**Ain Shams University**

**Faculty of Computer & Information Sciences**

**Scientific Computing Department**

**Speech2Face Generator**

**By**

| Ahmed Magdy | Scientific Computing |
| --- | --- |
| Sara Mohamed | Scientific Computing |
| Ahmed Mohamed | Scientific Computing |
| Nourhan Mahmoud | Scientific Computing |
| Abdelrahman Yasser | Scientific Computing |
| Abdelrahman Mohamed | Scientific Computing |

**Under Supervision of**

Associate Prof. Manal Tantawy,  
Scientific Computing Department,  
Faculty of Computer and Information Sciences,  
Ain Shams University.

T.A. Mirna Al-Shetairy,  
Compter Science Department,  
Faculty of Computer and Information Sciences,  
Ain Shams University.

T.A. Aya Nasser,  
Scientific Computing Department,  
Faculty of Computer and Information Sciences,  
Ain Shams University.

# **Acknowledgement**

We would like to use this opportunity to express our gratitude to everyone who supported us throughout this project. We are thankful for their indispensable guidance, invaluably constructive criticism, friendly advice, and the most obliged provision of their genuine and illuminating views.

We would like to offer our special thanks to our supervisor Dr. Manal Tantawy. Her extensive knowledge, guidance, and mentorship have been instrumental in shaping the trajectory of our group project. Her insightful feedback, critical analysis, and profound understanding of the subject matter have greatly enriched our research and its outcomes. Dr. Manal's unwavering commitment to academic excellence and her ability to challenge and inspire have propelled us to push the boundaries of our capabilities and achieve the desired outcomes.

We would also like to extend our thanks to T.A. Mirna Al-Shetairy and T.A Aya Nasser have been an invaluable asset throughout this project, offering consistent support, practical insights, and guidance. Their dedication to our success and their willingness to go above and beyond in providing assistance has greatly contributed to the development and refinement of our project. Their expertise, approachability, and patience have created an environment conducive to learning and growth, where we felt comfortable seeking advice and clarifications.

Finally, We would like to express our gratitude to the entire faculty and staff who have indirectly contributed to the success of our graduation project. Their collective dedication to providing a nurturing and intellectually stimulating environment has been instrumental in our overall academic journey.

# Chapter 1: Introduction

In this innovative project, we delve into the realm of artificial intelligence to bridge the gap between spoken words and visual representation. Using advanced deep learning techniques, we have developed a cutting-edge model that can transform human speech into a lifelike image of the speaker's face.

Human communication is a complex interplay of verbal and non-verbal cues. While words convey meaning, facial expressions and gestures add depth and emotion to our interactions. Our project aims to unlock the potential of speech by creating a visual counterpart, allowing us to better understand the holistic nature of human expression.

The implications of speech-to-face generation are vast and diverse. From enhancing virtual reality experiences and animated movies to aiding in various fields like law enforcement, this technology holds immense potential. Imagine a world where a simple voice message could be transformed into a vivid visual representation, creating a more immersive and personalized communication experience.

In this project, we have made significant strides toward realizing this vision. Through extensive training on large datasets of speech and corresponding facial images, our model has learned the intricate correlations between vocal nuances and facial expressions. The results are remarkable, with generated images closely resembling the speaker's unique features, capturing the essence of their identity.

## 1.1 Motivation

The project draws inspiration from several compelling motivations, including:

* Enhancing Communication: Communication transcends mere verbal exchanges; non-verbal cues and facial expressions play an indispensable role in conveying meaning. By harnessing the power of generating realistic faces from speech, we seek to enrich communication experiences, particularly in contexts where visual cues are restricted, such as audio-only calls or voice messaging platforms.
* Transforming Entertainment Industries: Within animation, gaming, and film industries, voiceover performances constitute a cornerstone of creative expression. The ability to generate visually immersive representations of characters solely based on the performances of voice actors has the potential to revolutionize entertainment, offering seamless integration of voice and visuals, and elevating the audience's engagement and immersion.
* Assisting Forensic Investigations: In the realm of forensic investigations, where audio evidence often plays a pivotal role, the "Speech to Face Generator" can provide invaluable assistance. By generating facial avatars from audio recordings, this technology has the potential to aid law enforcement agencies in identifying suspects, reconstructing the appearance of unknown individuals, and potentially uncovering crucial leads in unsolved cases.
* Advancing AI Research: Positioned at the confluence of machine learning, audio analysis, and computer vision, the "Speech to Face Generator" project represents an intellectually stimulating endeavor. By venturing into uncharted territories of audio-visual synthesis, we strive to contribute to the broader advancement of AI research, exploring novel techniques and methodologies to bridge the perceptual gap between non-visual and visual content generation.
* Pioneering Technological Advancements: Beyond the immediate applications in communication and entertainment, the development of a reliable and accurate "Speech to Face Generator" holds transformative potential for various domains. It has the capacity to inspire breakthroughs in human-computer interaction, virtual reality, augmented reality, and numerous other fields, offering unprecedented opportunities for technological advancements and novel use cases

## 1.2 Problem Definition

For a long time, there have been several threatening crimes, such as extortion with images or death threats, among others. But what all of this has in common is that you may easily obtain the blackmailer's voice or at least a 5-second audio recording. Knowing only the voice of ther criminal will not help anyone to catch him, and most of those crimes been closed to unkown criminal.

