A NATURAL LANGUAGE PROCESSING MODEL FOR TEXT TO SPEECH AND SPEECH TO TEXT CONVERSION

[1]G. Prudhvi Raj, [2]A.D. Satish, [3]M. Poojitha, [4]N. Lokeswari, [4]Md K Kutubuddin
[1]Student, [2]Student, [3]Student, [4]Assistant Professor, [4]Student
Department of Computer Science and Engineering,
Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, India.

***Abstract*:** Technology is getting sophisticated as days are passing by. Many inventions, research and development activities are being happening all around the world to help everyone in need. A natural language processing model for speech to text and text to speech conversion is one among those technologies. The prime motivation of the project is to convert the speech into a visible text for the user enabling the deaf to learn the information while the other part of the work which is text to speech conversion helps the blind to hear the information. Integrating a wide range of systems with loads and lots of sophisticated calculations and computations of various aspects of language and the development of wide range of systems. These systems integrate spoken language mechanism , cooperative interface to knowledge and databases, that do considerate and model the aspects of human to human interaction , multilingual interfaces , machine translation and the message understanding systems among others. Any research in the domain of natural language processing system will do comprise the most complex topics of computer science, logic, linguistics and psychology. This domain of natural language processing possess an exceptional acknowledgment in the parenthood of computer science because of its contribution and a numeral aspects in this field deal with linguistic features of computational and the natural language processing seeks to model language computationally. Gaining fluency in language is an incommodious task. Many software were having its pace in the market to make it a cakewalk.

Our prime aim is to develop a software that enhances the user's way of speech through correctness of pronunciation following the English phonetics by converting that into text. Speech-to-text-conversion is a useful tool for integrating people with hearing impairments in oral communication settings, e. g. counselling interviews or conferences. This software allows one to learn, judge and recognize their potential in English language. It also facilitates an extra add-on feature which nourishes the user's communication skills by an option of text to speech conversion also.

Keywords - **Natural Language Processing, spoken language, integrate speech, cooperative interfaces, multilingual interfaces, hearing impairments, communication skills.**

1. INTRODUCTION

1.1 Literature Survey

Machine translation (MT) is a known to be a very hard problem, because natural languages are actually very highly complex with many words that have various meanings and different possible translations, slangs may vary and meaning in the communication is also combined with many physical traits of the person who speaks it. The sound, stress and intonations were also to be considered in this aspect. The sentences might have various readings, and the relationships between linguistic entities are often vague. In addition, it is sometimes necessary to take world knowledge into account. The number of relevant dependencies is much too large, and those dependencies are too complex to take them all into account in a machine translation system. Given these boundary conditions, a machine translation system has to make various decisions (produce translations) given incomplete knowledge. This problem may be approached in several ways.[7]

2. PROPOSED METHODOLOGY FOR THE TEXT TO SPEECH AND SPEECH TO TEXT CONVERSION

The primary aim is to use the hidden markov model to produce the speech to text conversion by using the speech database. The speech signal is processed for the extraction of the excitation parameter and the spectral parameter to train the HMM. The HMM is also fed with the label of the speech signal from the speech database. The synthesis of the trained model output starts post the training with the context dependent HMMs with the text analysis from the text database to the parameter generation from the HMM. The synthesised speech is produced post the excitation parameter extraction and the spectral parameter extraction from the excitation generator and the synthesis filter respectively.

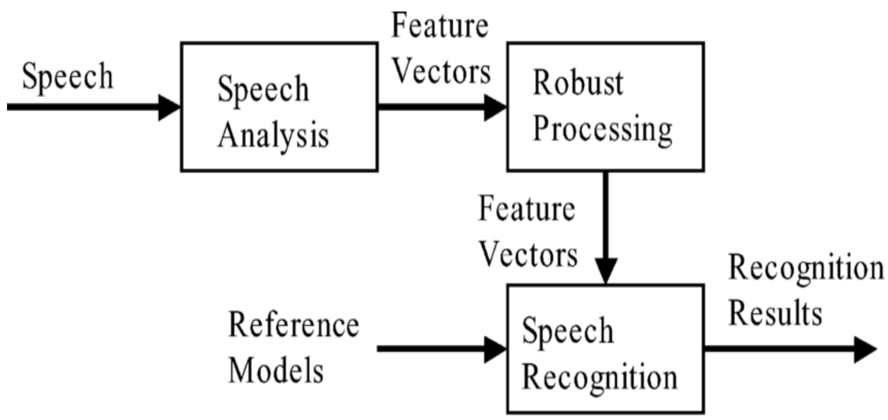


Fig 1 represents the system architecture of the hidden markov model for the speech to text conversion.

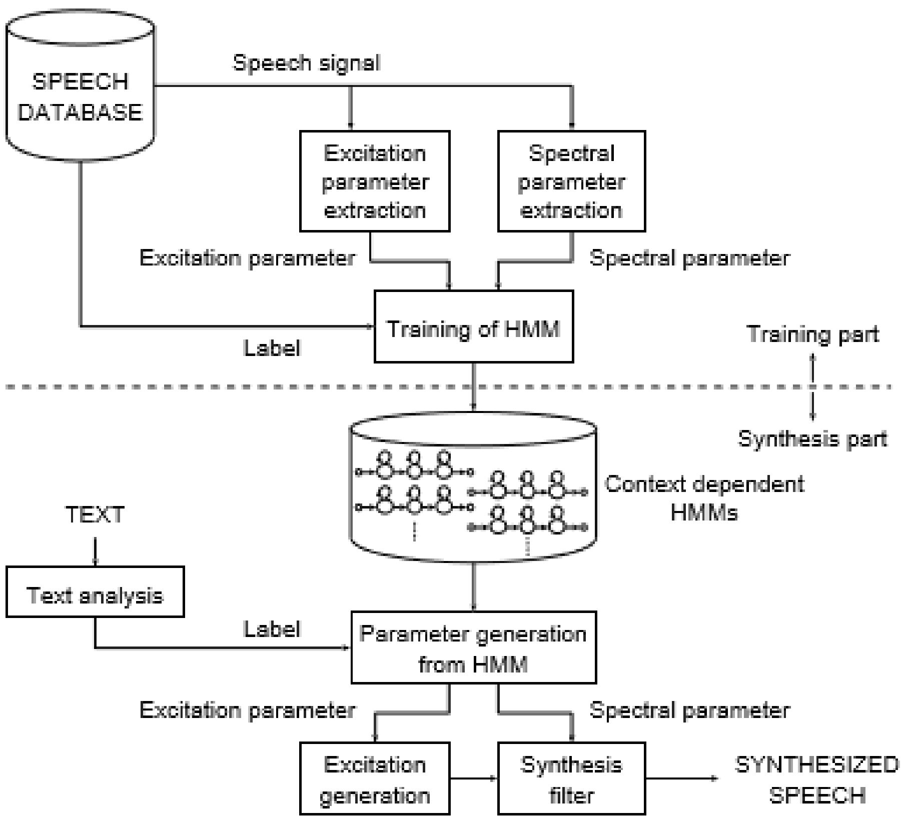


Fig 2 flow chart architecture for hidden Markov Model with DNN for text to speech conversion.

