Project Report

on

SPEECH SYNTHESIS

Artifical Intelligence for Data Science

By

V.ABHIRAM (2010030180)

K.SIDHARTH RAO (2010030443)

A .RAGHAVENDRA GOUD (2010030394)

R .VIVEK VARDHAN REDDY (2010030142)

under the supervision of

Dr. Arpita Gupta



Department of Computer Science and Engineering

K L University Hyderabad,

Aziz Nagar, Moinabad Road, Hyderabad – 500 075, Telangana, India.

March, 2022

**DECLARATION**

The Project Report entitled “**Speech Synthesis**”is a record of bonafide work of V.Abhiram (2010030180), K.Sidharth Rao (2010030443),A .Raghavendra Goud (2010030394), R .Vivek Vardhan Reddy (2010030142) , submitted as a requirement for the completion of the course **Artifical Intelligence for Data Science** in the Department of Computer Science and Engineering to the K L University, Hyderabad. The results embodied in this report have not been copied from any other Departments/University/Institute.

<Signature of the Students >

## CERTIFICATE

This is to certify that the Project Report entitled “**Speech Synthesis**” is being submitted by of V.Abhiram (2010030180), K.Sidharth Rao (2010030443),A .Raghavendra Goud (2010030394), R.Vivek Vardhan Reddy (2010030142) as a requirement for the completion of the course **Artifical Intelligence for Data Science** in the Department of Computer Science and Engineering, K L University, Hyderabad is a record of bonafide work carried out under our guidance and supervision.

The results embodied in this report have not been copied from any other departments/ University/Institute.

## Signature of the Supervisor

Dr. Arpita Gupta

## Signature of the HOD Signature of the Examiner

**ACKNOWLEDGEMENT**

First and foremost, we thank the lord almighty for all his grace & mercy showered upon us, for completing this project successfully.

We take grateful opportunity to thank our beloved Founder and Chairman who has given constant encouragement during our course and motivated us to do this project. We are grateful to our Principal **Dr. L. Koteswara Rao** who has been constantly bearing the torch for all the curricular activities undertaken by us.

We pay our grateful acknowledgement & sincere thanks to our Head of the Department **Dr. Chiranjeevi Manike** for his exemplary guidance, monitoring and constant encouragement throughout the course of the project. We thank **Dr. Arpita Gupta** of our department who has supported throughout this project holding a position of supervisor.

We whole heartedly thank all the teaching and non-teaching staff of our department without whom we won’t have made this project a reality. We would like to extend our sincere thanks especially to our parents, our family members and friends who have supported us to make this project a grand success.

**ABSTRACT**

A text-to-speech synthesis system typically consists of multiple stages, such as a text analysis frontend, an acoustic model and an audio synthesis module. Building these components often requires extensive domain expertise and may contain brittle design choices. In this paper we present speech synthesis in which we can convert text into speech format**,** we can convert many linguistic languages into the audio format.

Text-to-Speech (TTS) is a useful technology that converts any text into a speech signal. It can be utilized for various purposes, TTS makes it possible to dramatically improve the naturalness of synthetic speech compared with the early TTS. However, no general-purpose TTS has been developed that can consistently synthesize sufficiently natural speech. Text-To-Speech (TTS) conversion is a computerbased system that can be able to read any text aloud, whether it was directly introduced in the computer by an operator or scanned

**INDEX**

Page no

**INTRODUCTION**

Speech synthesis is the artificial production of human speech. A computer system used for this purpose is called a speech synthesizer, and can be implemented in software or hardware . A text-to-speech (TTS) system converts normal language text into speech; other systems render symbolic linguistic representations like phonetic transcriptions into speech. Text-to-speech (TTS) convention transforms linguistic information stored as data or text into speech. It is widely used in audio reading devices for blind people now a days . In the last few years however, the use of text-to-speech conversion technology has grown far beyond the disabled community to become a major adjunct to the rapidly growing use of digital voice storage for voice mail and voice response systems. Also developments in Speech synthesis technology for various languages have already taken place.