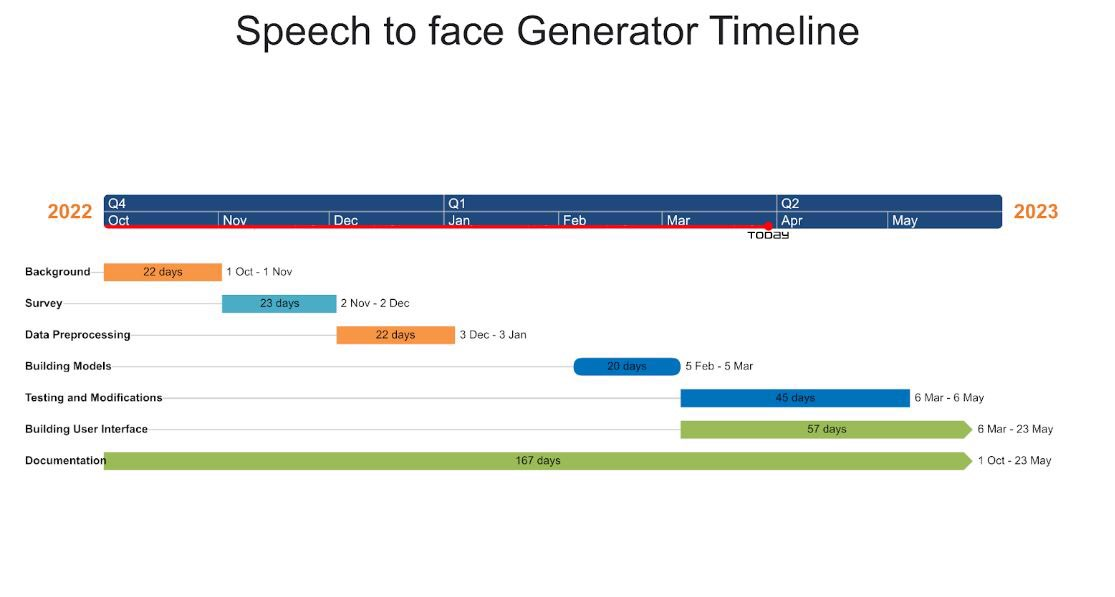
Police men need something more thann just a voice, they need at least image of that criminal. So, our system “speech2face generator” will take this audio and reconstruct a person's facial image from a brief audio recording of that person speaking to know the speaker and prevent the crime.

## 1.3 Objective

The objective is to meticulously develop and deploy an advanced system capable of generating highly realistic visual representations of individuals based exclusively on their recorded voices. By leveraging state-of-the-art deep learning techniques, sophisticated audio analysis algorithms, and cutting-edge models, our aim is to bridge the perceptual divide between audio and visual information, thereby pushing the boundaries of audio-visual synthesis. The specific objectives of this project encompass:

* Designing and implementing an intricate deep learning architecture proficient in extracting pertinent features from audio recordings and translating them into detailed facial attributes.
* Training the model using an extensive dataset of audio-visual pairs, ensuring the acquisition of accurate and visually plausible facial generation capabilities.
* Pioneering innovative methodologies to address inherent challenges including limited training data, variations in voice quality, and the diverse range of facial appearances encountered in real-world scenarios.
* Conducting comprehensive evaluations of the system's performance, employing rigorous quantitative metrics and meticulous user studies to quantitatively and qualitatively assess the fidelity and likeness of the generated faces.
* Optimizing the system for real-time or near real-time face generation, thereby fostering practical applications in communication and entertainment industries, where timely and seamless integration of audio and visual components is of utmost importance.
* Demonstrating a steadfast commitment to ethical considerations and privacy protection by rigorously adhering to responsible data usage practices, ensuring robust safeguards against potential misuse or unauthorized access to generated visual content.

## 1.4 Time Plan



**Figure 1.1: Time plan**

* 22 days for background establishing.
* 23 days for studying the survey, literature published and related work.
* 22 days for preparing the project data and preprocessing.
* 20 days for building the models and project architecture.
* 45 days for the testing and model modification in parallel with trying another model architecture.
* 57 days for integrating the model and building user interface.
* Working on the documentation in parallel with the whole work.

## **1.5 Document Organization**

**Chapter 2:** Background

This chapter includes background about the project, basic concepts, and the related work according to our research.

**Chapter 3:** Analysis & Design

This chapter includes an overview of the whole system along with the intended user, system architecture, analysis, and design.

**Chapter 4:** VAE Model

This chapter includes an overview of the dataset used in the model, describes the system functions with details about the implementation of the project’s modules, and includes some of the experiments that we did through our work to improve the results.

**Chapter 5:** GAN’S Model

This chapter includes an overview of the dataset used in the model, describes the system functions with details about the implementation of the project’s modules, and includes some of the experiments that we did through our work to improve the results

**Chapter 6:** User Manual

This chapter talks about the needed packages and libraries that must be installed before using the application, and also shows the user how to use it.

**Chapter 7:** Conclusion & Future Work

This chapter includes the conclusion and results of our work and the future work that may be done based on this project.

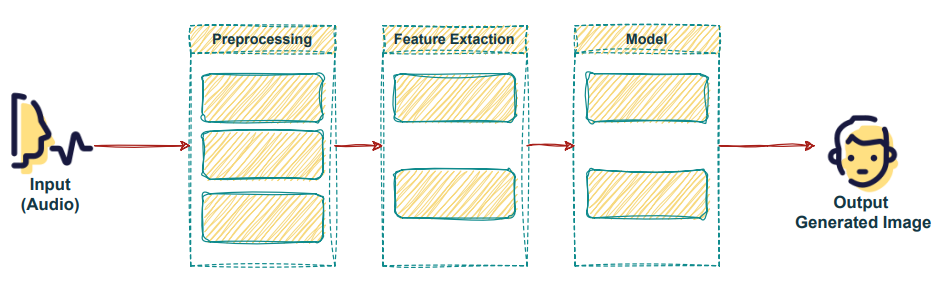
**References:** This chapter includes the papers, books, and links that were used in the survey and in implementation.

# Chapter 3: Analysis & Design

## 3.1 System Overview

We introduce the entire architecture of our system (*see figure 3.1*). We have two approaches, the first one is Generative Adversarial Networks (GANs) and the other is Variational Autoencoder (VAE).

### 3.1.1 System Architecture



**Figure 3.1 System Architecture**

First, The input is audio, it gets passed to a preprocessing step to normalize volume, remove noise, and Filter out silence. In the feature extraction step, we extract the log Mel spectrogram. Then feed the model with the output of the previous step. In the end, the output is a generated image for the person who provided the audio input.

### 3.1.2 System Users

1. Intended Users:

The Reconstructing Face from Voice system is built for end-users who wish to generate a face image from the audio input, such as for police officers, use in entertainment, artistic endeavors, or other creative projects.