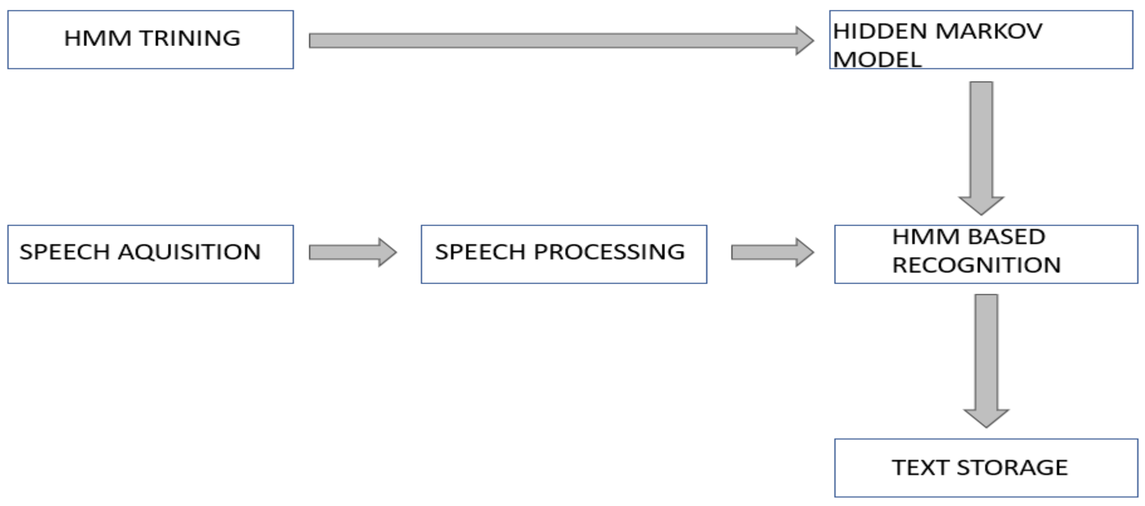


Figure 3 : System architecture of speech to text conversion model

2.1 Module Elucidation

This module consists of the following processes to be able to convert the speech signals into text format.

1. Visualizing Audio Signals - Reading from a File and Working on it
2. Characterizing the Audio Signal: Transforming to Frequency Domain
3. Recognition of Spoken Words

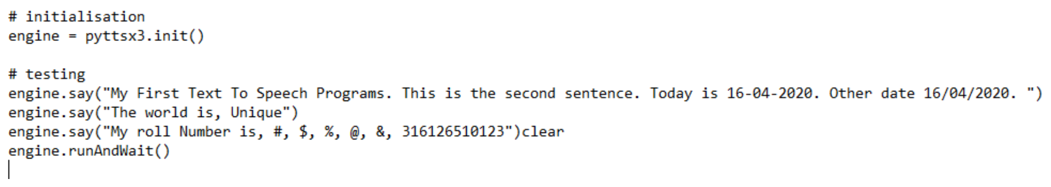
Each of the above processes are elucidated below with the sample code of execution, with its limitations and challenges encountered

We are implementing the Text To Speech and Speech To Text Synthesizing process using the python programming language which can be run on the Command Prompt in Windows or Terminal in Linux Systems. Here we are using the help of Command prompt in Windows for executing the Python programs. We particularly chosen python because it was the most widely used language by all the people and lot of enhancements can be made to the current work without generating new problems.

2.2 Execution of text to speech

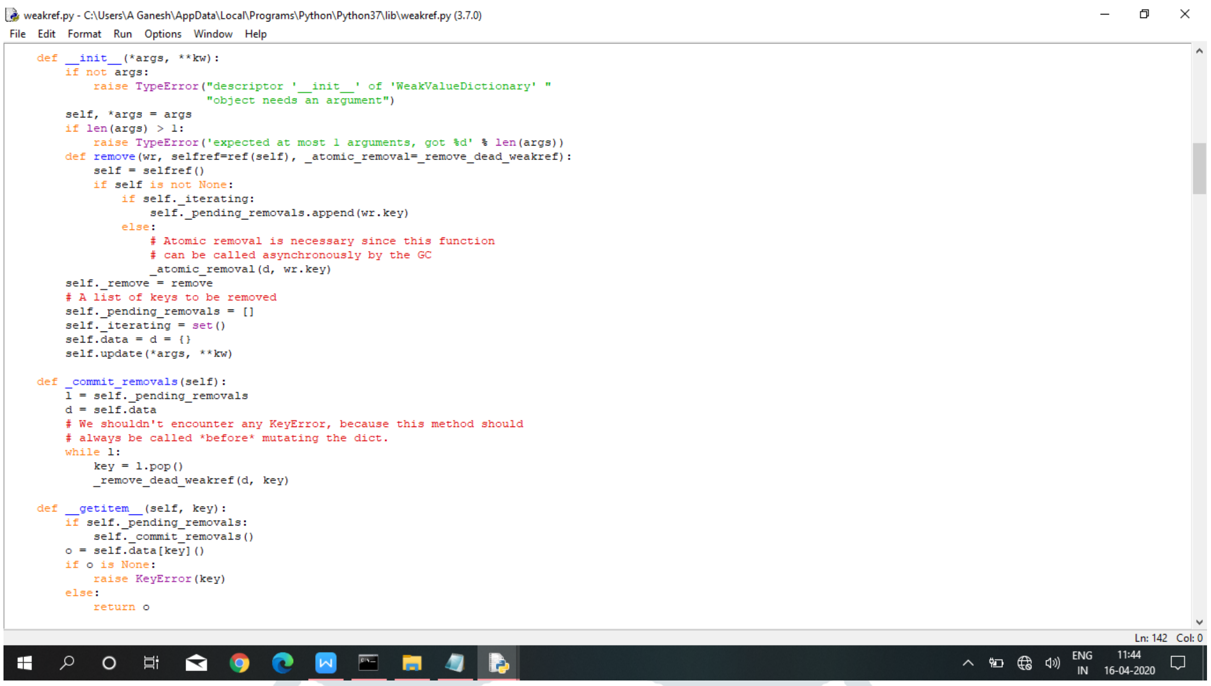
Initially we have to write the required text in a python program which is easily done , using the help of Hidden Markov Model and Pyttsx3 then we have to characterize each letter or word or number or special character from the given text by the breaking the word or letter from space to space region and then this word or a character is stored in a python dictionary and then these values are again retrieved after the end of the text.

