**SPEECH TO TEXT CONVERSION**

**By**

**RUHI H PATEL (17BIT097)**

**HET UTPAL SHAH(17BIT103)**

******

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**Ahmedabad 382481**

**SPEECH TO TEXT CONVERSION**

**Mini Project - I**

Submitted in fulfillment of the requirements

For the degree of

**Bachelor of Technology in Information Technology**

By

**RUHI H PATEL (17BIT097)**

**HET UTPAL SHAH(17BIT103)**

Guided By

**PROF. PRONOYA BHATTACHARYA**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

******

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**Ahmedabad 382481**

**CERTIFICATE**

This is to certify that the project entitled “SPEECH TO TEXT CONVERSION” submitted by RUHI H PATEL and HET UTPAL SHAH , towards the partial fulfillment of the requirements for the degree of Bachelor of Technology in Information Technology of Nirma University is the record of work carried out by him/her under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination.

Prof Pronoya Bhattacharya Dr. Madhuri Bhavsar

Assistant professor HOD, Dept. of Information Technology

Department of Information Technology, Institute of Technology,

Institute of Technology, Nirma University,

Nirma University, Ahmedabad

Ahmedabad

**ACKNOWLEDGEMENT**

We would like to express my deepest thanks to all of our professors ,friends and family who have given me vital help and direction towards the successful completion of my work, We would also like to express special gratitude to our mini project guide Prof. Pronoya bhattacharya for giving important guidance and insight from his exhaustive experience and expertise .

**ABSTRACT/ Outline**

In this project we basically are trying to explore the field speech to text conversion , how the process works , the different models that are being used by the industry today and also to implement it via the different APIs offered by various organisations

**CONTENTS**

Certificate

Acknowledgement

Abstract

Table of Contents

List of figures

Fig 1.1 the timeline of models

Fig 2.1 flowchart of ASR

Fig 3.1 The above figure shows the dictionary of pocketsphinx

Fig 3.2 code snippet for pocketsphinx

Fig 3.3 WER calculation

Fig 3.4 code snippet of jiwer

Fig 3.5 Output of Pocket sphinx and its accuracy

Fig 3.6 word error rate found

Fig 3.7 SRT file

List of tables

Table 2.1 comparison of feature extraction techniques

Table 2.2Comparison of Speech Models

Table 2.3 Comparison of the APIs

**Chapter 1 Introduction 1**

1.1 General

**Chapter 2 Literature survey 2**

* 1. General 2
  2. Comparative analysis 3

2.2.1 On basis of feature extraction techniques 3

2.2.2 On basis of the models 4

2.2.3 On the basis of APIs 6

**Chapter 3 Project features 7**

3.1 Why python? 7

3.2 Libraries used 7

3.2.1 PocketSphinx 8

3.2.2 Google API 10

3.3 Inputs considered 12 3.4 Testing mechanism 13

3.5 How was Conversion done? 15

3.6 Results 16

**Chapter 4 Application 17**

4.1 Subtitle Generation 17

**Chapter F Summary and Conclusion 19**

F.1 Summary 20

F.2 Conclusions 20

**References**

**Appendix -A**

**1 INTRODUCTION**

**1.1 General**

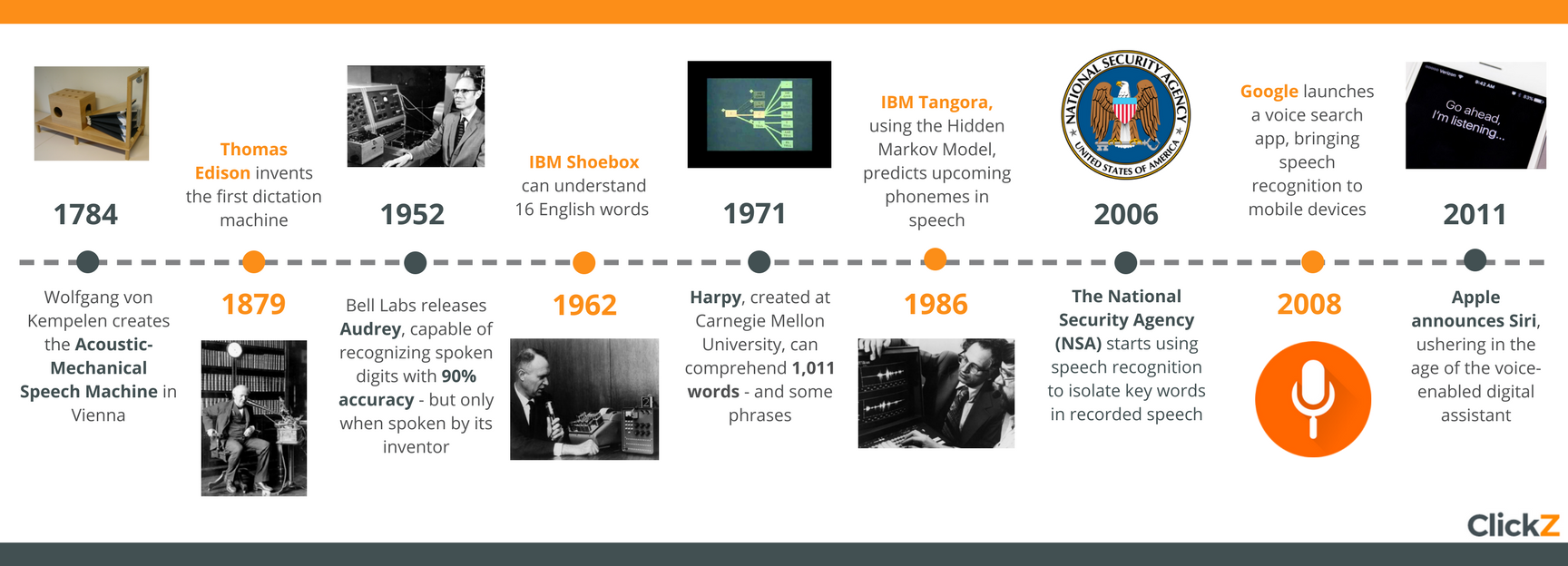
If we start with the basic question what is speech then the the answer comes as it is basically a form in which humans interact or talk with one another by using their vocal cords and certain combinations of words .

