4, epochs=50, batch_size=16	0.50	0.55	2.38	2.35

Table 9: ANN Epochs Numbers

In Table 9 we started with 100 epochs and then we decreased to 80 and then 50, so we can determine the suitable number for the epochs. As observed from Table 9, the accuracy decreases when we decrease the epochs number, so based on these results, we won't keep investigating the epochs number and we will keep it =100 epochs.

Parameters	Training accuracy	Testing accuracy	Training loss	Testing loss
Learning rate=le-4, epochs=100, batch_size=16	0.72	0.69	1.63	1.64
Learning rate=le-3, epochs=100, batch_size=16	0.82	0.69	0.95	1.23
Learning rate=le-2, epochs=100, batch_size=16	0.82	0.74	0.64	0.92
Learning rate=le-1, epochs=100, batch_size=16	0.28	0.33	14.92	13.44

Table 10: ANN Adam Optimizer

In Table 10, we experimented with the Adam optimizer with different learning rates and decay values, and with fixed epoch numbers and batch size. Based on these results, it is determined that the model performs better with a learning rate=1e-2 which achieved the highest accuracy value.

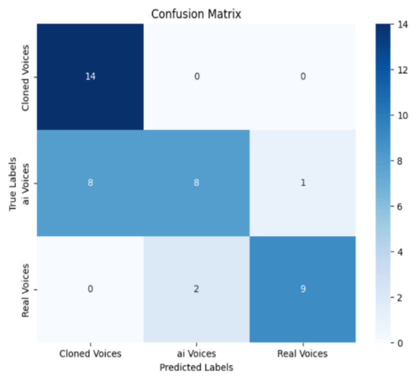
Parameters	Training accuracy	Testing accuracy	Training loss	Testing loss
Learning rate=le-4, epochs=100, batch_size=16	0.66	0.64	1.72	1.72
Learning rate=le-3, epochs=100, batch_size=16	0.88	0.76	0.53	0.70
Learning rate=le-2, epochs=100, batch_size=16	0.76	0.62	0.71	0.76

Table 11: RMSprop Optimizer Results

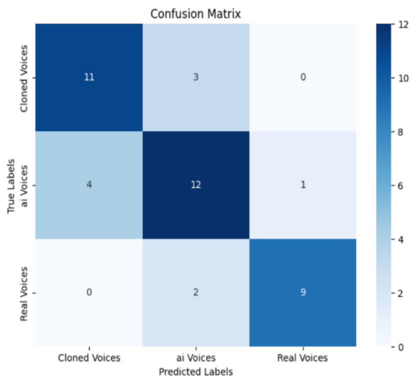
As we can see in Table 11, the results of RMSprop optimizer showed that the model's accuracy decreases while we train it with a larger learning rate, so we stopped at le =-1. The highest accuracy achieved with a learning rate=1e-3.

Optimizer name	RMSprop	Adam
Parameters	Learning rate= 1e-3	Learning rate= 1e-2
Final Training Accuracy	0.88	0.82
Final Testing Accuracy	0.76	0.74
Accuracy for class 0	0.79	1
Accuracy for class 1	0.71	0.47
Accuracy for class 2	0.82	0.82

Table 12: ANN Model Result



(a) Adam Optimizer



(b) RMS Optimizer

Figure 25: ANN Optimizers

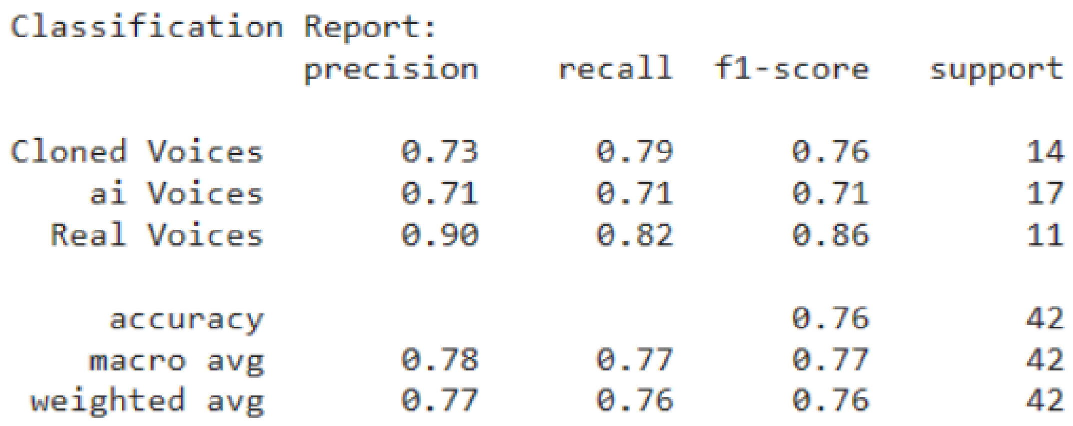
In Table 12, we compared the two optimizers, and based on the results, it appears that the RMSprop has a higher accuracy result than the Adam optimizer. Also, as observed from the confusion matrix in Figure 25, the RMSprop can differentiate between the classes better than Adam. RMSprop has classified 32 samples correctly out of 42 samples. While Adam has classified 31 samples correctly out of 42 samples. More details about the model's performance are in Figure 25.