Here we aren’t training any personal voice for reading the input text and we are considering the System Engine or inbuilt voices provided by the windows Operating system and with these voices we are able to read the extracted and stored words or characters present in the dictionaries. This system is very accurate as compared to that of Google GTTS Engine and we are able to maintain this accuracy even the input size of the text is very large which is not possible by others. By the end we are capable of reading characters, words, numbers and symbols.



```
# initialisation
engine = pyttsx3.init()

# testing
engine.say("My First Text To Speech Programs. This is the second sentence. Today is 16-04-2020. Other date 16/04/2020. ")
engine.say("The world is, Unique")
engine.say("My roll Number is, #, $, %, @, &, 316126510123")clear
engine.runAndWait()
```

(a)


weakref.py - C:\Users\A Ganesh\AppData\Local\Programs\Python\Python37\lib\weakref.py (3.7.0)

```python
    def __init__(*args, **kw):
        if not args:
            raise TypeError("descriptor '__init__' of 'WeakValueDictionary' "
                            "object needs an argument")
        self, *args = args
        if len(args) > 1:
            raise TypeError('expected at most 1 arguments, got %d' % len(args))
        def remove(wr, selfref=ref(self), _atomic_removal=_remove_dead_weakref):
            self = selfref()
            if self is not None:
                if self._iterating:
                    self._pending_removals.append(wr.key)
                else:
                    # Atomic removal is necessary since this function
                    # can be called asynchronously by the GC
                    _atomic_removal(d, wr.key)
        self._remove = remove
        # A list of keys to be removed
        self._pending_removals = []
        self._iterating = set()
        self.data = d = {}
        self.update(*args, **kw)

    def _commit_removals(self):
        l = self._pending_removals
        d = self.data
        # We shouldn't encounter any KeyError, because this method should
        # always be called *before* mutating the dict.
        while l:
            key = l.pop()
            _remove_dead_weakref(d, key)

    def __getitem__(self, key):
        if self._pending_removals:
            self._commit_removals()
        o = self.data[key]()
        if o is None:
            raise KeyError(key)
        else:
            return o
```

Ln: 142 Col: 0

(b)

Figure 3 : (a) and (b) The above images elucidates the practical elucidation of code to process the execution of text to speech

2.3 Execution of speech to text

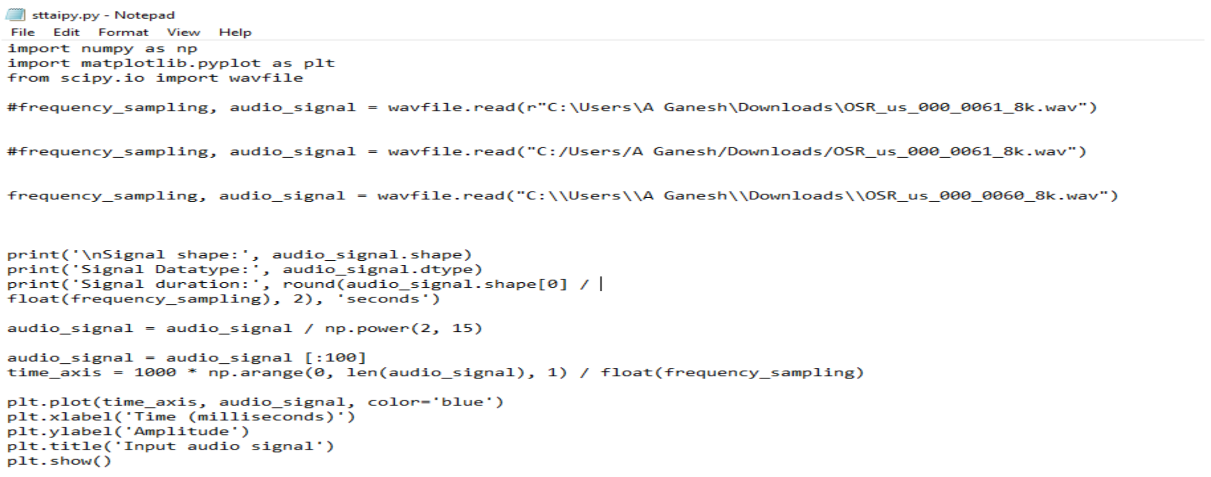
In order to convert the required Speech to Text we have to two methods of providing the input to the system one is by taking any file from a large data set which are in .wav file format and the other one is directly providing the input by the user from the microphone.

This Speech To Text Conversion consists of multiple steps as mentioned in the Module Elucidation and the first step is Visualizing the Audio Signals by parsing the input .wav file or user's voice from microphone and the graph produced depicts the high and low pitch of the input voice and the signal duration, data type.

The second step is Characterizing the Audio Signal where we convert the given voice into frequency domain signals of x and y axis so that this coordinates can be used for further steps in converting the Speech To Text. Further the graph depicts the exact rise and fall of the voice signals for the entire input.