So how does one language differ from another the answer lies in the phonetic permutations and combinations used by different languages which is there Peculiar characteristic The ways to demarcate a speech are as follows:-

1. there can be no speech that is just the silence
2. there can be speech where the trepidity of the voice chord in non-existent .
3. and finally there is a speech where the vibrations are periodic in nature and forms a uniform wave form

Speech recognition is that the ability of a machine or program to spot the words or phrases in speech communication and convert them to a computer code format. However , terribly basic speech to text applications might use solely restricted vocabulary and should be ready to discover the words of they're clearly spoken.Speech to text conversion is one in all the applications of speech recognition.Speech recognition may be a big leap towards automation , from higher driving expertise to directory help to aiding individuals with disabilities. It helps towards building a lot of refined technologies .

The timeline of the speech to text techniques is given in the Fig 1.1 as follows , the pioneer of this technology was Prof Reddy of carnegie mellon

university   


*Fig 1.1 the timeline of models*

**2 Literature review**

**2.1 General**

Speech can be converted to text using two models

1. Word model -vocabulary is small and the words are modeled on their own.

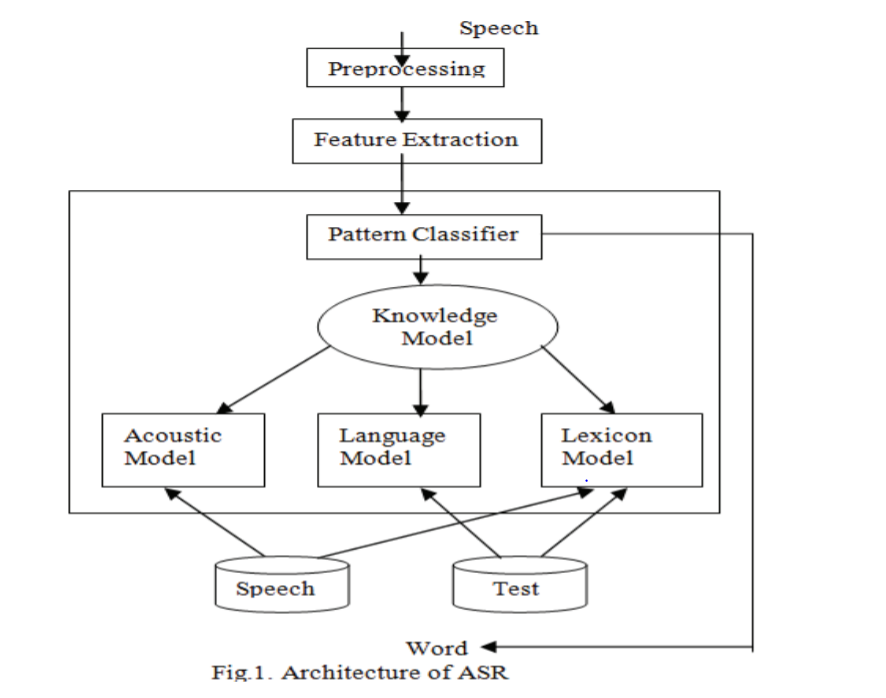
2. Phone model - based on parts of words

Phone model has 2 subparts

1. Context independent - only the subparts are considered

2. Context dependent - subparts along with their neighbors are considered

the automation of speech recognition can take place as the method described in the Fig2.1



*Fig 2.1 flowchart of ASR*

**2.1 Comparative analysis**

Traditionally speech to text was done using models such as Hidden Markov model (HMM) and Guassian Mixture Model(GMM).HMM is a flexible model since it allows addition of unobserved states. But more flexibility doesn’t mean that it will fit the data well and will give accurate results. GMM is used when you have some hidden (non-observable) parameters. In order to overcome all the shortcoming a new model was created known as NN-GMM-DMM. But we first need to study the feature extraction methods.

**2.1.1 On basis of Feature extraction techniques**

After researching many paper we have been able to create a comprehensive review of the different feature extraction techniques being used for this purpose and the how do they compare against one another

and pitch their advantages and disadvantages in Table 2.1, this provides the people a clear knowledge of how the different models work are which is better than other and in what way