1. User Characteristics:

The end-user of the system should have the following minimum requirements:

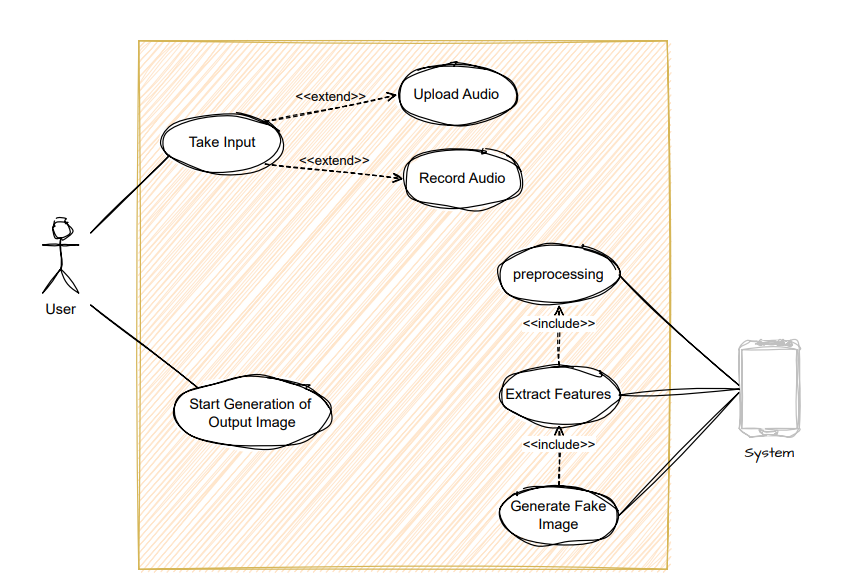
1. Basic computer skills: The end-user should have basic knowledge of how to operate a computer, such as navigating through the operating system, opening and closing applications, and using input devices like a mouse and keyboard.
2. Knowledge of audio recording: The end-user should have some experience with audio recording, such as understanding how to use a microphone or other recording equipment.
3. Understanding of limitations and potential biases.
4. Minimum application requirements: The end-user should have a device that meets the minimum requirements for installing the application. This could include a specific operating system, processor speed, memory, and storage requirements.
5. No specific hardware requirements: The system does not require any specific hardware to be used, meaning that the end-user can use any device that meets the minimum application requirements.

## 3.2 Description of method and procedure used

### 3.2.1 Use Case Diagram

The use case diagram for the Reconstructing Face from Voice system includes a User Actor that is responsible for allowing the user to input audio into the system through two use cases: Record Audio and Upload Audio. These use cases provide the user with two methods for inputting audio into the system, depending on their preferences and available resources.

Once the user has provided the necessary audio input, the system takes the audio and applies a series of processing steps like Normalize volume, remove noise, and Filter out silence. After pre-processing the audio, the system performs feature extraction and generates a fake image as shown in figure 3.2.



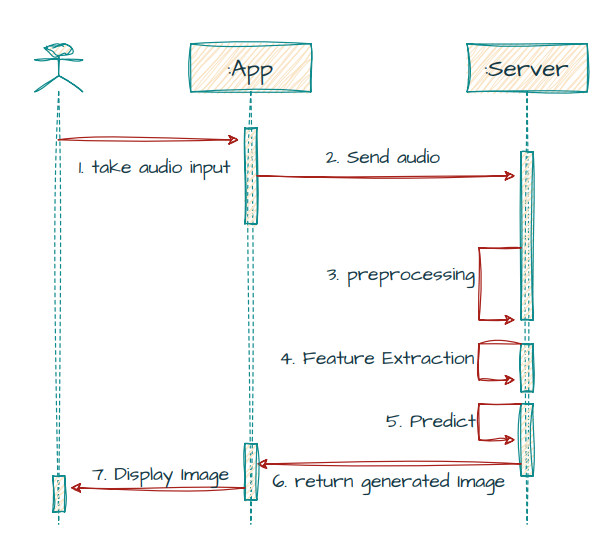
**Figure 3.2 System’s Use Case Diagram**

### 3.2.2 Flow Of Events

* The user initiates the process by either recording audio or uploading it to the system.
* The system performs preprocessing on the audio, including normalizing the volume, removing noise, and filtering out silence.
* Once the audio has been preprocessed, the system extracts features from it.
* The system generates a fake image based on the extracted features.
* Finally, the system displays the reconstructed face to the user.

### 3.2.3 Sequence Diagram

In the sequence diagram, as shown in figure 3.3, the User actor initiates the process by either recording audio or uploading it to the system. The system then performs preprocessing on the audio, including normalizing the volume, removing noise, and filtering out silence. Once the audio has been preprocessed, the system extracts features from it and generates a fake image based on those features. Finally, the system displays the reconstructed face to the user.



**Figure 3.3 System’s Sequence Diagram**

# Chapter 4: VAE Model

## 4.1 Environment Setup & Tools

Python programming language was used due to its functionalities which were compatible with the fields needed in the project i.e. Deep Learning.

### 4.1.1 Environment

* Google Colab : Google Colab was used due to the GPU Engine Provided by it on the website, it was more helpful than the localhost because the model needed about 10 GB GPU ram and the localhost had only 6 GB.

### 4.1.2 Packages and Libraries

1. **PyTorch** : PyTorch is an optimized tensor library for deep learning using GPUs and CPUs.
2. **Keras** : Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow
3. **NumPy** : NumPy is the fundamental package for scientific computing in Python.
4. **Pydub** : Manipulate audio with a simple and easy high level interface.
5. **Os** : This module provides a portable way of using operating system-dependent functionality.
6. **OpenCV** : An optimized Computer Vision library for reading, writing and processing images.
7. **Facemorpher** : A library automatically detects frontal faces and skip images if none is detected.
8. **Librosa** : A python package for music and audio analysis, It provides the building blocks necessary to create music information retrieval systems.
9. **Glob** : A module finds all the pathnames.

## 4.2 Dataset

[AVSpeech](https://looking-to-listen.github.io/avspeech/) dataset was used which is a large-scale audio-visual dataset comprising speech video clips with no interfering background noises. The segments are 3-10 seconds long, and in each clip the audible sound in the soundtrack belongs to a single speaking person, visible in the video. In total, the dataset contains roughly 4700 hours of video segments, from a total of 290k YouTube videos, spanning a wide variety of people, languages and face poses.

Unfortunately only 2611 clips of the data were used due to the difficulty of uploading and extraction of the data on google drive servers because of the bad internet services in egypt, and after the filtration of the clips using the “facemorpher”.



**Figure 4.2.1 AVSpeech Sample**

## 4.3 Data Preprocessing

### **4.3.1 Audio Data Preprocessing**

1. The audio segment was repeated until it reached 6 secs if it was less than that.
2. Then the audio was converted to a wav file and, a STFT with a hann window and sampling rate 16000 was applied to it.

### 4.3.2 Face Data Preprocessing

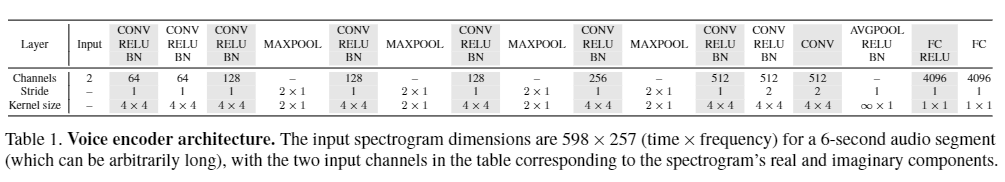
1. The cropped RGB face images of size 224x224x3 are obtained by similarity transformation.
2. Each pixel in the RGB images is normalized by subtracting 127.5 and then dividing by 127.5.