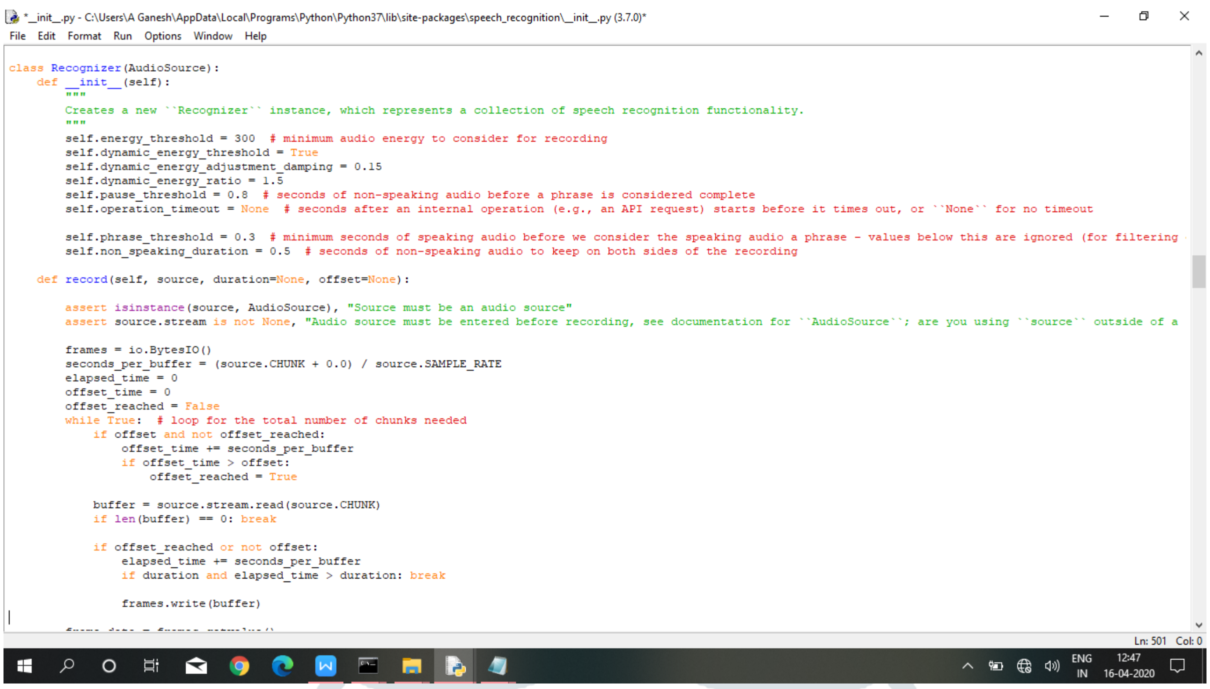
The third step is Recognition of spoken words where we use the help of Hidden Markov Model's MFCC (Mel Frequency Cepstral Coefficient) and thus two graphs are maintained in the output for the given input signals which are MFCC and Filter Bank which also depicts the number of windows or frames formed by the input signal and length of the each window .

The final step is using the recognizer method where the output signals of the frequency domain graph and the MFCC, Filter Bank graphs and using these coordinates a particular frame of voice is converted into the corresponding character or word or number or symbol and that text is stored in some python list within a dictionary. Finally after the entire input signal is processed the corresponding python dictionary containing the lists is organized and the output is generated.


sttaipy.py - Notepad

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import wavfile

#frequency_sampling, audio_signal = wavfile.read(r"C:\Users\A Ganesh\Downloads\OSR_us_000_0061_8k.wav")


#frequency_sampling, audio_signal = wavfile.read("C:/Users/A Ganesh/Downloads/OSR_us_000_0061_8k.wav")


frequency_sampling, audio_signal = wavfile.read("C:\\Users\\A Ganesh\\Downloads\\OSR_us_000_0060_8k.wav")



print('\nSignal shape:', audio_signal.shape)
print('Signal Datatype:', audio_signal.dtype)
print('Signal duration:', round(audio_signal.shape[0] / |
float(frequency_sampling), 2), 'seconds')

audio_signal = audio_signal / np.power(2, 15)

audio_signal = audio_signal [:100]
time_axis = 1000 * np.arange(0, len(audio_signal), 1) / float(frequency_sampling)

plt.plot(time_axis, audio_signal, color='blue')
plt.xlabel('Time (milliseconds)')
plt.ylabel('Amplitude')
plt.title('Input audio signal')
plt.show()
```

(a)


init.py - C:\Users\A Ganesh\AppData\Local\Programs\Python\Python37\lib\site-packages\speech_recognition_init_.py (3.7.0)
File Edit Format Run Options Window Help
Ln: 501 Col: 0

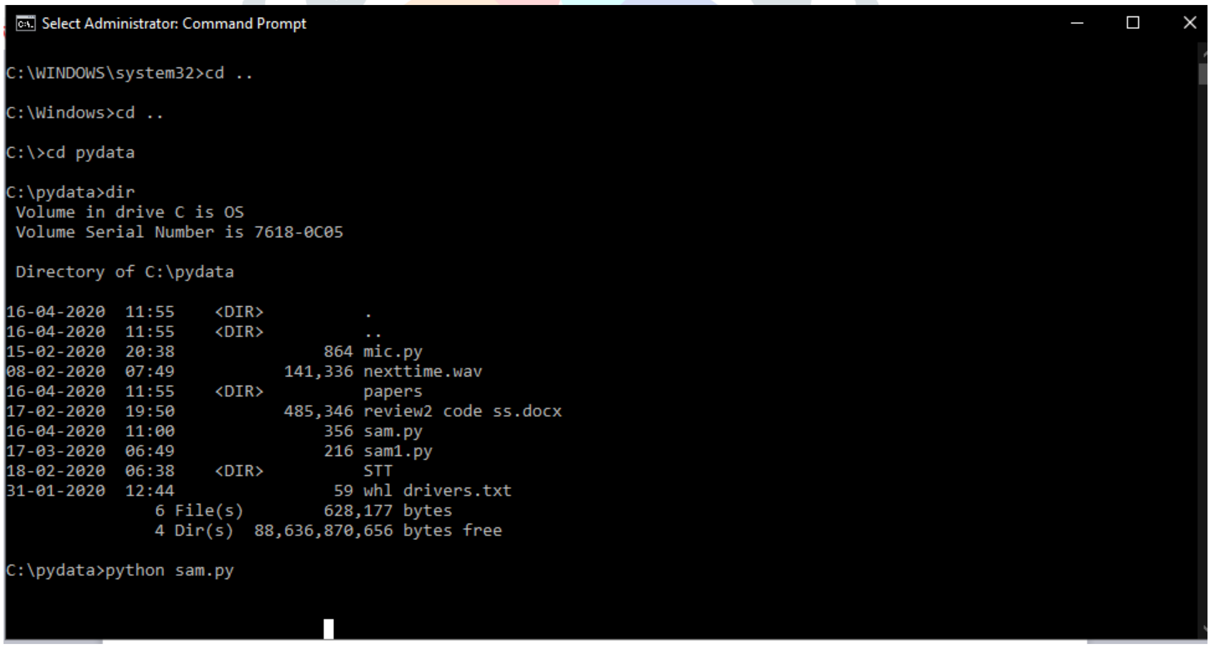
```
class Recognizer(AudioSource):
    def __init__(self):
        """
        Creates a new ``Recognizer`` instance, which represents a collection of speech recognition functionality.
        """
        self.energy_threshold = 300  # minimum audio energy to consider for recording
        self.dynamic_energy_threshold = True
        self.dynamic_energy_adjustment_damping = 0.15
        self.dynamic_energy_ratio = 1.5
        self.pause_threshold = 0.8  # seconds of non-speaking audio before a phrase is considered complete
        self.operation_timeout = None  # seconds after an internal operation (e.g., an API request) starts before it times out, or ``None`` for no timeout