*Table 2.1 comparison of feature extraction techniques[1]*

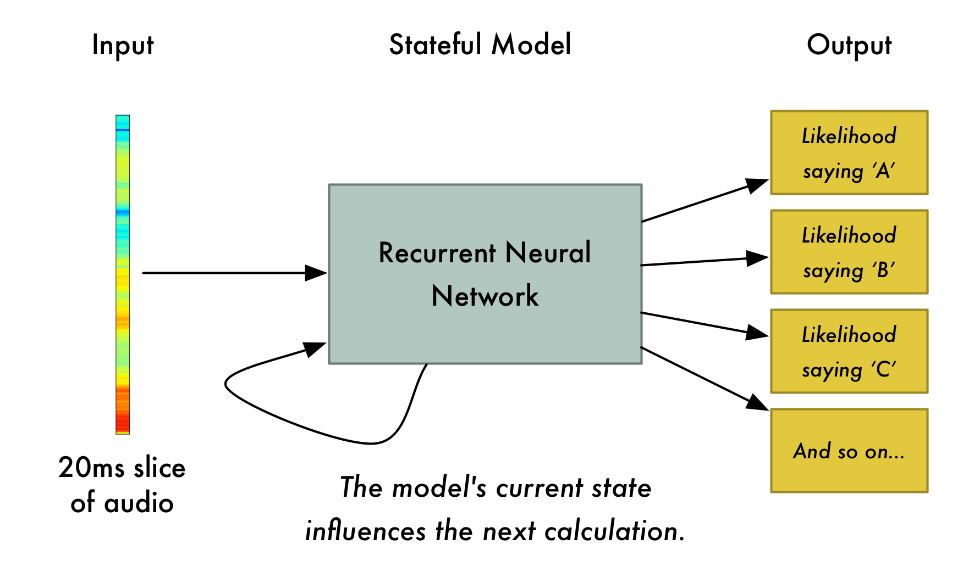
|  |  |  |  |
| --- | --- | --- | --- |
| SR no | Feature Extraction | Advantages | Disadvantages |
| 1 | Linear Predictive coding | It is useful for modelling vowels which are periodic but not for nasalized vowels It is accurate and also the computation time is also good.It is mostly used for encoding at low bit rate | However it cannot distinguish vowels with similar sounds. |
| 2 | Mel-frequency cepstrum (MFFCs) | MFFC tries to mimic the working of the human ear by using a series of mathematical Formulas and functions MFFC is not very complex to implement and also it has a very high accuracy | MFFC doesn't work well when there is background noise Also performance may be affected by the number of filters used. |
| 3 | RASTA filtering | Unlike MFFC it can work in noisy environment It doesn't depend on the choice of microphone nor does it depend upon the distance of mouth relative to the microphone. It can capture low modulation frequencies | It requires silence to detect each utterance of a word |
| 4 | Principal component Analysis(PCA) | It is a used to reduce dimensionality of data while keeping the variance same | it depends upon the scaling of data and there is no way to know what way is best to scale the data to obtain optimum results . |

**2.1.2 On the basis of the models[4][5]**

The main issue with normal methods is that it cannot distinguish hello from Heelllooo. Hence deep learning is used**.**

But we cannot directly feed the sound waves to the neural network. Sound waves can be expressed in one dimension and at each particular time it has an amplitude associated with it. To make it easier for the computer to understand we convert the speech to numbers. This can be easily done by noting the height of the wave at equally spaced points. This process is known as sampling . For human voice sampling rate of 16khz is enough to cover all the frequency

Even after sampling the sound samples may contain different kinds of sounds and to make it easier for the neural network to understand we break the sample into the its component parts by a mathematical process called Fourier Transform.



*Fig 2.2 how models do prediction*

We feed the small slices of audio sample to the RNN. For each audio slice it will try to recognize the letter corresponding to the sound .In case of RNN it has a memory that influences the future predictions. If the RNN has interpreted the sound as HEL till now then the mostly likely next two letters will be LO and not PQR. So the memory of the neural network helps in making more accurate predictions

The accuracy of such deep learning models depends on the training set used and it increases over time as it is tested more and more. Currently Deep learning models are being used with accuracies up to 97%.

In the following table we present the different models being used for the process of speech recognition and list all of their advantage and disadvantages.

*Table 2.2**Comparison of Speech Models[2][3]*

|  |  |  |  |
| --- | --- | --- | --- |
| SR No | Model Name | Advantages | Disadvantages |
| 1 | Hidden Markov model(HMM) | • HMM is a mathematical model and it can be conceptually used in a many fields  • HMM model if implemented properly can be used for a vast number of applications | • Trial and error method for choosing a model topology. • HMM requires setting up of large number of parameters • HMM require vast amount of data to be trained • The assumption that successive observations are independent. • The Markov assumption itself |
| 2 | Guassian Mixture Model(GMM) | • The GMM algorithm is a good algorithm to use for the classification of static parameters • GMMs are well known for their ability to represent arbitrarily complex distributions into clusters.  •GMM based models are used in speech recognition applications(e.g de noising) | •GMM can't work with problems of high dimensionality. •Isn't suitable for temporal patterns. •the number of mixture models to be set up for the algorithm to fit the training dataset |
| 3 | Long short-term memory(LSTM) RNN | As we increase the number of hidden layers the error decreases and the accuracy increases , also they are immune to gap lengths in word sequence as compared to HMMs. | Although as compared to deep Feed forward networks, performance of RNN is dull. |

**2.2.3 On the basis of the API**

The APIs have been compared in this segment and while doing so many intricacies were taken into consideration such as the open sourcing of those apis , user friendliness , their performance against a common set of inputs , their robustness, the added benefits of recognizing commas, full stops and other punctuation marks.

The following table clearly concludes our remarks on these various APIs and the programmer can choose accordingly which one to use .

With a mean word error rate of 16 PF, Google Speech (Video) is that the most correct ASR engine in our testing. for several audio samples, Google’s engine scored a word error rate well underneath 10% — as low as two for a few high-quality audiobook samples.

