**Voice Analysis and task Implementation**

A Project Report

Submitted in the partial fulfilment of the requirements for the award of the degree of

**Bachelor of Technology in**

**Department of Artificial Intelligence and Data Science**

By

2000080110 Anuhya Velagapudi

2000080123 Kode Likhitha

2000080138 K.K.K.Kusuma

2000080 Divya Sri

Under the supervision of

Mrs. Lakshmi Lalitha Vuyyuru



**DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**

**K L (Deemed to be) University**

Green Fields, Vaddeswaram, Guntur District – 522502

**(2020-2021)**

**ABSTRACT**

Voice recognition is a technology used to recognize a particular voice. Voice signals form the basis for the speaker identification. We can use voice targeting in many areas like voice mail, database access etc. One of the main applications of voice recognition is for security, one can enter based on their voice for verification. Speech is the basic form of communication between people. It is the process of converting speech sounds into corresponding text. Speech recognition technology has grown dramatically over the last few years. However, there are many important research challenges e.g. differences in speaker and language, environmental sound and word size etc. The purpose of our project is to present a holistic view of the acceptance of speech that describes various processes and summarizes the various methods used in the standard speech system.

**1.INTRODUCTION**

Voice is a basic, common, and effective way for people to communicate with one another. Today speech technology is available in a limited but admirable range of functions. This technology allows machines to respond appropriately and reliably to human voices and provide useful and valuable services.

As computer communication is faster than using a keyboard, then people will prefer that program. This can be done by developing a voice recognition system: computer-to-speech that allows the computer Translating voice request and calling from text. Voice recognition system: process-to-text is a process of to convert an acoustic signal that is held using a microphone to a set of words. Fundamentally, Speech Recognition, also known as the Automatic Speech Recognition (ASR) process alters speech signal. The speech recognition platform aims to develop speech recognition strategies.

The early computer programs were limited to scale and power. But a change in computer technology is already underway the default speech recognition field. Now a day it's easy to keep the big details of speech recognition due advances in computer technology. There are only a few languages that speech recognition systems have developed. Thus, many dimensions exist to construct the native language expressions. Automatic Expression recognition has limited human efforts in many fields, such as automated telephone processing on the telephone network, data entry, voice calling; is a question based on recycled travel and booking information, an understanding of natural languages translators etc.

Our Implementation goes with developing a module that recognises speech and identifies the meaning of it. Later based on user wish it performs the tasks. To make it easy and comfortable for user I also developed a GUI using Tkinter and Pygame Packages.

The project involves four different applications – sending email, recognizing voice , displaying the speech and generating voice for given text.

The papers and thesis used to develop this hybrid model are mentioned below.

1. **LITERATURE SURVEY**

**VOICE RECOGNITION:**

The voice recognition system is broadly divided into two ways

A. To identify the Speaker

B. Speaker authentication

The process of identifying the voice of a speech given to a given group of speakers is called a speaker identification a speaker with its highest vibrations is the same as the voice that has been served a speaker with its unmatched voice features eligible for a new entry into the database. Known sets of the sounds programmed in two edges are called Open-set mode and Open-set mode. In open-set mode the speaker does not need to be a part of some known speakers. This method is used for biometric proofing purposes security to identify an authorized person. Conversely, speaker authentication the process is the process of accepting or rejecting a speaker's claim. It is used to convince a person's request for authenticity.

The Speaker's verification procedure is usually referred to as the open set mode it needs to check the authenticity of the voice from the set of speakers. So there are three ways to distinguish a reliable voice recognition system - The text depends and Text is independent, Open set and Close set and voice recognition and voice verification. The sound of talking says was captured using a microphone to turn it into an electrical signal. Purpose of sound card inside computer Convert an analog signal to a digital signal. The sound card has the ability to store and play this speech signal.