        self.phrase_threshold = 0.3  # minimum seconds of speaking audio before we consider the speaking audio a phrase - values below this are ignored (for filtering
        self.non_speaking_duration = 0.5  # seconds of non-speaking audio to keep on both sides of the recording

    def record(self, source, duration=None, offset=None):

        assert isinstance(source, AudioSource), "Source must be an audio source"
        assert source.stream is not None, "Audio source must be entered before recording, see documentation for ``AudioSource``; are you using ``source`` outside of a

        frames = io.BytesIO()
        seconds_per_buffer = (source.CHUNK + 0.0) / source.SAMPLE_RATE
        elapsed_time = 0
        offset_time = 0
        offset_reached = False
        while True:  # loop for the total number of chunks needed
            if offset and not offset_reached:
                offset_time += seconds_per_buffer
                if offset_time > offset:
                    offset_reached = True

            buffer = source.stream.read(source.CHUNK)
            if len(buffer) == 0: break

            if offset_reached or not offset:
                elapsed_time += seconds_per_buffer
                if duration and elapsed_time > duration: break

                frames.write(buffer)
```

(b)

Figure 4 : (a) and (b) The above images elucidates the practical elucidation of code to process the execution of text to speech

3. RESULTS

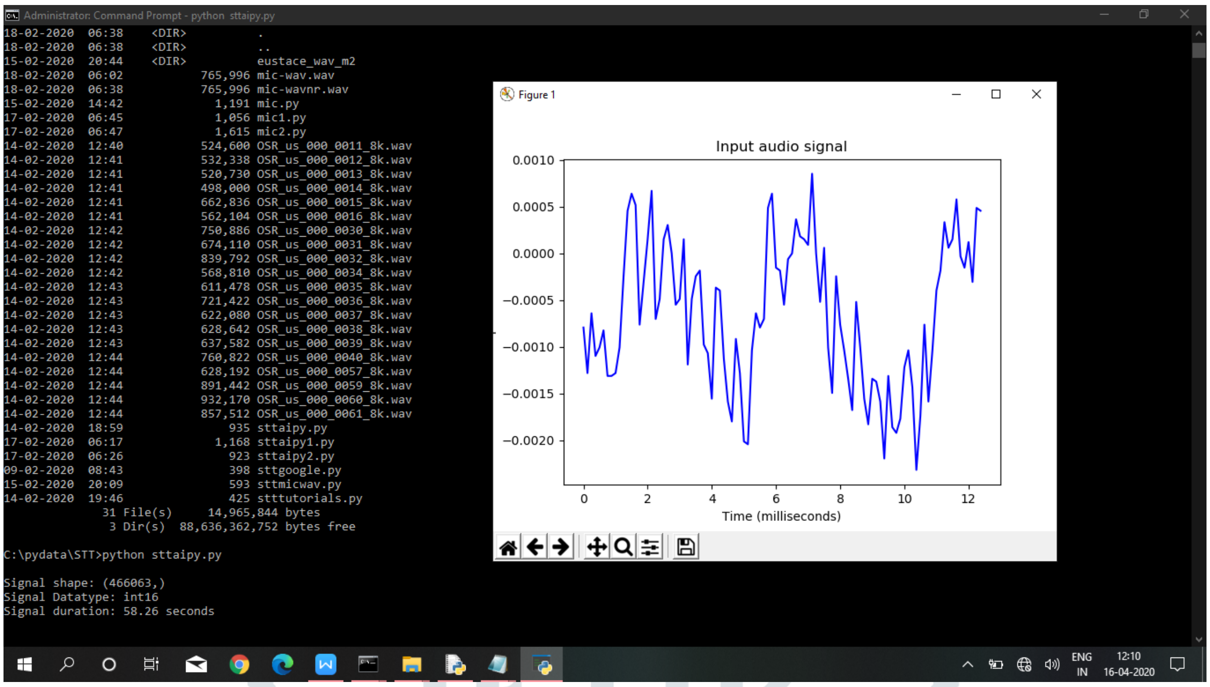
3.1 Execution of text to speech



```
C:\WINDOWS\system32>cd ..

C:\Windows>cd ..