*Table 2.3 Comparison of the APIs*

|  |  |  |  |
| --- | --- | --- | --- |
| Sr No | API | Advantages | Disadvantage |
| 1 | Google Cloud Speech API | • Support more than 110 languages • Can work in real time • Stable against side noises | •Only 60 minutes free per user After 60 minutes (0.006 USD per 15 seconds) |
| 2 | IBM Watson | • provide customization for particular acoustic condition | • few language supported and even fewer supported for the customization of acoustic model |
| 3 | Amazon Transcribe | • Automatically adds punctuation marks and does text formatting • Best feature is that it supports telephony audio • Adds timestamp for each word | •The feature which labels different  features is not accurate as of now |

**3 Project features**

**3.1 Why python?[6]**

In the implementation of this project we've got used the Python language because it clearly outperforms several alternative language in terms of the options offered and also the Brobdingnagian and intensive set of libraries it contains .python has varied applications in multitude of industries like that of gambling, net frameworks and applications, graphics , vogue et cetera . This provides us with various benefits like :-

Exhaustive number of libraries

many commonplace libraries that are extremely useful for various objectives like internet service tooling, interfaces and different protocols various character array operations or like in our case models of networks and word error rate calculations.

Merging qualities

Python integrates the Enterprise Application Integration that makes it straightforward to develop internet services by invoking COM or ophidian components. it's powerful management capabilities as a result of it calls directly through C, C++ or Java via Jython. Python collectively processes XML and completely different markup languages as a result of it'll run on all trendy operational systems through same storage unit code.

programmer friendliness

the language provide clean and crisp meanings and is extremely easy to learn since it closer to natural language and thus the programmer is saved from deliberating over complex syntaxes like those in other traditional languages .

**3.2 Libraries used**

We used two types of APIs- offline and online. Generally offline APIs use traditionals conversion methods such as HMM and GMM and tend to have

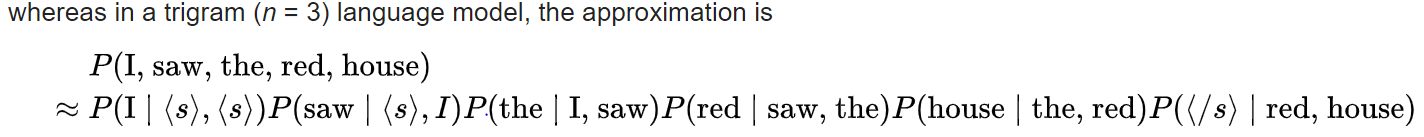
a lower accuracy as compared to online APIs. Online APIs on the other end tend to use deep learning algorithms. Using offline APIs has an inherent disadvantage since it needs to store in some cases the dictionary and which would occupy a lot of a space.Online APIs on the other hand always require a good internet connection and the response/output time will be proportional to the server load and sometimes the online APIs might require a lot of time to display the output.

**3.2.1 PocketSphinx**

PocketSphinx is an example of offline text to speech conversion API. It is a tri-gram statistical language model which means n=3.

Statistical language model is based on probability distribution over a sequence of words. PocketSphink is a tri-gram model which means that the model will take into consideration the previous 3 words which predicting the probability of a word**.**

The probability of a sentence is calculated is as follows



PocketSphinx has a dictionary has a total of 134,723 commonly used words which are listed along with their pronunciation. This dictionary is used to calculate the probability of a particular word based on the occurrence of the previous 3 words.

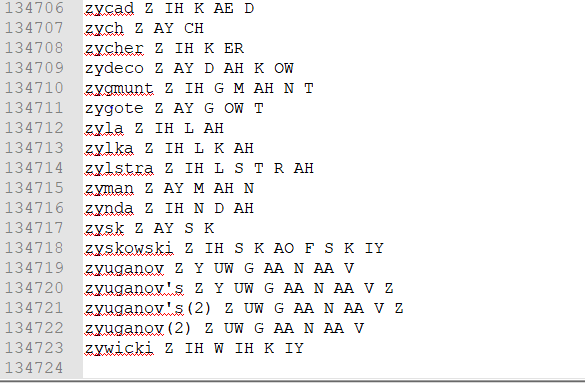
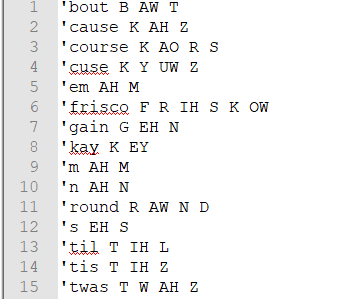


Fig 3.1The above figure shows the dictionary of pocketsphinx



Fig 3.2code snippet for pocketsphinx

As we can see that PocketSphinx gives us an option of customising the language model. It has the following parameters and by use trial and error we arrived at the best value for each parameters.

* **verbose** - is used to show the input file names (we kept it false)
* **hmm** - contains the acoustic model files (we set the path to english and USA)
* **lm** - contains the tri-gram language model files (again we set it to english and USA)
* **dict** - contains the dictionary files (we used the standard CMU dictionary)

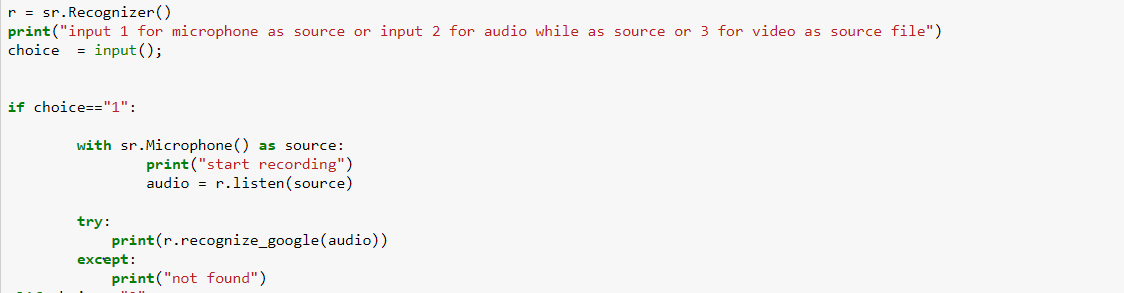
Pocketsphinx works best when the input audio form is in form of .wav format, so we convert the audio in the wav format .