Cloned Voices = 0, Generated by AI Voices = 1, Real Voices = 2.

4.4 An overall Result of the Deep Learning Models

Model	Optimizer	Training Accuracy	Testing Accuracy	Training Loss	Testing Loss
CNN	opt=keras.optimizers.Adam(learning_rate=0.00001, decay=1e-6, epochs=100, batch_size=16)	0.49	0.54	1.04	1.04
ANN	optimizer=RMSprop(learning_rate=1e3, epochs=100, batch_size=16)	0.88	0.76	0.53	0.70

Table 13: Model Results



```
Classification Report:
               precision    recall  f1-score   support

Cloned Voices       0.73      0.79      0.76        14
    ai Voices       0.71      0.71      0.71        17
  Real Voices       0.90      0.82      0.86        11

     accuracy                           0.76        42
    macro avg       0.78      0.77      0.77        42
 weighted avg       0.77      0.76      0.76        42
```

Figure 26: Classification Report

In Table 13, we compared the two models after deciding the best optimizer for each model with its suitable parameters. As observed from Table 13, the ANN model gives a higher accuracy than the CNN model and can distinguish the classes better than the CNN model. The low performance of the CNN model is due to our small dataset size as we mentioned earlier. The results of c classification reports is shown In Figure 26.

4.5 Discussion section

The models showed a good performance in classification based on the features extracted from the audio data, which are MFCC and spectral centroid[32] [40]. When we extracted the MFCC from a CSV file, the MFCC were in a lists of string data type, which needed to be converted to numbers so we converted it into a list of floats. When we converted the string values to float we divided the MFCC across the columns, therefore, each row is an audio sample. We had to do this to access them individually and feed them into the ML and DL models.

The data was collected by us, 408 audio were collected, but we faced a huge problem with the size of the data, which was 602 megabytes. It was difficult for us to work on the 408 audio in the process of feature extraction, training, and classification. We decided to reduce the data size to 138 audio, with a size of 74.7 megabytes, so we can deal with it. Our focus was on collecting audio with some noise characteristics, such as background noise and hard Clipping noises.

As mentioned previously, the models were trained on a small sample of data and not on all the data, yet they led to a good classification of most of the data. We noticed some problems in classifying the voice generated by artificial intelligence, and we believe that the poor performance of the classification in this class is due to two reasons. The system may be unable to classify it because it is very close to the human voice, given that the MFCC of the voice generated by artificial intelligence is very close to the MFCC of the real human voice. This may be one of the reasons for the bad classification, as it is unable to distinguish between the two voices.

We believe that the second reason may be the lack of data, as the models are unable to classify it well enough, because they were not trained on sufficiently large amounts of data. We think of the possibility of getting rid of the generated by AI class, as it will be combined with the cloned voice. They are all considered cloned, since it is an unreal voice. The number of data will also be increased to reach high accuracy and a good accuracy of classification.

As for the cloned Voices, they were collected manually by software. We faced another problems which is finding a software, that generate cloned voices while showing emotions so that it feels like a real voice. Most of the software focused on the cloned voices without showing emotions, but we finally found the software playht[22] that helped us in collecting the cloned data by providing cloned voices with emotions.

We also noticed that Machine Learning models performed much better than Deep learning models CNN and ANN, which indeed as a result of the dataset size. We believe that in order to benefit from the capability of deep learning models we need to increase the data size which is something we will work on it. due to the fact that deep learning models needs large dataset, but machine learning can work with small datasets better than deep learning models.

Further development in CNN model: we will increasing the dataset size. Moreover we will use callbacks to monitor the training progress, perform early stopping, save the best model, and adjust learning rates dynamically. Also as a result of the data size we split the data only into 2 sets but we will split it into three subsets in the future.

for the limitation of CNN: CNNs have many parameters, such as the number of layers, filter sizes, and learning rates, which need to be carefully tuned to obtain optimal performance. With our small dataset, it becomes challenging to perform robust hyperparameter tuning. In order for CNN architectures to learn discriminative features that are well-suited to new instances, lots of data sets are needed. Insufficient diversity in the dataset can make it more difficult for the model to identify representative characteristics, which could lead to less-than-ideal performance. Limited model complexity with a small dataset, the model's complexity may need to be limited to avoid overfitting.