Text-to-speech synthesis -TTS - is the automatic conversion of a text into speech that resembles, as closely as possible, a native speaker of the language reading that text. Text-tospeech synthesizer (TTS) is the technology which lets computer speak to you. The TTS system gets the text as the input and then a computer algorithm which called TTS engine analyses the text, pre-processes the text and synthesizes the speech with some mathematical models. The TTS engine usually generates sound data in an audio format as the output. The text-to-speech (TTS) synthesis procedure consists of two main phases. The first is text analysis, where the input text is transcribed into a phonetic or some other linguistic representation, and the second one is the generation of speech waveforms, where the output is produced from this phonetic and prosodic information. These two phases are usually called high and low-level synthesis . A simplified version of this procedure is presented in figure 1 below. The input text might be for example data from a word processor, standard ASCII from e-mail, a mobile text-message, or scanned text from a newspaper. The character string is then pre-processed and analyzed into phonetic representation which is usually a string of phonemes with some additional information for correct intonation, duration, and stress. Speech sound is finally generated with the low-level synthesizer by the information from high-level one. The artificial production of speech-like sounds has a long history, with documented mechanical attempts dating to the eighteenth century.

LITERATURE SURVEY

|  |
| --- |
|  |

**DATASET**

|  |
| --- |
|  |

**TECNIQUES**

|  |
| --- |
|  |

**FLOW CHAT**

|  |
| --- |
|  |

**PRE-PROCESSING OF DATASET**

* **Removing Punctuations: -**
* This is a technique which can be used for removing all the punctuation marks from the text file. In this, we can add or discard the punctuation marks according to the user requirement.
* **Code used for removing the punctuations**
* filename = input("Enter filename: ")  
    
    
  def remove\_punc(string):  
   punc = '''!()-[]{};:'"\,<>./?@#$%^&\*\_~'''  
   for ele in string:  
   if ele in punc:  
   string = string.replace(ele, "")  
   return string  
    
    
  try:  
   with open(filename, 'r', encoding="utf-8") as f:  
   data = f.read()  
   with open(filename, "w+", encoding="utf-8") as f:  
   f.write(remove\_punc(data))  
   print("Removed punctuations from the file", filename)  
  except FileNotFoundError:  
   print("File not found")

**Tokenization**

Tokenization is the process of turning a meaningful piece of data, such as an account number, into a random string of characters called a token that has no meaningful value if breached.

**Code used for Tokenization**

#from future import print\_function  
from nltk.tokenize import word\_tokenize  
with open ('LasVegasFeb21\_2020.txt') as fin, open('token15.txt','w') as fout:  
 for line in fin:  
 tokens = word\_tokenize(line)  
 print(' '.join(tokens), end='\n', file=fout)

**IMPLEMENTATION**

import os  
import gtts  
from flask import Flask, render\_template, request, redirect  
#from langdetect import detect  
from playsound import playsound  
  
app = Flask(\_\_name\_\_)  
  
@app.route("/")  
def customer():  
 return render\_template('text.html')  
  
  
@app.route('/success', methods=['POST', 'GET'])  
def home():  
 if request.method == 'POST':  
 text1 = str(request.form.getlist("name"))  
 #text1 = text1.replace('[','')  
 #text1 = text1.replace(']','')  
 tts = gtts.gTTS(text1, lang="en")  
 tts.save("D:/python/aids/my-translation1.mp3")  
 playsound("D:/python/aids/my-translation1.mp3")  
 #os.remove("my-translation1.mp3")  
 return render\_template("result\_data.html", result=text1)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(debug=True)