**3.2.1 Google API**

Google API is an online based speech to text converter developed by google Corperation.It offers four machine learning models each suited well for different environment and uses.They are as follows -

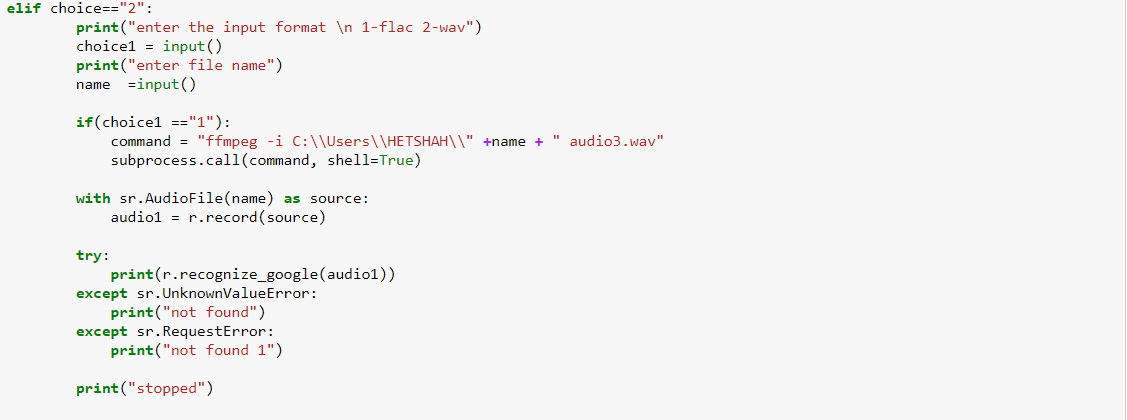
1. For voice commands
2. For phone calls
3. For processing audio from video
4. For all scenarios

We preferred the fourth option and we always provided the input in the form of .wav format.



We provide choice for the user to select in which format the user wants to input the data.

As a part of 1st choice we offered the user with the ability to speak from the microphone and gets its synthesis printed directly.



As a part of 2nd choice we offered the user to input in form of audio formats and we further asked in which format the user is going to input eg.if the user selected 1 then the user is going to enter in flac format.



As a part of the third option the user can input the video files in the desired format( mp4,flv,avi) and the user gets the SRT file as output which then can be used while playing the video to get the transcript for better understanding.

**3.3 Inputs considered**

1. We are basically considering the speech to text conversion from following inputs:
   1. From microphone:In this conversion we can activate the microphone input from the code and the speaker can give his/her input, it gets processed by the Google Cloud Speech Recognition API,included in the speech recognition library in python.

b)From audiofile: In this conversion we took audio files of .mp3, .flac etc format and converted them into .wav format for their synthesis .

c)From video File: In this conversion we took audio files of .mp4,.flv,.avi etc format and converted them into .wav format for their synthesis .

It clearly evident from this that an attempt to generate the subtitles from the given input video file was made apart from the speech to text conversion which took place by inputting a general audio file in the form .wav format . To process a lot of audio a loop was made to run through all the input audio files , each clip recording was not over 2 or 3 seconds since they were extremely high quality recordings and needed to be not over that allotted time therefore to just generate two to three lines of text many audio files had to be looped over

DATASET USED:-

We took data set of LIBRISPEECH and from it we took 10 audio files in .flac format and we tested it for google API and PocketSphinx We were provided with the transcript of the audio files from the data set itself and we used a python library called JIWER to count the error rate PocketSphinx had an error rate of about 55 percent and google api had an error rate of around 29 percent Hence we used google API for our project.

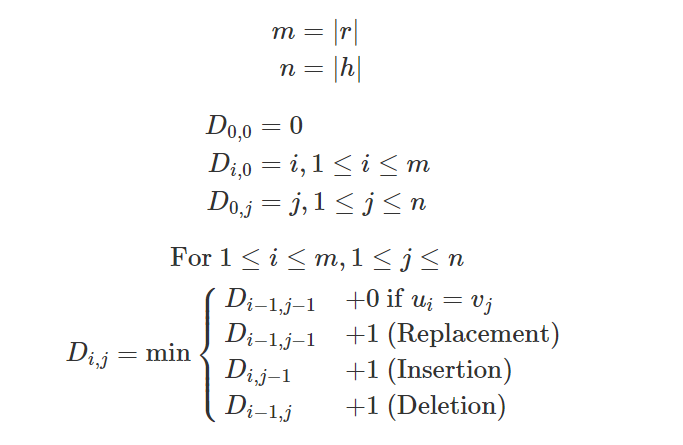
The reason behind choosing the librispeech corpus as our data set is the free availability of the dataset apart from being extremely compatible with the coding environment also they provide distinctly the training sets and testing sets which are essential parts of training any models , it also provides high quality audio clips .