C:\>cd pydata

C:\pydata>dir
 Volume in drive C is OS
 Volume Serial Number is 7618-0C05

 Directory of C:\pydata

16-04-2020  11:55    <DIR>          .
16-04-2020  11:55    <DIR>          ..
15-02-2020  20:38               864 mic.py
08-02-2020  07:49           141,336 nexttime.wav
16-04-2020  11:55    <DIR>          papers
17-02-2020  19:50           485,346 review2 code ss.docx
16-04-2020  11:00               356 sam.py
17-03-2020  06:49               216 sam1.py
18-02-2020  06:38    <DIR>          STT
31-01-2020  12:44                59 whl drivers.txt
               6 File(s)        628,177 bytes
               4 Dir(s)  88,636,870,656 bytes free

C:\pydata>python sam.py
```

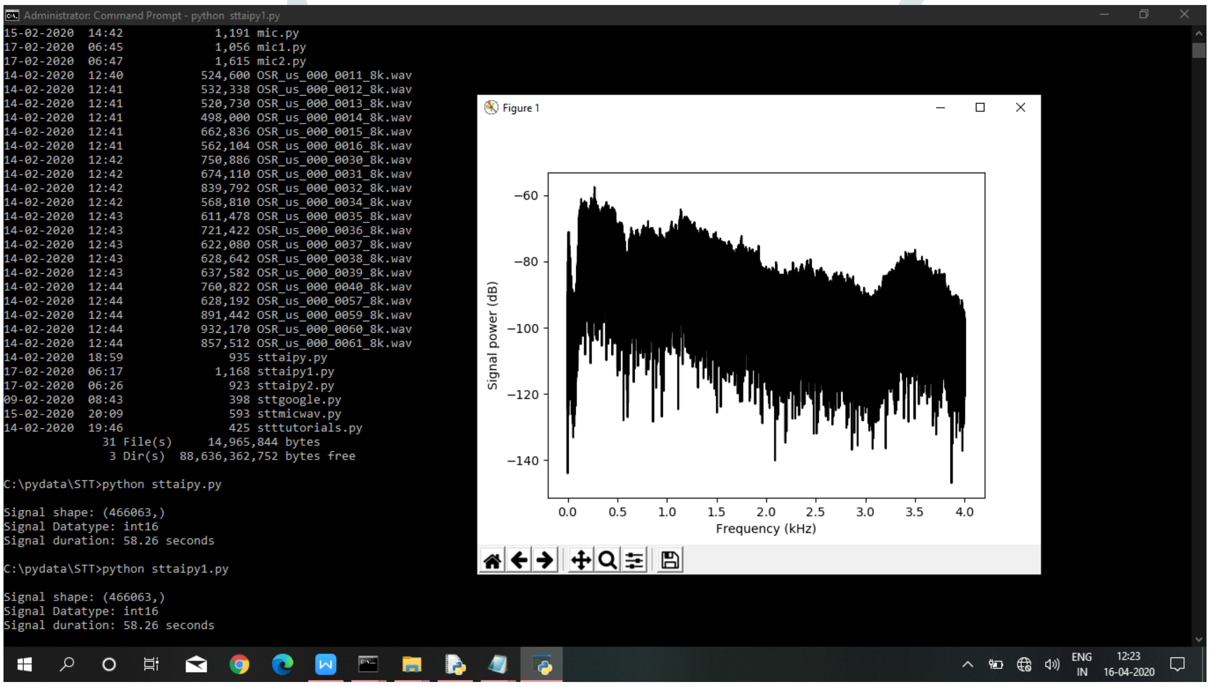
Figure 4 : The above picture decipicts the execution of the text to speech conversion

3.2 Execution of speech to text



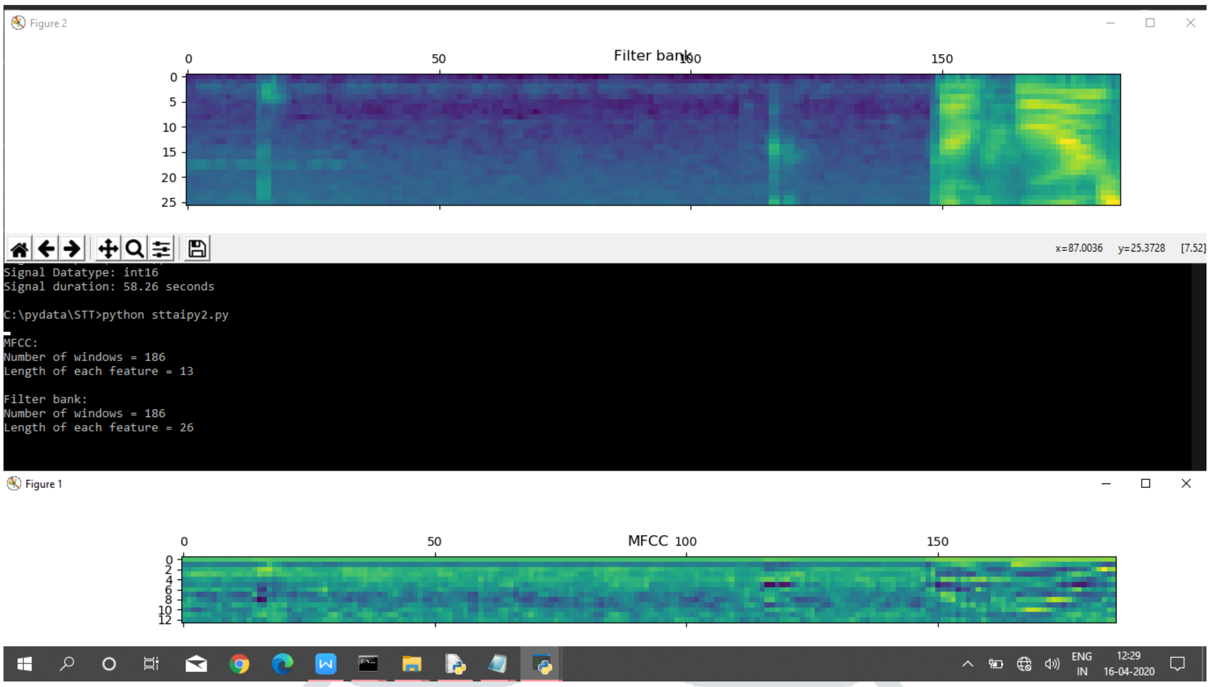
```
18-02-2020  06:38    <DIR>          .
18-02-2020  06:38    <DIR>          ..
15-02-2020  20:44    <DIR>          eustace_wav_m2
18-02-2020  06:02           765,996 mic-wav.wav
18-02-2020  06:38           765,996 mic-wavnr.wav
15-02-2020  14:42             1,191 mic.py
17-02-2020  06:45             1,056 mic1.py
17-02-2020  06:47             1,615 mic2.py
14-02-2020  12:40           524,600 OSR_us_000_0011_8k.wav
14-02-2020  12:41           532,338 OSR_us_000_0012_8k.wav
14-02-2020  12:41           520,730 OSR_us_000_0013_8k.wav
14-02-2020  12:41           498,000 OSR_us_000_0014_8k.wav
14-02-2020  12:41           662,836 OSR_us_000_0015_8k.wav
14-02-2020  12:41           562,104 OSR_us_000_0016_8k.wav
14-02-2020  12:42           750,886 OSR_us_000_0030_8k.wav
14-02-2020  12:42           674,110 OSR_us_000_0031_8k.wav
14-02-2020  12:42           839,792 OSR_us_000_0032_8k.wav
14-02-2020  12:42           568,810 OSR_us_000_0034_8k.wav
14-02-2020  12:43           611,478 OSR_us_000_0035_8k.wav
14-02-2020  12:43           721,422 OSR_us_000_0036_8k.wav
14-02-2020  12:43           622,080 OSR_us_000_0037_8k.wav
14-02-2020  12:43           628,642 OSR_us_000_0038_8k.wav
14-02-2020  12:43           637,582 OSR_us_000_0039_8k.wav
14-02-2020  12:44           760,822 OSR_us_000_0040_8k.wav
14-02-2020  12:44           628,192 OSR_us_000_0057_8k.wav
14-02-2020  12:44           891,442 OSR_us_000_0059_8k.wav
14-02-2020  12:44           932,170 OSR_us_000_0060_8k.wav
14-02-2020  12:44           857,512 OSR_us_000_0061_8k.wav
14-02-2020  18:59               935 sttaipy.py
17-02-2020  06:17             1,168 sttaipy1.py
17-02-2020  06:26               923 sttaipy2.py
09-02-2020  08:43               398 sttgoogle.py
15-02-2020  20:09               593 sttmicwav.py
14-02-2020  19:46               425 stttutorials.py
              31 File(s)     14,965,844 bytes
               3 Dir(s)  88,636,362,752 bytes free