Known Samples

Identified

Unknown speech is of speaker X

Sample of speaker 1

Unknown Sample Speech

Could not be idientified

No match found

Sample of speaker 2

\_\_\_\_\_\_\_

New entry in database

Sample of speaker n

**Types of speech:**

Speech recognition is divided by what kind of words they can understand. They categorized as:

1) **Remote Name:** Remote Identifier sometimes requires each spoken word to be silent (lack of sound) signal) in the bot

2) **Connected Name:** Same with a different name, but allows different expressions to work together " contains at least a break between them.

3) **Continuous Speech:** allows users to speak naturally and similarly the computer will decide content.

4) **Sound Speech:** It is a kind of speech that is loud and ineffective. Speech recognition systems can be of different types depending on the types of words to be observed.

This section provides a review of the development of speech recognition technologies. There is also a brief discussion of Various methods has been used to improve the recognition process. A survey of the last few years involved technology has reported significant improvements in the sector.

• The first attempts to develop automatic talk recognition was made in the 1950s when many researchers tried to explore the basic acoustic-phonetic concept.

• In 1952 Davis et.al (1952), set out to establish a motive for digital recognition. The proposed plan was in place we use the concept of spectacle to show the digital vowel region.

• Olson and Belar (1956), attempted it find ten different characters for a single speaker composed of 10 monosyllabic words.

• In 1959, Fry and Denes tried to build phonetic identifier for vowels and analytical concrete. the screen and the similarity. In 1960, a hardware-based approach emerged, where several Japanese laboratories had entered this field.

• Suzuki and Nakata (1961), developed hardware for the recognition of vowels.

• Sakoe and Chiba in Japan propose the importance of robust synchronization systems. In 1970, the recognition of speech-based speech in individual words was a major focus of investigators. Line Predictive Coding (LPC) was optimized for low-level coding and it worked to use speech recognition systems for optimizing its visual parameters.

• Pruthi et al. (2000), developed a one-time word-dependent word recognition. Continuous HMM was used to see a different name for that Hindi language in 2006 was designed by Gupta. Al- Qatab et. al (2010), done is an Arabic speaking system that uses HTK that can recognize both isolated and continuous words.

**Applications:**

• Search for reports or documents on your computer.

• Create graphs or tables using data.

• State the information you want to include in the document.

• Print documents on request.

• Start video conferencing.

• Schedule meetings.

• Record minutes.

• Make travel arrangements

**Existing Models for Speech Recognition:**

1. Speech Recognition Using Google Cloud Speech API.
2. Speech Recognition Using Deep Neural Networks.
3. Speech Recognition Using Hidden Markov Models.

**SEND EMAIL:**

SMTP model is used to send emails to the requested address

The SMTP design is based on the following model of communication: as the result of a user mail request, the sender-SMTP establishes a two-way transmission channel to a receiver-SMTP. The receiver-SMTP may be either the ultimate destination or an intermediate. SMTP commands are generated by the sender-SMTP and sent to the receiver-SMTP. SMTP replies are sent from the receiver-SMTP to the sender-SMTP in response to the commands.

RECIEVER

SENDER EMAIL--SMTP

USER

FILE SYSTEM

FILE SYSTEM

The first step in the procedure is the MAIL command. The contains the source mailbox. MAIL FROM: This command tells the SMTP-receiver that a new mail transaction is starting and to reset all its state tables and buffers, including any recipients or mail data.

The second step in the procedure is the RCPT command. RCPT TO: This command gives a forward-path identifying one recipient. If accepted, the receiver-SMTP returns a 250 OK reply, and stores the forward-path.

The third step in the procedure is the DATA command. DATA If accepted, the receiver-SMTP returns a 354 Intermediate reply and considers all succeeding lines to be the message text.