**3.4 Testing mechanism**

Whenever testing of ground truth statements against the output of our models is taken into consideration while dealing with words , the word error rate becomes an important concept in testing the validity and firmness of our model and whether or not tweaks have to be made to improve the general predictions of the audio files the . Mathematically the word error rate can be calculated as the equation 3.1

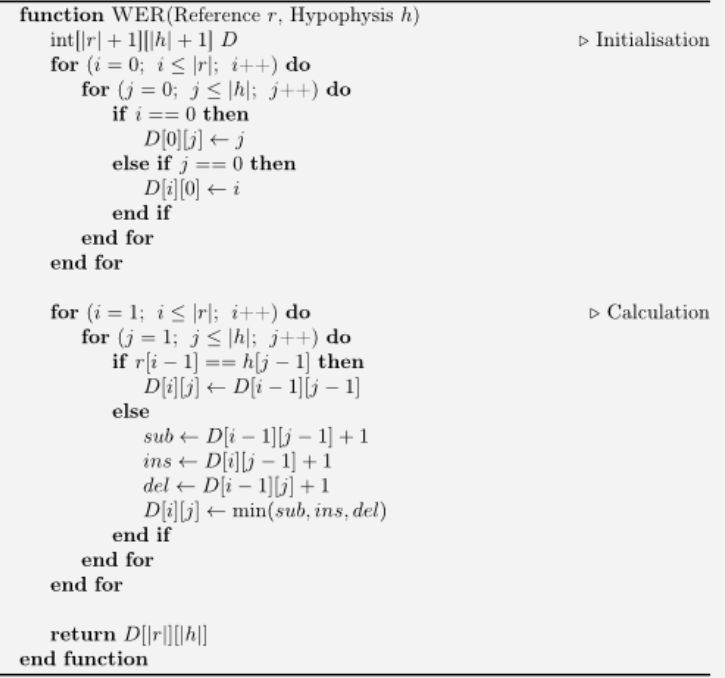
…(3.1)

In this equation the variable S stand for the number of replacements made , the variable D stands for the number of cancellations made and the variable I stand for the number of inputs made and N are the number of words being referred to the ground truth document . word error rate can also be considered analogous to the distance of words paradigm



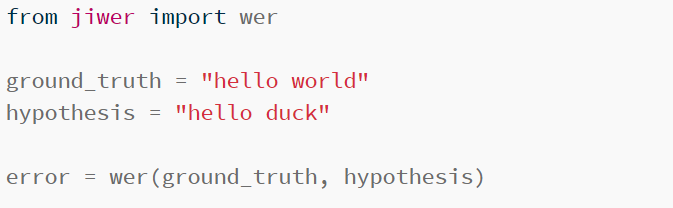
*Fig 3.4 WER calculation*

the pseudo code can be given as follows :



*Fig 3.5 WER algorithm*

Fortunately there is a jiwer library in python that can calculate the word error rate for us by just inputting the absolute truth statement from the transcript and the output of our model and the output of the code can show the wer , which coincidentally tells us about the accuracy of our model



*Fig3.3 code snippet of jiwer*

the ouput of this gives error of about 50%.

**3.5 How conversion was done?**

To convert from one from to another for example mp3 to wav or mp4 to wav we used FFMPEG.FFMPEG is a open source toolbox for audio/video manipulation. FFMPEG works directly from the command line and the command is of the form



The input file is given in the required format For the flags/actions part we define the parameters.For audio to audio conversion we don't need to define any parameters but for video to audio conversion we need to define a few parameters.



As we can we to convert from flac to wav format we don't need any for the actions/flags part



For converting it from mp4 to wav format we needed to define a few parameters such as

* **-vn** it tells us that there is no video in the output
* **-ac** is used to set the number of audio channels
* **-ar** it is used to set the audio frequency sampling
* **-ab**  it is used to set the bit rate

We use the python command called subprocess to open the command prompt.The instruction is passed on in form of string and given to cmd.