References

[1] Zaynab Almutairi and Hebah Elgibreen. “A review of modern audio deepfake detection methods: challenges and future directions”. In: *Algorithms* 15.5 (2022), p. 155.

[2] McAfee Cybersecurity. *Beware the Artificial Impostor: A McAfee Cybersecurity Artificial Intelligence Report*. Retrieved February 24, 2024, from `https://shorturl.at/rBPVX`. 2023.

[3] Federal Trade Commission. *Preventing Harms from AI-Enabled Voice Cloning*. `https://www.ftc.gov/policy/advocacy-research/tech-at-ftc/2023/11/preventing-harms-ai-enabled-voice-cloning`. Accessed: 2024-02-24. 2023.

[4] Chengzhe Sun et al. “AI-Synthesized Voice Detection Using Neural Vocoder Artifacts”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 904–912.

[5] Zhenzi Weng et al. “Deep Learning Enabled Semantic Communications With Speech Recognition and Synthesis”. In: *IEEE Transactions on Wireless Communications* 22.9 (2023), pp. 6227–6240.

[6] Simone Scardapane et al. “On the use of deep recurrent neural networks for detecting audio spoofing attacks”. In: *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2017, pp. 3483–3490.

[7] Hafiz Malik. “Fighting AI with AI: fake speech detection using deep learning”. In: *2019 AES INTERNATIONAL CONFERENCE ON AUDIO FORENSICS (June 2019)*. 2019.

[8] Naroa Amezaga and Jeremy Hajek. “Availability of Voice Deepfake Technology and its Impact for Good and Evil”. In: *Proceedings of the 23rd Annual Conference on Information Technology Education*. 2022, pp. 23–28.

[9] Shi Qiu. “Construction of English Speech Recognition Model by Fusing CNN and Random Deep Factorization TDNN”. In: *ACM Transactions on Asian and Low-Resource Language Information Processing* (2023).

[10] Nishant Subramani and Delip Rao. “Learning efficient representations for fake speech detection”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 5859–5866.

[11] Logan Blue et al. “Who Are You (I Really Wanna Know)? Detecting Audio {DeepFakes} Through Vocal Tract Reconstruction”. In: *31st USENIX Security Symposium (USENIX Security 22)*. 2022, pp. 2691–2708.

[12] Ameer Hamza et al. “Deepfake audio detection via MFCC features using machine learning”. In: *IEEE Access* 10 (2022), pp. 134018–134028.

[13] Mvelo Mcuba et al. “The Effect of Deep Learning Methods on Deepfake Audio Detection for Digital Investigation”. In: *Procedia Computer Science* 219 (2023), pp. 211–219.

[14] Robin San Roman et al. “Proactive Detection of Voice Cloning with Localized Watermarking”. In: *arXiv e-prints* (2024), arXiv–2401.

[15] Yuxuan Wang et al. “Tacotron: A fully end-to-end text-to-speech synthesis model”. In: *arXiv preprint arXiv:1703.10135* 164 (2017).

[16] Aaron van den Oord et al. “Wavenet: A generative model for raw audio”. In: *arXiv preprint arXiv:1609.03499* (2016).

[17] Joel Frank and Lea Schönherr. “Wavefake: A data set to facilitate audio deepfake detection”. In: *arXiv preprint arXiv:2111.02813* (2021).

[18] Massimiliano Todisco et al. “ASVspoof 2019: Future horizons in spoofed and fake audio detection”. In: *arXiv preprint arXiv:1904.05441* (2019).

[19] Enkhtogtokh Togootogtokh and Christian Klasen. “DeepEMO: deep learning for speech emotion recognition”. In: *arXiv preprint arXiv:2109.04081* (2021).

[20] *LibriVox — free public domain audiobooks.* `https://librivox.org/`. Accessed on May 2, 2024.

[21] *Archive.org.* `https://archive.org`. Accessed on May 2, 2024.

[22] *PlayHT*. PlayHT Website. 2024. URL: `https://play.ht/`.

[23] *AI Voice Generator Text to Speech — ElevenLabs.* `https://elevenlabs.io/`. Accessed on May 2, 2024.

[24] *AI Voice Generator: Versatile Text to Speech Software — Murf AI.* `https://murf.ai/`. Accessed on May 2, 2024.