C:\pydata\STT>python sttaipy.py

Signal shape: (466063,)
Signal Datatype: int16
Signal duration: 58.26 seconds
```

(a)



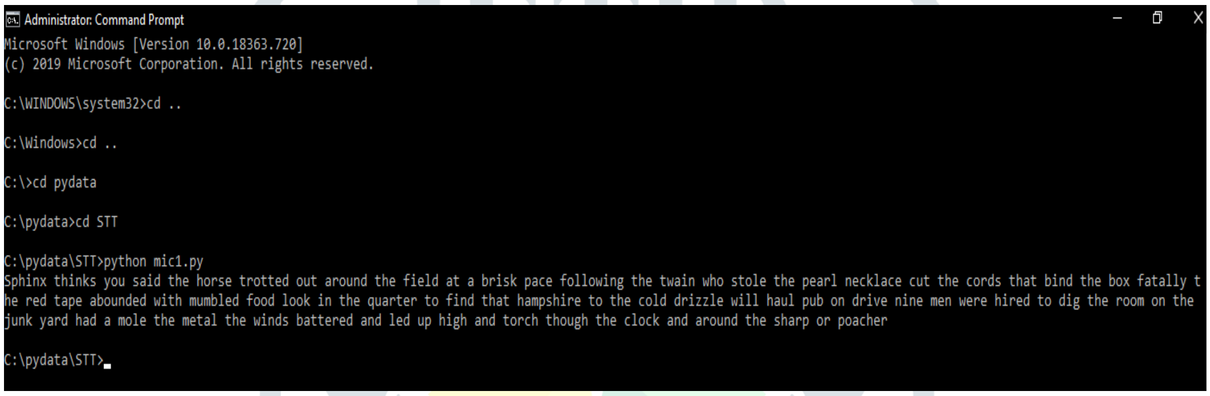
```
15-02-2020  14:42             1,191 mic.py
17-02-2020  06:45             1,056 mic1.py
17-02-2020  06:47             1,615 mic2.py
14-02-2020  12:40           524,600 OSR_us_000_0011_8k.wav
14-02-2020  12:41           532,338 OSR_us_000_0012_8k.wav
14-02-2020  12:41           520,730 OSR_us_000_0013_8k.wav
14-02-2020  12:41           498,000 OSR_us_000_0014_8k.wav
14-02-2020  12:41           662,836 OSR_us_000_0015_8k.wav
14-02-2020  12:41           562,104 OSR_us_000_0016_8k.wav
14-02-2020  12:42           750,886 OSR_us_000_0030_8k.wav
14-02-2020  12:42           674,110 OSR_us_000_0031_8k.wav
14-02-2020  12:42           839,792 OSR_us_000_0032_8k.wav
14-02-2020  12:42           568,810 OSR_us_000_0034_8k.wav
14-02-2020  12:43           611,478 OSR_us_000_0035_8k.wav
14-02-2020  12:43           721,422 OSR_us_000_0036_8k.wav
14-02-2020  12:43           622,080 OSR_us_000_0037_8k.wav
14-02-2020  12:43           628,642 OSR_us_000_0038_8k.wav
14-02-2020  12:43           637,582 OSR_us_000_0039_8k.wav
14-02-2020  12:44           760,822 OSR_us_000_0040_8k.wav
14-02-2020  12:44           628,192 OSR_us_000_0057_8k.wav
14-02-2020  12:44           891,442 OSR_us_000_0059_8k.wav
14-02-2020  12:44           932,170 OSR_us_000_0060_8k.wav
14-02-2020  12:44           857,512 OSR_us_000_0061_8k.wav
14-02-2020  18:59               935 sttaipy.py
17-02-2020  06:17             1,168 sttaipy1.py
17-02-2020  06:26               923 sttaipy2.py
09-02-2020  08:43               398 sttgoogle.py
15-02-2020  20:09               593 sttmicwav.py
14-02-2020  19:46               425 stttutorials.py
              31 File(s)     14,965,844 bytes
               3 Dir(s)  88,636,362,752 bytes free

C:\pydata\STT>python sttaipy.py

Signal shape: (466063,)
Signal Datatype: int16
Signal duration: 58.26 seconds

C:\pydata\STT>python sttaipy1.py

Signal shape: (466063,)
Signal Datatype: int16
Signal duration: 58.26 seconds
```

(b)



```
Signal Datatype: int16
Signal duration: 58.26 seconds

C:\pydata\STT>python sttaipy2.py

MFCC:
Number of windows = 186
Length of each feature = 13

Filter bank:
Number of windows = 186
Length of each feature = 26
```

(c)



```
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd ..