## 4.4 Model

### **4.4.1 VAE Model**

**Figure 4.4.1 VAE Architecture**

### **4.4.2 Voice Encoder**



**Figure 4.4.2 Voice Encoder Architecture**

* voice encoder, which takes a complex spectrogram of speech as input, and predicts a low-dimensional face feature that would correspond to the associated face
* The output is a feature vector (1x4096)

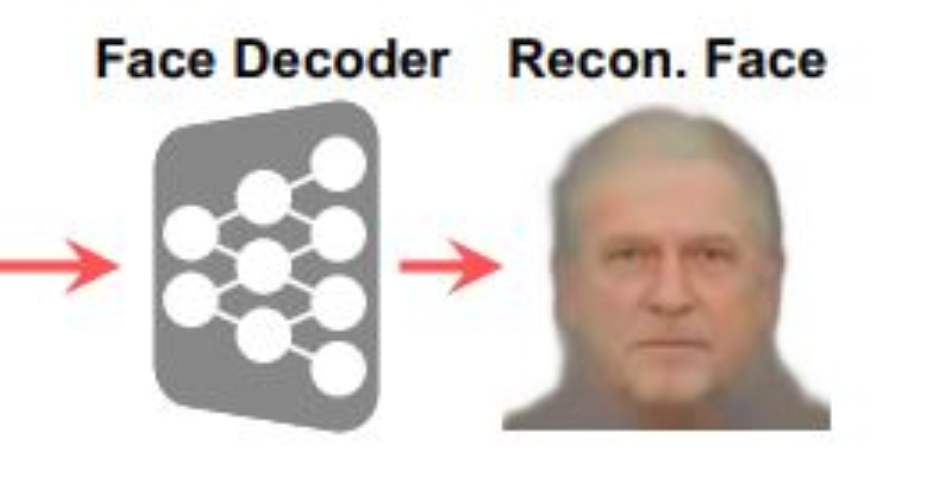
### 4.4.3 Face Recognition (VGG-Network)

**Figure 4.4.3 VGG-Network Architecture**

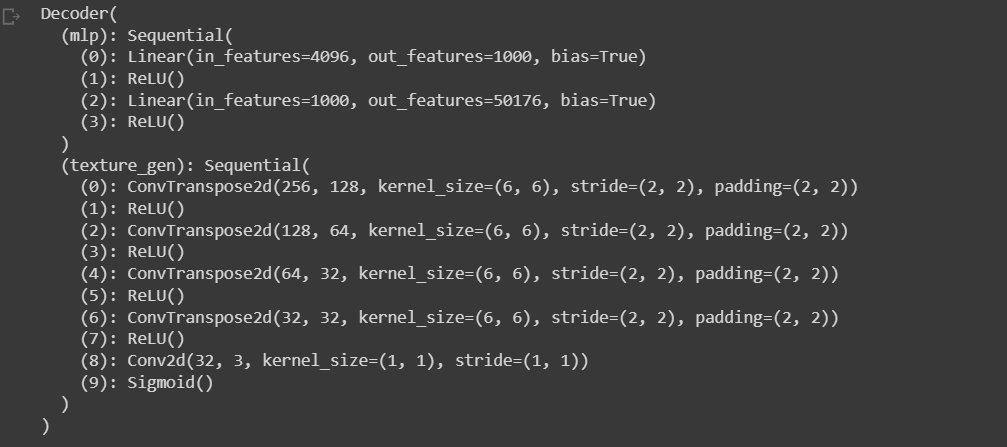
VGG-Face model, a pre- trained face recognition model trained on a large-scale face dataset, and extract a 4096-D face feature from the penultimate layer ( fc7 ) of the network

### **4.4.4 Face Decoder**

which takes as input the face feature and produces an image of the face in a canonical form (frontal-facing and with neutral expression), We trained the model by 120 iteration using 149,000 face images to reconstruct the images from the feature vector (1x4096)D only.



**Figure 4.4.4 Face Decoder reconstruct face**



**Figure 4.4.4.1 Face Decoder Network**

## 4.5 Experiments & Results

### 4.5.1 Experiments

We built the same architecture as in a paper [3] but try a different optimizer and hyperparameters. Paper set the ‘Optimizer’: ‘Adam’ ‘learning rate’: 0.0002, ‘beta1’: 0.5, ‘beta2’: 0.999.

We try different hyperparameters, but finally we set the ‘Optimizer’: ‘RMSProp’, alpha: 0.9, eps: 1e-08, weight\_decay=0, momentum=0, centered=False.

We also add some extra layers such as we add batch normalization layers before LRelu layers in generators, discriminators, we add tanh layer as last layer in the generator.

### 4.5.2 Quantitative Results

We evaluate the performance of a generative model on a sample of 30 generated images, it is common to use a combination of metrics to assess different aspects of the generated samples.

Three commonly used metrics are L1 (mean absolute error), and Cosine Similarity.

#### **4.5.2.2 L1 (mean absolute error)**

L1 measures the pixel-wise difference between the real and generated images. A lower L1 score indicates better similarity between the two images.

To calculate L1, we first need to resize both the real and generated images to the same size. We can then calculate the mean absolute error between the two images using the formula:

Where n is the total number of pixels in the image, is the pixel value in the real image, and is the pixel value in the generated image.

#### **4.5.2.3 Cosine Similarity**

Cosine Similarity measures the similarity between the feature representations of the real and generated images. A higher Cosine Similarity score indicates better similarity between the two feature representations.

To calculate Cosine Similarity, we first need to extract the feature representation of both the real and generated images using a pre-trained feature extractor network. We can then calculate the cosine similarity between the two feature representations using the formula:

Where u and v are the feature representations of the real and generated images, and ‘.’ Denotes the dot product.

Once we have computed these metrics for the 30 generated images, we can use them to evaluate the performance of the generative model. It is important to note that no single metric can fully capture the quality of the generated images, So, use a combination of metrics to get a more comprehensive evaluation.