[25] *AI Voice Generator with Text to Speech and Speech to Speech.* `https://www.resemble.ai/`. Accessed on May 2, 2024.

[26] *YouTube.* `https://www.youtube.com/?app=desktop&hl=ar`. Accessed on May 2, 2024.

[27] *Freesound.* `https://freesound.org/`. Accessed on [Access Date].

[28] *Voices.* `https://www.voices.com/`. Accessed on [Access Date].

[29] *Wavepad.* `https://www.nch.com.au/wavepad/index.html`. Accessed on [Access Date].

[30] *Audacity ® — Free Audio editor, recorder, music making and more!* `https://www.audacityteam.org/`. Accessed on May 2, 2024.

[31] Enkhtogtokh Togootogtokh and Christian Klasen. “AntiDeepFake: AI for Deep Fake Speech Recognition”. In: *arXiv preprint arXiv:2402.10218* (2024).

[32] R H Aljuhani, A Alshutayri, and S Alahdal. “Arabic speech emotion recognition from Saudi dialect corpus”. In: *IEEE Access* 9 (2021), pp. 127081–127085.

[33] K Sivarama Krishnan and K Sivarama Krishnan. “MFAAN: Unveiling Audio Deepfakes with a Multi-Feature Authenticity Network”. In: *arXiv e-prints* (2023). eprint: `arXiv-2311`.

[34] D Prabakaran and S Sriuppili. "Speech processing: MFCC based feature extraction techniques-an investigation". In: *Journal of Physics: Conference Series*. Vol. 1717. 1. IOP Publishing, 2021, p. 012009.

[35] Mujtaba Raza. "Voice Classification Using MFCC Features and Deep Neural Networks: A Step-by-Step Guide". In: *Medium* (2023). Accessed on May 2, 2024.

[36] *Audio Emotion — Part 2 - Feature Extract — Kaggle*. `https://www.kaggle.com/code/ejlok1/audio-emotion-part-2-feature-extract?scriptVersionId=20847848`. Accessed on May 2, 2024.

[37] Noraziahtulhidayu Kamarudin et al. "Feature extraction using spectral centroid and mel frequency cepstral coefficient for Quranic accent automatic identification". In: *2014 IEEE Student Conference on Research and Development*. 2014, pp. 1–6.

[38] Jia Min Karen Kua et al. "Investigation of spectral centroid magnitude and frequency for speaker recognition." In: *Odyssey*. 2010, p. 7.

[39] T Aditya Sai Srinivas et al. "From Waves to Insights: A Visual Exploration of Sound". In: *Journal of Advances in Computational Intelligence Theory* 6.1 (2023), pp. 1–7.

[40] Shreya Kumar and Swarnalaxmi Thiruvenkadam. "An Analysis of the Impact of Spectral Contrast Feature in Speech Emotion Recognition." In: *Int. J. Recent Contributions Eng. Sci. IT* 9.2 (2021), pp. 87–95.

[41] A Bombatkar et al. "Emotion recognition using Speech Processing Using k-nearest neighbor algorithm". In: *International Journal of Engineering Research and Applications* 4 (2014), pp. 68–71.

[42] S Prabavathy, V Rathikarani, and P Dhanalakshmi. "Classification of Musical Instruments using SVM and KNN". In: *International Journal of Innovative Technology and Exploring Engineering* 9.7 (2020), pp. 1186–1190.

[43] B Vimal et al. "Mfcc based audio classification using machine learning". In: *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. IEEE. July 2021, pp. 1–4.

[44] T. G. Nick and K. M. Campbell. "Logistic regression". In: *Topics in Biostatistics* (2007), pp. 273–301.

[45] S. Sperandei. "Understanding logistic regression analysis". In: *Biochemia Medica* 24.1 (2014), pp. 12–18.

[46] Khalid Zaman et al. "A Survey of Audio Classification Using Deep Learning". In: *IEEE Access* 11 (2023), pp. 106620–106649. DOI: `10.1109/ACCESS.2023.3318015`.

[47] Vikramjit Mitra and Chia-Jiu Wang. "Content based audio classification: a neural network approach". In: *Soft Computing* 12 (2008), pp. 639–646.

[48] ejlok1. *Audio Emotion Part 3: Baseline Model*. https://www.kaggle.com/code/ejlok1/audio-emotion-part-3-baseline-model. Year Accessed.

[49] dikshabhati2002. *Let's Classify Audio (ML DL)*. https://www.kaggle.com/code/dikshabhati2002/let-s-classify-audio-ml-dl#Data. Year Accessed.