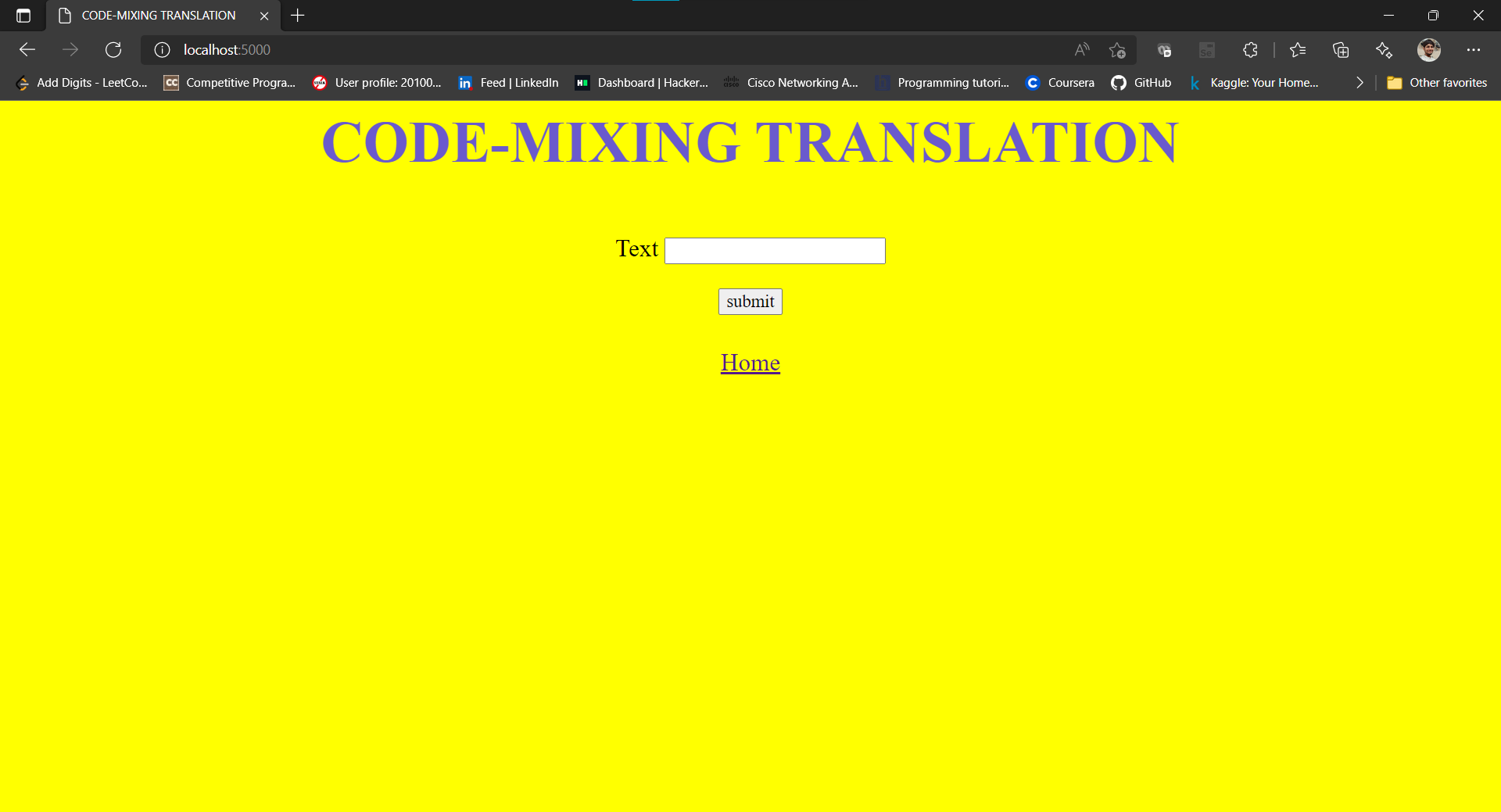
**text.html**

<html>  
<head>  
 <title>CODE-MIXING TRANSLATION</title>  
</head>  
 <body style="background-color:yellow;text-align: center;">  
 <h3 style="font-family:Amasis MT Pro;font-size: 60px;text-decoration: none;text-align: center;color:#6A5ACD;">CODE-MIXING TRANSLATION</h3>  
 <form style="text-align: center;" action = "http://localhost:5000/success" method = "POST">  
 <p style="font-family:Amasis MT Pro;font-size: 25px;">Text <input style="font-family:Amasis MT Pro;font-size: 18px;text-decoration: none;" type = "text" name = "name" /></p>  
 <p><input style="font-family:Amasis MT Pro;font-size: 18px;text-decoration: none;" type = "submit" value = "submit" /></p>  
 </form>  
 <br>  
 <a href = "/" style="font-family:Amasis MT Pro;font-size: 25px;">Home</a>  
 </body>  
</html>

**result\_data.html**

<!doctype html>  
<html>  
 <body style="background-color:powderblue;text-align: center;">  
  
 <h1 style="font-family:Amasis MT Pro;font-size: 35px;text-decoration: none;text-align: center;color:#3c3c3c;"><strong>CONVERTED AUDIO OUTPUT</strong></h1>  
 <p style="font-family:Amasis MT Pro;font-size: 28px;text-decoration: none;text-align: center;">{{ result }}</p>  
 <a href = "/" style="font-family:Amasis MT Pro;font-size: 25px;">Home</a>  
 </body>  
</html>

**OUTPUT**



**CONCLUSION**

So far we have many models on text to speech synthesis. But with this model we can choose multiple voices for giving out the speech output, and better-quality speech will be generated. This system can be improved by considering the punctuation marks while converting text to speech.

This system of Text-To-Speech can be implemented for to different languages like English, Hindi, Punjabi etc. , depending upon the user’s requirement.