#### **4.5.2.4 Evaluation Metric**

|  | **Our Model** | **Paper’s Model** |
| --- | --- | --- |
| **Epochs** | DECODER: 130  ENCODER: 10 | DECODER: Pre-trained  ENCODER: 10 |
| **Time** | DECODER: 16 H  ENCODER: 1 H | DECODER: Pre-trained  ENCODER: 45 H |
| **Data Size** | 15 GB | 1.5 TB |
| **Results** | L1 Loss : 36650.0  Cos Sim Loss : 0.01 | L1 Loss : 8.34  Cos Sim Loss : 10.92 |

N.B : Due to a huge problem in data availability and its usage on google drive, the model wasn’t completed.

# Chapter 5: GANs Model

## 4.1 Environment Setup & Tools

We created this project with lots of scripts. We used “python” programming language because it was used in most of the recently published work, also it was the easiest, most useful language to use due to its libraries, as some of them are designed specifically to help develop deep learning and machine learning models.

### **4.1.1 Environment**

Localhost

* While developing our scripts (preprocessing, training, and testing), we tried running them on Google Colab, but our dataset was bigger than 80 GB, so we ran our scripts locally on Jupyter Notebooks.

**Table 1. Laptop Specs**

| GPU | RTX 2070 8 GB |
| --- | --- |
| RAM | 32 GB |

### **4.1.2 Packages and Libraries**

1. **PyTorch**: PyTorch is an optimized tensor library for deep learning using GPUs and CPUs.
2. **NumPy**: NumPy is the fundamental package for scientific computing in Python.
3. **Pandas**: Pandas is an open-source, Python package that is most widely used for data science/data analysis and machine learning tasks.
4. **SciPy**: SciPy (pronounced “Sigh Pie”) is an open-source software for mathematics, science, and engineering.
5. **Pydub**: Manipulate audio with a simple and easy high level interface.
6. **OS**: This module provides a portable way of using operating system-dependent functionality.

## 4.2 Dataset

The Dataset consists of the voice recordings that are from the Voxceleb [1] dataset and the face images that are from the manually filtered version of VGGFace [2] dataset. Both datasets have identity labels. We use an intersection of the two datasets with the common identities, leading to 149,354 voice recordings and 139,572 face images of 1,225 persons. We use the whole dataset for training and testing on any recorded voice, Dataset sample in figure 4.1.



**Figure 4.1 VGGFace Sample**

## 4.3 Data Preprocessing

### 4.3.1 Audio Data Preprocessing

1. Use a voice activity detector interface from the WebRTC project to isolate speech-bearing regions of the recordings.
2. Extract 64-dimensional log mel-spectrograms using an analysis window of 25ms, with a hop of 10ms between frames.
3. Perform mean and variance normalization of each mel-frequency bin.
4. We randomly crop an audio clip around 3 to 8 seconds for training, but use the entire recording for testing. And if the audio is less than 10 seconds, we repeat it until reaches 10 seconds.

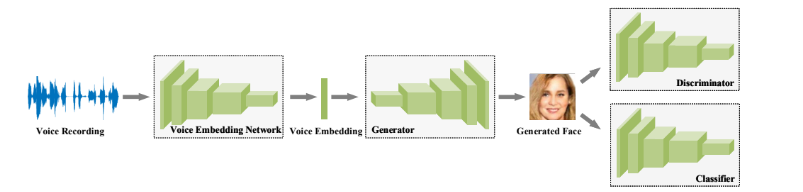
### **4.3.2 Face Data Preprocessing**

1. The cropped RGB face images of size 64 × 64 × 3 are obtained by similarity transformation.
2. Each pixel in the RGB images is normalized by subtracting 127.5 and then dividing by 127.5.

## 4.4 Model

### 4.4.1 General Model

Architecture start with a voice recording as input and applied the pre-trained Voice Embedding Network on it, to extract the Voice Embedding vector then the Generator take the Voice Embedding to generate the image and gives the Discriminator a real image and generated image to decide is it real or fake then classifier learns to assign any real face image to its identity label According to the loss function.



**Figure 4.2 General Model**

### 4.4.2 Voice Embedding Network

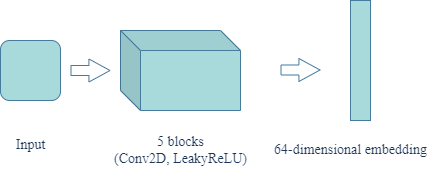
We used 1D convolutional layers with a kernel size of 3, where the stride and padding are 2 and 1, respectively. Each convolutional layer is followed by a batch normalization layer and Rectified Linear Units (ReLU). The output shape is shown accordingly, where . The final outputs are pooled over time, yielding a 64-dimensional embedding like in figure 4.3. We used this model with pretrained weights.



**Figure 4.3 Voice Embedding Network**

### 4.4.3 Face Embedding Network

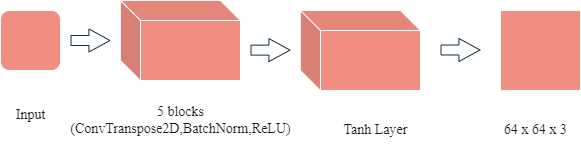
We used a 2D convolutional layer with a kernel size of 1, where the stride and padding are 1 and 0, respectively, as an input layer, but the other convolutional layer with a kernel size of 4, where the stride and padding are 2 and 1, respectively. Each convolutional layer is followed by a Leaky ReLU layer as in figure 4.4. The final output yields a 64-dimensional embedding.



**Figure 4.4 Face Embedding Network**

### 4.4.4 Generator

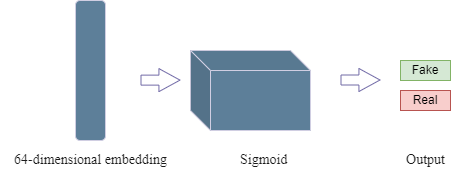
We used a 2D convolutional Transpose layer with a kernel size of 4, where the stride and padding are 1 and 0, respectively, as an input layer, but the other convolutional Transpose layer with a kernel size of 4, where the stride and padding are 2 and 1, respectively. Each convolutional Transpose layer is followed by a Batch Normalization 1D layer and ReLU layer. And the last layer in the model is a 2D convolutional Transpose layer with a kernel size of 4, where the stride and padding are 1 and 0, respectively, followed by a Tanh layer as the last layer. Final outputs yielding a 64 × 64 × 3 image like in figure 4.5.



**Figure 4.5 Generator Model**

### 4.4.5 Discriminator (Classifier)

We used the results of the generator model, which is a 64 × 64 × 3 image. We feed it to the Face Embedding Network, and get the results of Face Embedding Network, which is a 64-dimensional embedding. And feed the discriminator with this embedding, then classify it if it’s real or fake. By only a Sigmoid, take as input the image and output 0 or 1 like in figure 4.6.