C:\Windows>cd ..

C:\>cd pydata

C:\pydata>cd STT

C:\pydata\STT>python mic1.py
Sphinx thinks you said the horse trotted out around the field at a brisk pace following the twain who stole the pearl necklace cut the cords that bind the box fatally t
he red tape abounded with mumbled food look in the quarter to find that hampshire to the cold drizzle will haul pub on drive nine men were hired to dig the room on the
junk yard had a mole the metal the winds battered and led up high and torch though the clock and around the sharp or poacher

C:\pydata\STT>
```

(d)

Figure 4 : (a) ,(b) ,(c) and (d) The above images elucidates the practical elucidation of the execution of speech to text.

(a) is Visualizing the Audio Signals by parsing the input .wav file or user's voice from microphone and the graph produced depicts the high and low pitch of the input voice and the signal duration, data type.

(b) Characterizing the Audio Signal, converting the given voice into frequency domain signals of x and y axis. The graph depicts the exact rise and fall of the voice signals for the entire input.

(c) The later is Recognition of spoken words where we use the help of Hidden Markov Model's MFCC and thus two graphs are maintained in the output for the given input signals which are MFCC and Filter Bank which also depicts the number of windows or frames formed by the input signal and length of the each window .

(d) Using the recognizer method where the output signals of the frequency domain graph and the MFCC, Filter Bank graphs and using these coordinates a particular frame of voice is converted into the corresponding character or word or number or symbol and that text is printed out on the output.

4. CONCLUSION

We have introduced a flavour of python technology to embed an element of speech processing by text to speech and speech to text conversions. We presented an accurate conversion of speech to text and text to speech with many embedded system features and the use of python technology. The added essence to this work is its ability to convert by seeking the input from live speech or from any audio file. The use of the Hidden Markov Model and the Mel Frequency cepstral coefficient has yielded a way more precisive result.

We believe that our work opens additional opportunities for inventively and more accurately grasping the different slangs of different intonated voices altering and joining the sound signals to produce a way more accurate and fast conversions to help the disabled to be more dynamic and lead a normal life as everybody. The immediate future work may include the enhancement of this project by making it able to feed on different file formats like VLC media player or mp3 media player etc. Different roads of future work incorporate fast feed-forward system approximations of various slangs and conversions.

5. ACKNOWLEDGMENT

The project team members would like to express our thanks to our guide Asst. Prof N. Lokeswari , Dept. of Computer Science and Engineering, ANITS for her valuable suggestions and guidance in completing our research work. We also thank the other teaching staff in the department for their great efforts and for giving us the advice that has helped us to achieve our project's goals.

REFERENCES

[1] Kumbharana, Chandresh K., 2007, *"Speech Pattern Recognition for Speech to Text Conversion",* thesis PhD, Saurashtra University

[2] A study of different issues involved in software based approach to voice/speech synthesis and recognition with investigation regarding pattern recogntion- A Ph.D. thesis bu Dr. GR Kulkarani.

[3] http://tdil.meity.gov.in/Research_Effort.aspx

[4] http://tdil.meity.gov.in/Publications/Vishwabharatnew.aspx

[5] Speech Py-A Library for Speech Processing and Recognition Amirsina Torfi arXiv:1803.01094v3 [cs.SD] 25 May 2018

[6] A REVIEW ON SPEECH TO TEXT CONVERSION METHODS Miss.Prachi Khilari1 1Department of E&TC Engineering. G.H.R.C.O.E.M, Ahmednagar. Savitribai Phule University of Pune. Prof. Bhope V. P.2 2Department of E&TC Engineering. G.H.R.C.O.E.M, Ahmednagar. Savitribai Phule University of Pune. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 4 Issue 7, July 2015

[7] IJCSI International Journal of Computer Science Issues, Vol. 11, Issue 5, No 2, September 2014 ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784 www.IJCSI.org Machine Translation Approaches: Issues and Challenges M. D. Okpor Department of Computer Science, Delta State Polytechnic, Ozoro Delta State, Nigeria

[8] International Refereed Journal of Engineering and Science (IRJES) ISSN (Online) 2319-183X, (Print) 2319-1821 Volume 3, Issue 1 (January 2014), PP. 39-41 www.irjes.com Novel Model for Speech to Text Conversion Deepa V.Jose, Alfateh Mustafa, Sharan R Department of Computer Science, Christ University, Hosur Road, Bangalore-23

[9] International Journal of Innovation and Scientific Research ISSN 2351-8014 Vol. 17 No. 2 Aug. 2015, pp. 271-277 © 2015 Innovative Space of Scientific Research Journals http://www.ijisr.issr-journals.org/ Corresponding Author: Md. Safaet Hossain Speech to Text Conversion in Real-time Nuzhat Atiqua Nafis1 and Md. Safaet Hossain Department of Electronic and Telecommunication Engineering, University of Development Alternative, Dhaka, Bangladesh Department of Computer Science and Engineering, University of Development Alternative, Dhaka, Bangladesh

[10] INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH VOLUME 4, ISSUE 06, JUNE 2015 ISSN 2277-8616 349 IJSTR©2015 www.ijstr.org Speech-To-Text Conversion (STT) System Using Hidden Markov Model (HMM) Su Myat Mon, Hla Myo Tun

[11] https://www.sciencedirect.com/science/article/pii/S2352484717300616

[12] http://www.microsoft.com/typography/AboutFontsOverview.mspx

[13] https://it.toolbox.com/articles/what-is-natural-language-processing-definition-applicati ons-techniques-and-tools

[14] A study of different issues involved in software based approach to voice/speech synthesis and recognition with investigation regarding pattern recogntion- A Ph.D. thesis bu Dr. GR Kulkarani.

[15] Project on "device control using speech recognition" by Arora Arti H., Langhnoja Namrata V., Marvaniya Komal N. of VVP Engineering college, Rajkot

[16] Auditory toolbox (Version – 2) Malcolm Slaney (Technical report #1998-010 Interval Research Corporation)