**3.6 Results**

The transcript

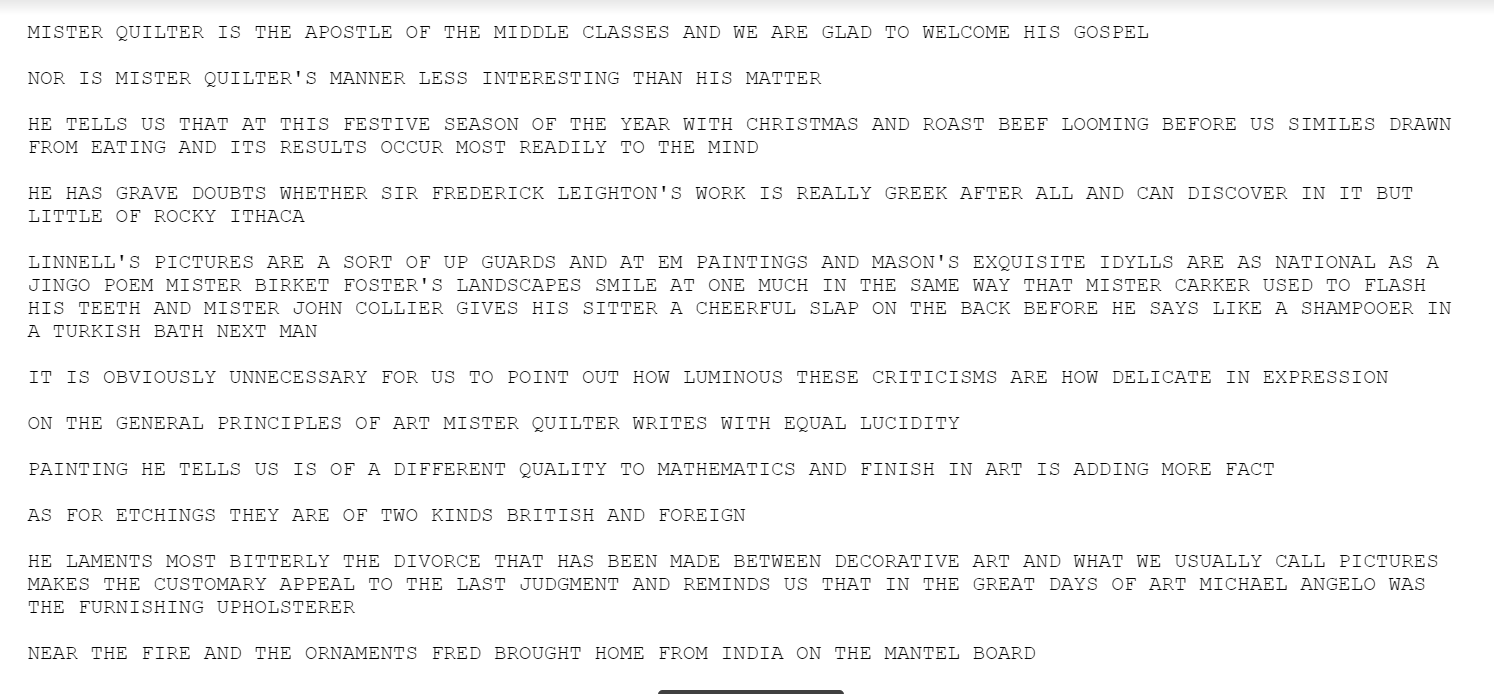
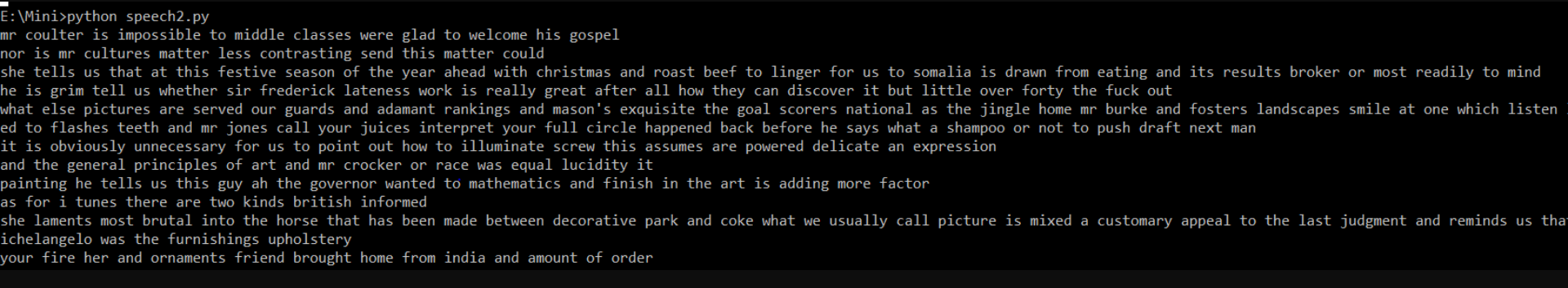


Fig 3.5Output of Pocket sphinx and its accuracy



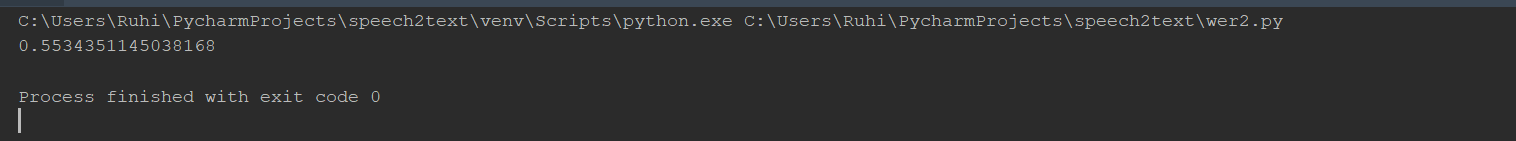


Fig 3.6 word error rate found

We can see that the error rate in the above API is approx 55%.

Similarly we tested for google API and we got an accuracy of 70%

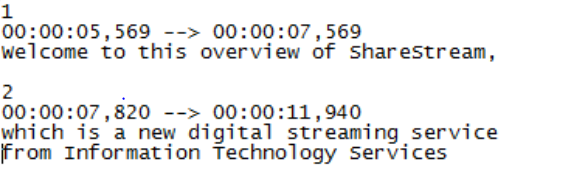
**4. Application**

**4.1 Subtitle generation**

Apart from the normal speech to text from the audio files we have also tried at generating word files from videos which is nothing but the process of subtitle generation .

* From the video/audio we generated the transcript and made a subtitle file from it

* Even though online resources suggest to use add 1 sec for each word , we preferred to use .5 sec per word and we were able to accurately generate the subtitle file this helped us to increase the accuracy of our model which significantly impacts the performance , on the lop side it is a bit computationally expensive and needs a robust computers and processing units as twice as much resolution is given to each word
* The following format for srt file was used Fig 3.3, an srt file is basically a subrip subtitle file in which we can store our subtitle generated text a dot srt file contains the time frames in which the statements occurred and is generally opened with video streaming application .