**Figure 4.6 Discriminator Model**

## 4.5 Experiments & Results

### 4.5.1 Experiments

We built the same architecture as in [3] but changed the optimizer and hyperparameters. In [3] they set the ‘Optimizer’: ‘Adam’ ‘learning rate’: 0.0002, ‘beta1’: 0.5, ‘beta2’: 0.999. We tried different hyperparameters, but finally we set the ‘Optimizer’: ‘RMSProp’, alpha: 0.9, eps: 1e-08, weight\_decay=0, momentum=0, centered=False. We also add some extra layers such as we add batch normalization layers before LRelu layers in generators, discriminators, we add tanh layer as the last layer in the generator.

### 4.5.2 Quantitative Results

We evaluate the performance of a generative model on a sample of 30 generated images, it is common to use a combination of metrics to assess different aspects of the generated samples. Three commonly used metrics are FID (Fréchet Inception Distance), L1 (mean absolute error), and Cosine Similarity.

#### **4.5.2.1 FID (Fréchet Inception Distance)**

FID measures the distance between the distribution of real images and the distribution of generated images. A lower FID score indicates better similarity between the two distributions. To calculate FID, we first need to compute the mean and covariance of the feature representations of real and generated images using a pre-trained Inception network. We can then calculate the distance between these two distributions using the formula:

Where and are the mean feature representations of the real and generated images, and are their covariance matrices.

#### **4.5.2.2 L1 (mean absolute error)**

L1 measures the pixel-wise difference between the real and generated images. A lower L1 score indicates better similarity between the two images. To calculate L1, we first need to resize both the real and generated images to the same size. We can then calculate the mean absolute error between the two images using the formula:

Where n is the total number of pixels in the image, is the pixel value in the real image, and is the pixel value in the generated image.

#### **4.5.2.3 Cosine Similarity**

Cosine Similarity measures the similarity between the feature representations of the real and generated images. A higher Cosine Similarity score indicates better similarity between the two feature representations. To calculate Cosine Similarity, we first need to extract the feature representation of both the real and generated images using a pre-trained feature extractor network. We can then calculate the cosine similarity between the two feature representations using the formula:

Where u and v are the feature representations of the real and generated images, and ‘.’ Denotes the dot product. Once we have computed these metrics for the 30 generated images, we can use them to evaluate the performance of the generative model. It is important to note that no single metric can fully capture the quality of the generated images, So, use a combination of metrics to get a more comprehensive evaluation.

#### **4.5.2.4 Evaluation Metric**

Based on the evaluation metric of Fréchet Inception Distance (FID), our general model outperforms other models. However, when considering the metric of L1, the female model demonstrates superior performance. On the other hand, the paper model exhibits greater effectiveness in terms of cosine similarity. As shown in Table 2.

**Table 2. Evaluation Metric**

|  | FID | L1 | Cos Similarity |
| --- | --- | --- | --- |
| General Model | **114.4445991** | 61.9460271 | 0.244882 |
| Females Model | 126.365874 | **29.41308** | 0.399265 |
| Males Model | 129.303104 | 33.085962 | 0.2090025 |
| Paper | 117.42833 | 59.494304 | **0.33712798** |

### 4.5.3 Qualitative Results

The study was conducted to evaluate the qualitative performance of different generative models in image generation tasks. Specifically, we aimed to compare the quality of the images generated by a general model, a female model, and a male model, in terms of their realism, diversity, coherence, and visual appeal as shown in table 3.

**Table 3. Qualitative Metric**

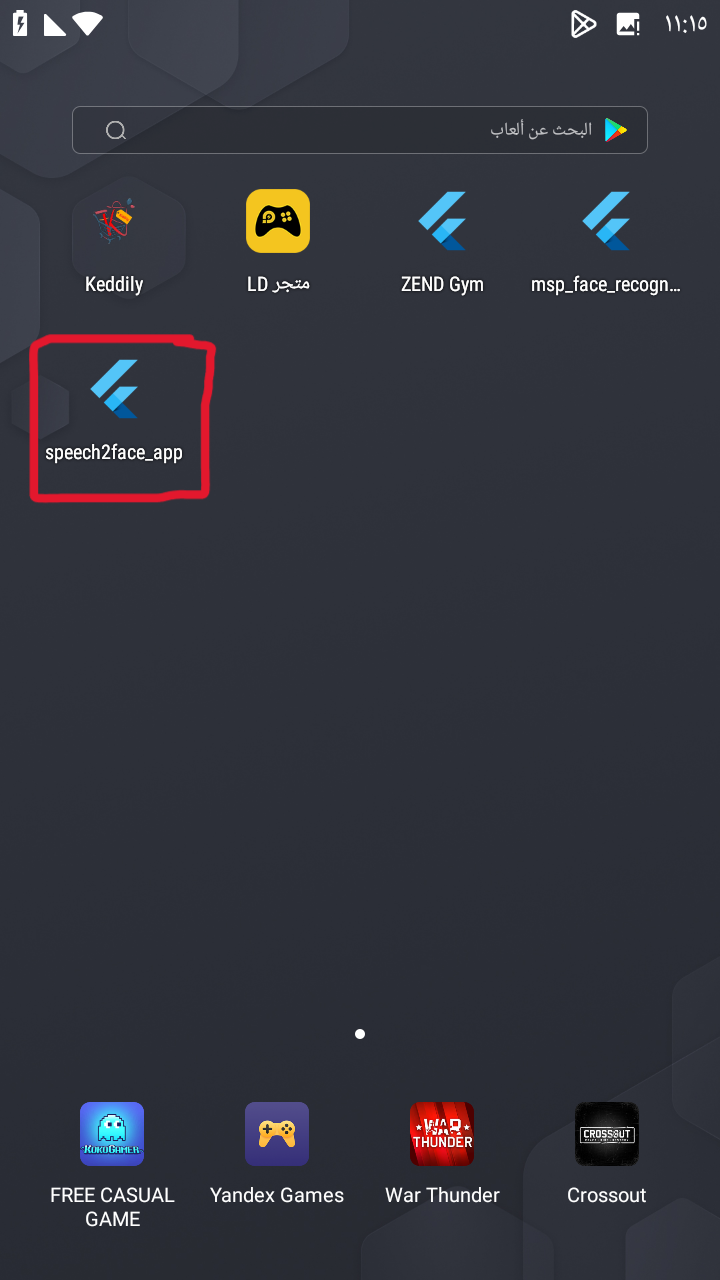
| Real image | General Model | Male model | Female model |
| --- | --- | --- | --- |
|  |  |  | **—** |
|  |  |  | **—** |
|  |  |  | **—** |
|  |  |  | **—** |
|  |  |  | **—** |
|  |  | **—** |  |
|  |  | **—** |  |
|  |  | **—** |  |
|  |  | **—** |  |
|  |  | **—** |  |

# Chapter 6: User Manual

The following steps describe in detail how to operate the speech2face application.