**TEXT TO SPEECH:**

Text-to-speech (TTS) is a type of assistive technology that reads digital text aloud. It’s sometimes called “read aloud” technology. The voice in TTS is computer-generated and reading speed can usually be sped up or slowed down. Voice quality varies, but some voices sound human. There are even computer-generated voices that sound like children speaking. Many TTS tools highlight words as they are read aloud. This allows kids to see text and hear it at the same time. Some TTS tools also have a technology called optical character recognition (OCR). OCR allows TTS tools to read text aloud from images. For example, your child could take a photo of a street sign and have the words on the sign turned into audio.

## Types of text-to-speech tools:

Depending on the device your child uses, there are many different TTS tools:

* **Built-in text-to-speech:** Many devices have built-in TTS tools. This includes desktop and laptop computers, smartphones and digital tablets and Chrome. Your child can use this TTS without purchasing special apps or software.
* **Web-based tools:** Some websites have TTS tools on-site. For instance, you can turn on our website’s “Reading Assist” tool, located in the lower left corner of your screen, to have this webpage read aloud. Also, kids with dyslexia may qualify for a free Bookshare account with digital books that can be read with TTS. There are also free TTS tools available online.
* **Text-to-speech apps: Kids can also download TTS apps on smartphones and digital tablets.** These apps often have special features like text highlighting in different colors and OCR. Some examples include Voice Dream Reader, Claro ScanPen and Office Lens.
* **Chrome tools:** Chrome is a relatively new platform with several TTS tools. These include Read&Write for Google Chrome and Snap&Read Universal. You can use these tools on a Chromebook or any computer with the Chrome browser. See more Chrome tools to help with reading.
* **Text-to-speech software programs:** There are also several literacy software programs for desktop and laptop computers. In addition to other reading and writing tools, many of these programs have TTS.

**pyttsx** is a cross-platform text to speech library which is platform independent. The major advantage of using this library for text-to-speech conversion is that it works offline.

Diagram

Description automatically generated with medium confidence

**GRAPHIICAL USER INTERFACE:**

GUI is implemented in our project to make easy and comfortable usage of our module

Tkinter and Pygame packages are used to effectively interact with user.

Standard builds of Python include an object-oriented interface to the Tcl/Tk widget set, called tkinter. This is probably the easiest to install (since it comes included with most binary distributions of Python) and use. For more info about Tk, including pointers to the source, see the Tcl/Tk home page. Tcl/Tk is fully portable to the macOS, Windows, and Unix platforms.

Pygame GUI is a module to help you make graphical user interfaces for games written in pygame. Theme-able UI elements/widgets - you can use JSON theme files to change the colours, fonts and other appearance related details of your UI without touching your code. A subset of HTML is supported for drawing word-wrapped text. Have bold styled words in the middle of a paragraph of text! Stick a link in there! Go absolutely hog wild, within the bounds of the defined subset of HTML the module supports. Buttons, text entry, scroll bars and drop down menus all supported, with more on the way. A window stack that will let you keep a bunch of moveable windows of 'stuff' around and correctly sorted. Support for localizing your GUI into different languages. As closely respecting of the pygame way of doing things as possible.

**3. PROPOSED WORK**

**3.1 Importing all the packages:**

All the required packages are mentioned below and the commands to install and import are as follows

#pip install google-api-python-client

#pip install SpeechRecognition

#pip install PyAudio

#pip install pyttsx3

#pip install pywin32

**Module 1**: Develop Speech Recognition Model

**Module 2**: Develop Text to Speech Converter

**Module 3**: Develop SMTP protocol

**Module 4**: Develop GUI connecting these models

**Requirements:**

1. Python (version 3.7 or higher).
2. Google Colab.
3. Jupyter notebook.
4. SpeechRecognition package.
5. Pyaudio package.
6. Tkinter
7. Smtplib
8. Pygame
9. Pyttsx3

**Methodology:**

To develop **SMTP** we used smtplib an inbuilt python package to send/manipulate mail data

Here is the detail of the parameters −

* **host** − This is the host running your SMTP server. You can specify IP address of the host or a domain name like tutorialspoint.com. This is optional argument.
* **port** − If you are providing *host* argument, then you need to specify a port, where SMTP server is listening. Usually, this port would be 25.
* **local\_hostname** − If your SMTP server is running on your local machine, then you can specify just *localhost* as of this option.