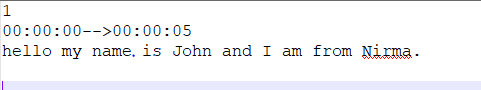
*Fig 3.7 SRT file*

* The number in top shows the sequence number, the following line shows the time duration and the the last line shows the subtitles, which helps to note when a line was spoken or heard ,this file can also be converted to other dot extensions for further use and is quite versatile in nature.
* For a Mp4 file we first converted the video file to audio file and then we got its text representation and we converted it to srt file in the required format
* For subtitle generation in real time, we made a recursive function which will take each sentence and will convert it and then copy it to the SRT file
* The recursive function ends when the user utters the words “I want to stop now”

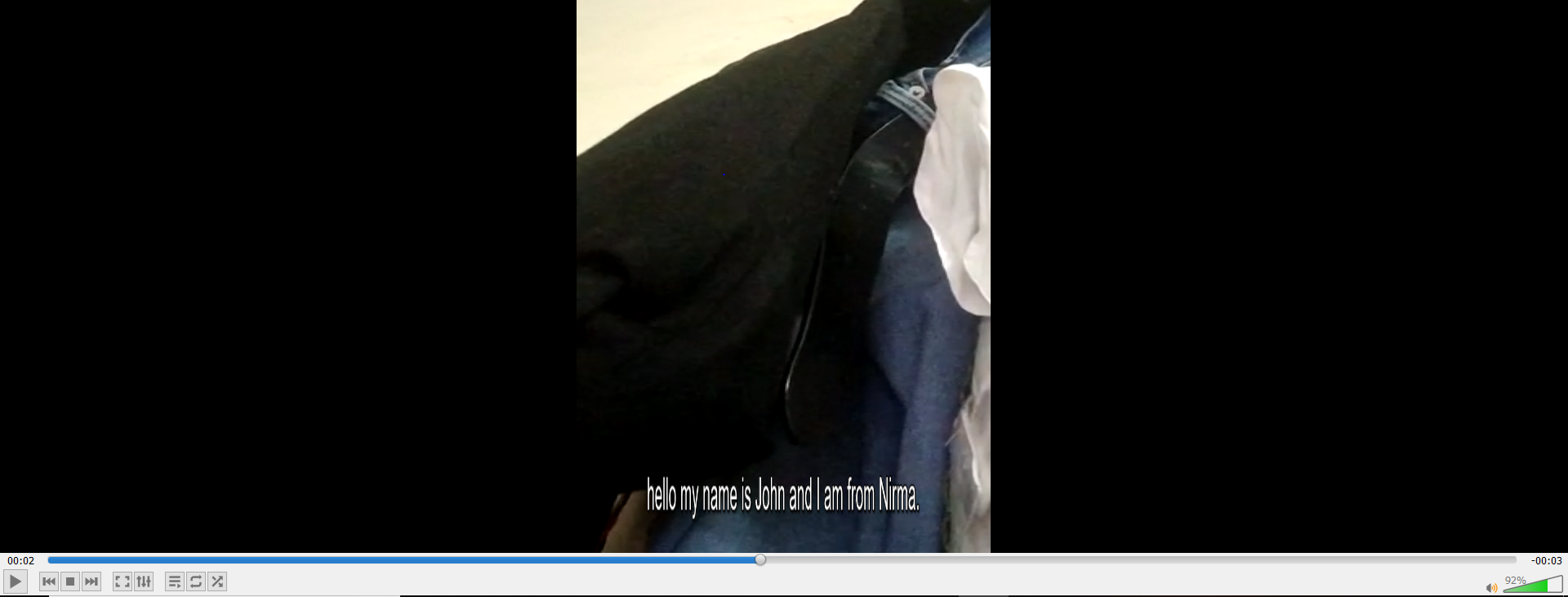
To test the mechanism we recorded a video in a noisy environment with the following transcript

“Hello my name john and I am from Nirma”

The subtitle file generated was



When testing it on the video file the following result was seen



**5 Summary and Conclusion**

**5.1 Summary**

We can say that of all the APIs tested the google speech recognition showed the most confident predictions as it made use of the enhanced models and the sequence to sequence models to significantly bring down the error , they can do this with the help of data logging and the final output can be tested efficiently by using the word error rate calculation provided by the jiwer library.

**5.2 Conclusion**

Hence we can conclude that of all the APIs available till date the google speech recognition framework works the best and constant efforts are being made in this domain to further improve the quality of predictions being made . The speech to text conversion concept has usefulness in many domains from automation of living spaces using internet of things to robots , it also helps specially abled people to live life to fullest and in general , this technology has worked towards the betterment of the mankind.

**References**

[1]Chakraborty, Chandralika, and P. H. Talukdar. "Issues and limitations of HMM in speech processing: a survey." *International Journal of Computer Applications* 141.7 (2016): 975-8887.

[2]Narang, Shreya, and Ms Divya Gupta. "Speech feature extraction techniques: a review." *International Journal of Computer Science and Mobile Computing* 4.3 (2015): 107-114.

[3]Garg, Anjali, and Poonam Sharma. "Survey on acoustic modeling and feature extraction for speech recognition." *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. IEEE, 2016.0

[4]Graves, A., Mohamed, A.R. and Hinton, G., 2013, May. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing* (pp. 6645-6649). IEEE.

[5]Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A.R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Kingsbury, B. and Sainath, T., 2012. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine*, *29*.

[6]Sanner, M.F., 1999. Python: a programming language for software integration and development. *J Mol Graph Model*, *17*(1), pp.57-61.

[7]Lutz, M., 2001. *Programming python*. " O'Reilly Media, Inc.".

Appendix A - List of Useful websites

<https://pypi.org/project/jiwer/>

<https://github.com/jitsi/asr-wer/tree/master/jiwer>

<https://pypi.org/project/SpeechRecognition/>

<https://realpython.com/python-speech-recognition/>

<http://www.openslr.org/12/>

<https://www.learnpython.org/>

<https://ffmpeg.org/>

<https://github.com/FFmpeg/FFmpeg>

<https://jupyter.org/>

<https://en.wikipedia.org/wiki/Language_model#n-gram>