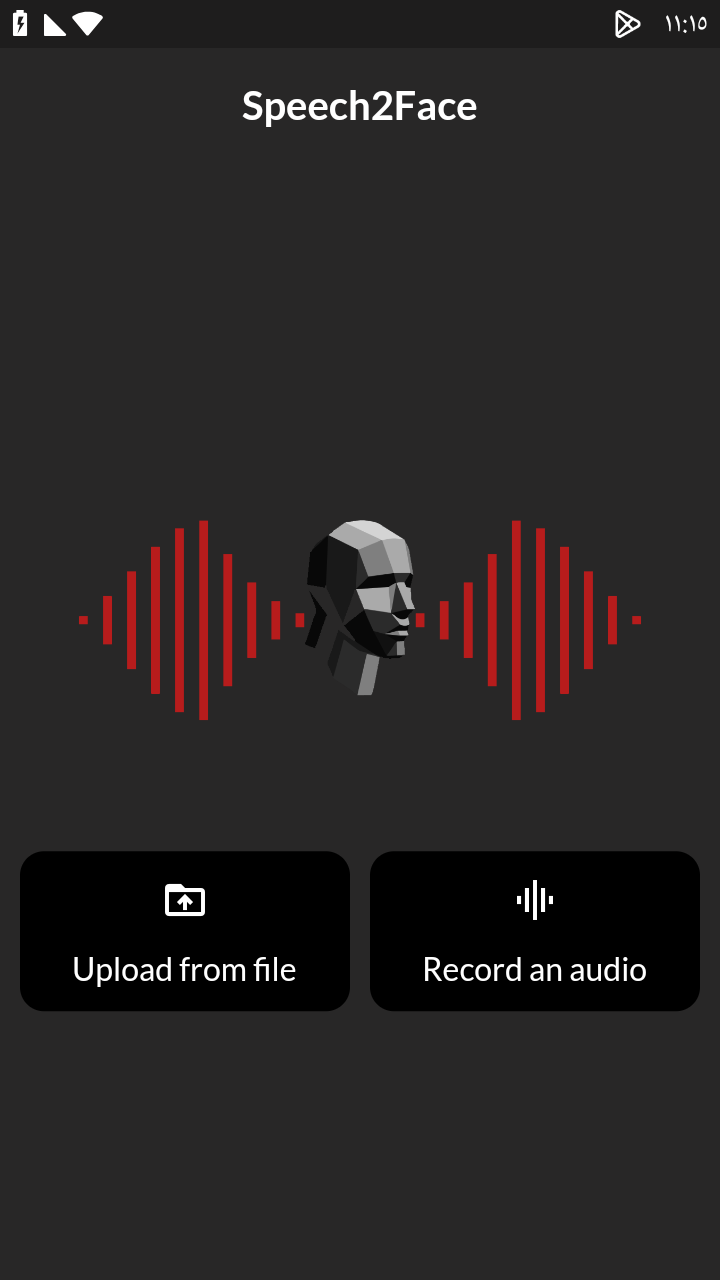
## **Step 1:**

install the application on an (android or ios) device.



## **Step 2:**

Run the application on the mobile device.