An SMTP object has an instance method called **sendmail**, which is typically used to do the work of mailing a message. It takes three parameters −

* The *sender* − A string with the address of the sender.
* The *receivers* − A list of strings, one for each recipient.
* The *message* − A message as a string formatted as specified in the various RFCs.

To send the mail you use *smtpObj* to connect to the SMTP server on the local machine and then use the *sendmail* method along with the message, the from address, and the destination address as parameters (even though the from and to addresses are within the e-mail itself, these aren't always used to route mail).

To send an e-mail with mixed content requires to set **Content-type** header to **multipart/mixed**. Then, text and attachment sections can be specified within **boundaries**.

A boundary is started with two hyphens followed by a unique number, which cannot appear in the message part of the e-mail. A final boundary denoting the e-mail's final section must also end with two hyphens.

Attached files should be encoded with the **pack("m")** function to have base64 encoding before transmission.

To develop **SPEECH TO TEXT** module we used google-api-python-client which used 4 different ML models to predict speech and give output based on voting mechanism.

Before going to learn about this package we need to know about google apis. Google APIs give you programmatic access to Google Maps, Google Drive, YouTube, and many other Google products. To make coding against these APIs easier, Google provides client libraries that can reduce the amount of code you need to write and make your code more robust.

Deep Neural Networks (DNNs) [revolutionized the field of speech recognition](http://googleresearch.blogspot.com/2012/08/speech-recognition-and-deep-learning.html). These multi-layer networks distinguish sounds better than GMMs by using “discriminative training,” [differentiating phonetic units](https://scholar.google.com/citations?view_op=view_citation&hl=en&user=RtQA6Z8AAAAJ&citation_for_view=RtQA6Z8AAAAJ:TQgYirikUcIC) instead of modeling each one independently. It was tough to get punctuation right. The old system relied on hand-crafted rules or “grammars,” which, by design, can’t easily take textual context into account. For example, in an early test our algorithms transcribed the audio “I got the message you left me” as “I got the message. You left me.” To try and tackle this, we again tapped into neural networks, teaching an LSTM to insert punctuation at the right spots. It’s still not perfect, but we’re continually working on ways to improve our accuracy.

To develop **TEXT TO SPEECH** we used pyttsx3 package. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3.Engine instance. During construction, the engine initializes a pyttsx3.driver.DriverProxy object responsible for loading a speech engine driver implementation from the pyttsx3.drivers module. After construction, an application uses the engine object to register and unregister event callbacks; produce and stop speech; get and set speech engine properties; and start and stop event loops.

The engine factory

pyttsx3.init([driverName : string, debug : bool]) → pyttsx3.

Engine Gets a reference to an engine instance that will use the given driver.

If the requested driver is already in use by another engine instance, that engine is returned.

Otherwise, a new engine is created.

Parameters

* + driverName – Name of the pyttsx3.drivers module to load and use. Defaults to the best available driver for the platform, currently:
    - sapi5 - SAPI5 on Windows – nsss

NSSpeechSynthesizer on Mac OS X

* + - espeak - eSpeak on every other platform
  + debug – Enable debug output or not.

Raises

* + ImportError – When the requested driver is not found
  + RuntimeError – When the driver fails to initializ

Meta Data of voice:

We can specify type of voice we wish to get using this function

**age**

the voice in years. Defaults to None if unknown.

**gender**

String gender of the voice: male, female, or neutral. Defaults to None if unknown.

**id** String identifier of the voice. Used to set the active voice via pyttsx3.engine.Engine. setPropertyValue(). This attribute is always defined.

**languages** List of string languages supported by this voice. Defaults to an empty list of unknown.

**name** Human readable name of the voice. Defaults to None if unknown

### 

**4.BLOCK DIAGRAM**

Develop GUI

Generate Speech from text

Send Email with recognized text

Recognize Speech

Set SMTP with desired sender,receiver,host

User input through text box

Accept user speech

Specify meta data of voice

Recognize Speech

Generate voice

Update output of message

Cannot

recognize

Words are trained

NO YES

1. **IMPLEMENTATION**

import speech\_recognition as sr

import pygame

import smtplib

import time

recording = sr.Recognizer()

**#speech reecognizer**

def googleapirec():#recognize text

    with sr.Microphone() as source:

        recording.adjust\_for\_ambient\_noise(source)

        print("Please Say something:")

        audio = recording.listen(source)

        try:

            res=recording.recognize\_google(audio)

            print("You said: " + res)

            return res

        except Exception as e:

            print(e)

            return "cannot recognize..Sorry!!"

#**SMTP protocol**

def send():#send email-SMTP

    text = googleapirec()

    print(text)

    message\_display(text)

    gmail\_user = 'undivinable@gmail.com'

    gmail\_password = 'anuhya11421'

    sent\_from = gmail\_user

    to = 'wellanuhya@gmail.com'

    subject = 'Email sent by'+'VRAI software!!'

    body = text

    email\_text = """\

    From: %s

    To: %s

    Subject: %s

    %s

    """ % (sent\_from, ", ".join(to), subject, body)

    try:

        smtp\_server = smtplib.SMTP\_SSL('smtp.gmail.com', 465)

        smtp\_server.ehlo()

        smtp\_server.login(gmail\_user, gmail\_password)

        smtp\_server.sendmail(sent\_from, to, email\_text)

        smtp\_server.close()

        print ("Email sent successfully!")

    except Exception as ex:

        print ("Something went wrong….",ex)

#**GUI**

**#Display message**

def message\_display(text):#message formatting

    largeText = pygame.font.Font("freesansbold.ttf",30)

    TextSurf, TextRect = text\_objects(text, largeText)

    TextRect.center = ((display\_width/2),(display\_height/2))

    gameDisplay.blit(TextSurf, TextRect)

    pygame.display.update()

def text\_objects(text, font):

    textSurface = font.render(text, True, alpha)

    return textSurface, textSurface.get\_rect()

**#add button**

def button(msg,x,y,w,h,ic,ac,action=None):#display button

    mouse = pygame.mouse.get\_pos()

    click = pygame.mouse.get\_pressed()

    if x+w > mouse[0] > x and y+h > mouse[1] > y:

        pygame.draw.rect(gameDisplay, ac,(x,y,w,h))

        if click[0] == 1 and action != None:

            action()

    else:

        pygame.draw.rect(gameDisplay, ic,(x,y,w,h))

    smallText = pygame.font.SysFont("freesansbold.ttf",30)

    textSurf, textRect = text\_objects(msg, smallText)

    textRect.center = ( (x+(w/2)), (y+(h/2)) )

    gameDisplay.blit(textSurf, textRect)

#**Text to Speech**

from tkinter import \*

import pyttsx3

def talk(text):

    engine=pyttsx3.init()

    engine.say(text.get())

    engine.runAndWait()

    text.delete(0,END)

def t2s():

    root=Tk()

    root.title("Text to speech")

    root.geometry("800x500")

    text=Entry(root,font=("Helvetica",20))

    text.pack(pady=20)

    engine=pyttsx3.init()

    engine.say(text.get())

    engine.runAndWait()

    text.delete(0,END)

    bu=Button(root,text="Speak!",command=lambda: talk(text))

    bu.pack(pady=40)

    root.mainloop()

#**Pygame window**

pygame.init()

display\_width = 800

display\_height = 600

black = (0,0,0)

alpha = (0,88,255)

white = (255,255,255)

red = (200,0,0)

green = (0,200,0)

bright\_red = (255,0,0)

bright\_green = (0,255,0)

gameDisplay = pygame.display.set\_mode((display\_width,display\_height))

pygame.display.set\_caption('Speech Recognition')

gameDisplay.fill(white)