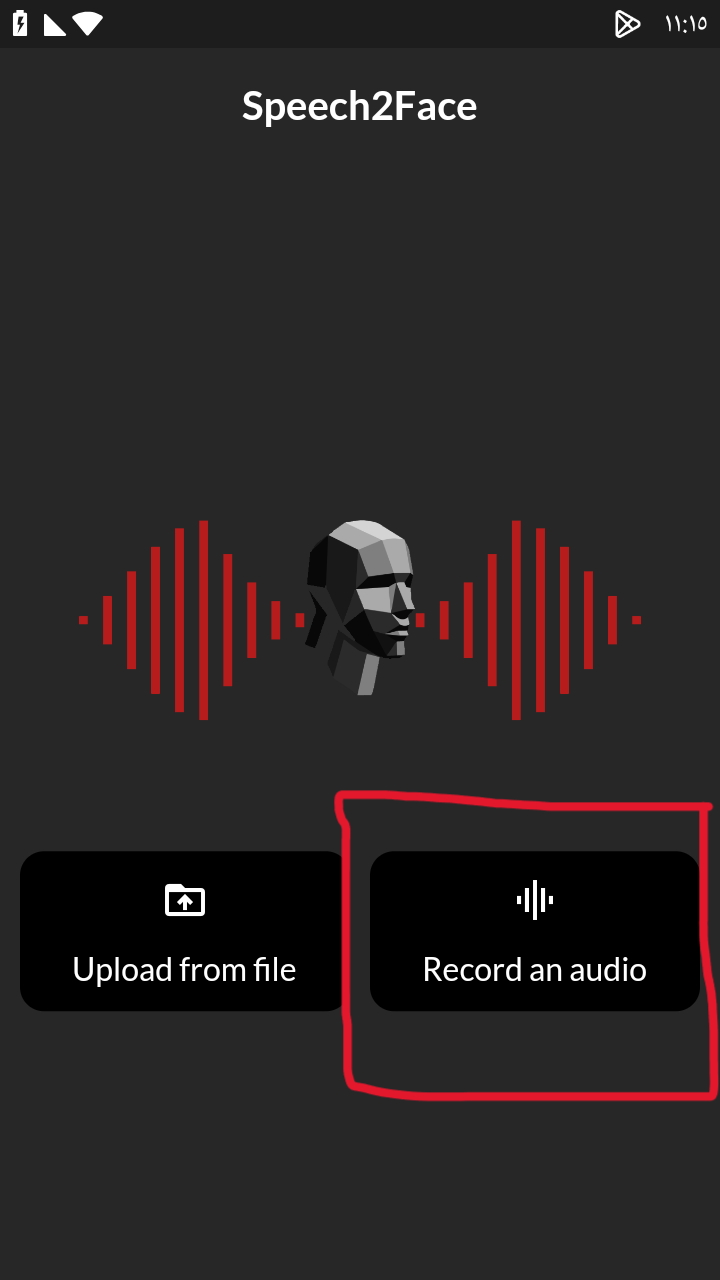


## 

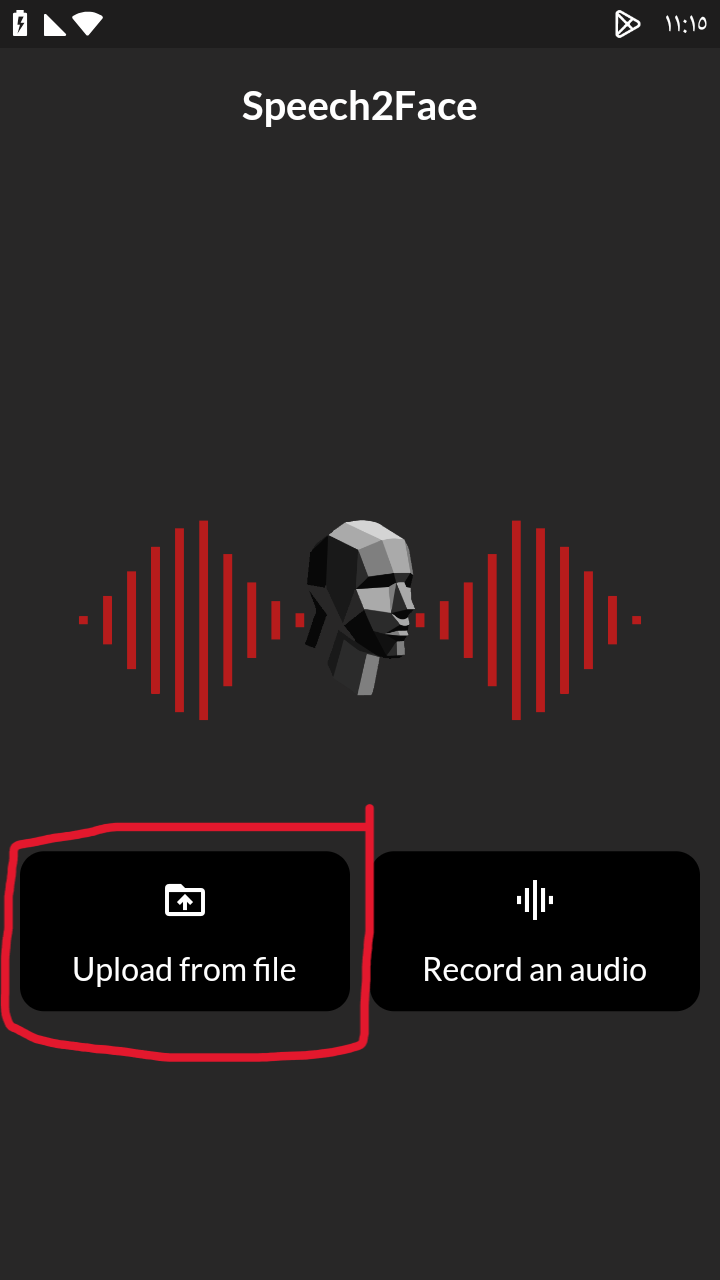
## **Step 3:**

Users have 2 options:

**Step 3.1:** if they press on record an audio, then they will go to screen to record a voice that they want to see the face from it,

**Step 3.2:** or they can upload an audio from a file without recording

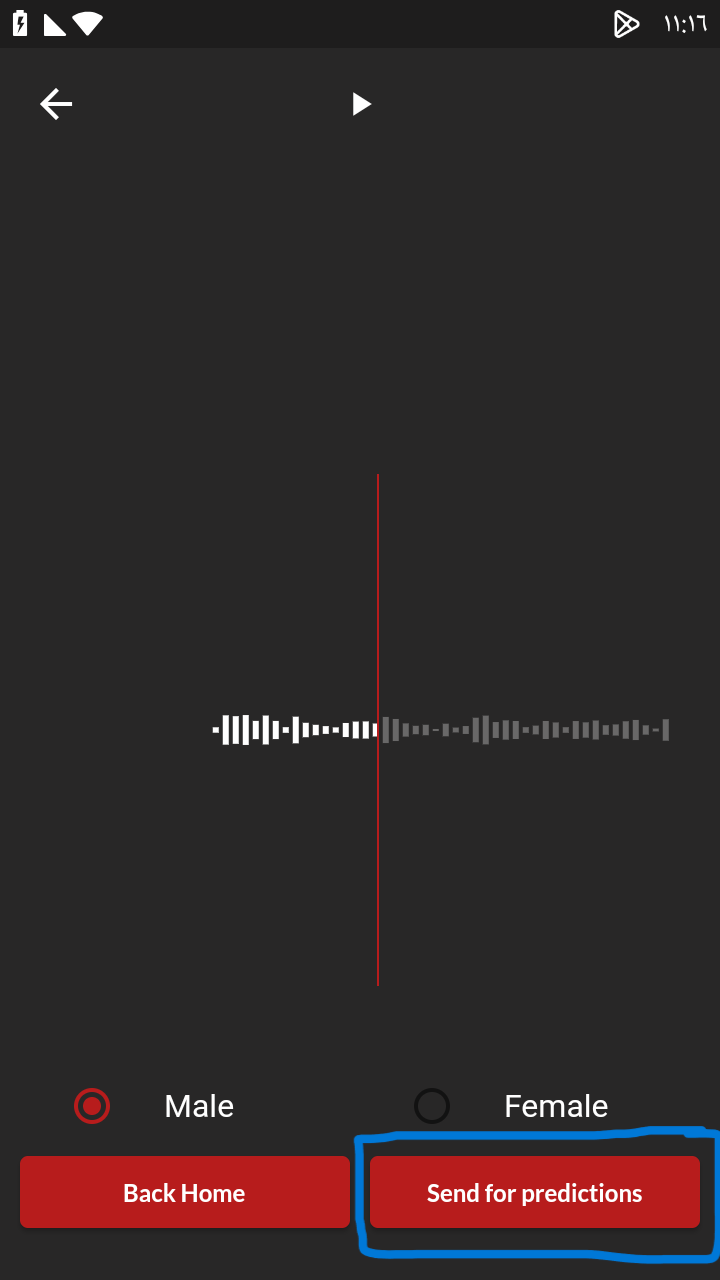


## **Step 4:**

choose the gender of the person that you want to know his/her face from the audio

## 

## **Step 5:**

press on send for prediction to see the result

## 

## Step 6:

press back to home to operate again