Img = pygame.image.load('img.jpeg')

Img=pygame.transform.scale(Img, (800, 600))

gameDisplay.blit(Img,(0,0))

def close():#quit

    pygame.quit()

    quit()

def s2t():#speech to text

    gameDisplay.blit(Img,(0,0))

    text = googleapirec()

    print(text)

    message\_display(text)

def rs():#reset screen

    gameDisplay.fill(white)

def main():

    while True:

        for event in pygame.event.get():

            if event.type == pygame.QUIT:

                pygame.quit()

                quit()

            button("Speak!",150,500,100,50,green,bright\_green,s2t)

            button("Send Email",350,500,110,50,green,bright\_green,send)

            button("Quit",550,500,100,50,red,bright\_red,close)

            button("Text to Speech",350,400,130,50,red,bright\_green,t2s)

            pygame.display.update()

main()

**6.RESULTS AND ANALYSIS**

**Graphical user interface

Description automatically generated**

This is the Interface of our project

Graphical user interface

Description automatically generated

When Speak! Button is pressed we are supposed to speak and It will be captured. The results are 90% accurate considering all the accent and language it is designed in. (sadly I had to train this so bad to get my name correctly recognize)

Graphical user interface

Description automatically generated

When Send Email button is clicked we need to tell some body for that mail and it will be delivered.

Text

Description automatically generated this isa reference to know is mail is sent

Text, letter

Description automatically generated

This is the mail received in spam folder

Graphical user interface, text, application

Description automatically generated

When the Text to Speech button is clicked this window is opened. After entering the text in text box we can listen the speech of it. The accuracy and accent of spells are 100% accurate.

Quit button enables to close the window

**7.CONCLUSION AND FUTURE SCOPE**

The module developed can be used as an imitation to alexa with GUI i.e., a mixture of user interface and speech!

Language and speech are incredibly complex. Words have meaning, sure. So does the context of those words, the emotion behind them, and the response of the person listening. It would seem the subtleties of the spoken word would be beyond the reach of even the most sophisticated computers. But in recent years, advances in text-to-speech (TTS) technologies – the ability of computers to convert sequences of words into natural sounding, intelligible audio responses – have made it possible for computers to sound more human-like.

The way we developed speech to mail might be used in future in a very important and regular way same as phones now!. Improving the accuracy of speech recognition and personalizing user data by recognizing voice of user can be implemented further.

**8. ACKNOWLEDGEMENT**

I would like to express my special thanks of gratitude to my Guide Lakshmi Lalitha Vuyyuru as well as our HOD Dr. Tirupathi Reddy sir who gave me the golden opportunity to do this wonderful project.

**9.REFERENCES**

[1] VOICE RECOGNITION SYSTEM: SPEECH-TO-TEXT Prerana Das, Kakali Acharjee, Pranab Das and Vijay Prasad JAFS|ISSN 2395-5554 (Print)|ISSN 2395-5562 (Online)|Vol 1(2)|November 2015

[2] David Kaspar, Alexander Bailey, Patrick Fuller, [Librosa: A Python Audio Library (2019)](https://medium.com/@patrickbfuller/librosa-a-python-audio-libary-60014eeaccfb)

[3] Smith R. An overview of the Tesseract OCR engine, USA:

Google Inc; 2007.

[4] P. Melville, W. Gryc, and R. D. Lawrence. “Sentiment analysis of blogs by combining lexical knowledge with text classification.” In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp. 1275-1284. 2009.

[5] Jianqiang Z, Xiaolin G, Xuejun Z (2018) Deep convolution neural networks for twitter sentiment analysis. IEEE Access 6:23253–23260.

[6] Iqbal F et al (2019) A hybrid framework for sentiment analysis using genetic algorithm based feature reduction. IEEE Access 7:14637–14652.