Speech Recognition

Presented by:

Mohamed Khaled Abdelmonem

Mohamed Abdelfattah Younes

Abdelrahman Mostafa Anwar

Mohamed Ibrahim Mohamed

Mohamed Yousef Mohamed

Mahmoud Ayman Ahmed

Presented to Dr: Hafez Abd El Wahab

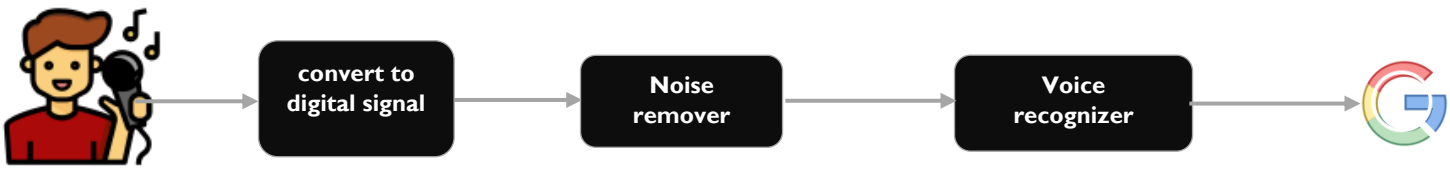
Introduction

Speech recognition, also known as computer speech recognition, or speech-to-text, is a capability which enables a program to process human speech into a written format. Speech recognition focuses on the translation of speech from a verbal format to a text one. can handle different accents and various languages. Speech recognition technology has become an increasingly popular concept in recent years. [1]

One of the most notable advantages of speech recognition technology includes the dictation ability it provides, With the help of technology, users can easily control devices and create documents by speaking. [2]

Speech recognition allows documents to be created faster because the software generally produces words as quickly as they uttered, which is usually much faster than a person can type. Dictation solutions are not only used by individuals but also by organizations that require massive transcription tasks such as healthcare and legal. [2]

Block diagram:



1) Recognizer Class:

The SpeechRecognition library has several libraries, but we will only be focusing on the Recognizer class. The Recognizer class will help us to convert the audio data into text files. [3]

2) Handling Noise:

To handle the background noise, the recognizer class has a built-in function called (adjust_for_ambient_noise function), which also takes a parameter of duration is the maximum number of seconds that it will dynamically adjust the threshold for before returning. This value should be at least 0.5 in order to get a representative sample of the ambient noise. Using this function the recognizer class listens to the audio for the specified duration seconds from the beginning of the audio and then adjusts the energy threshold value so that the whole audio is more recognizable. [3]

3) recognize_google():

we'll use the recognize_google() method on it to access the Google web speech API and turn spoken language into text.

recognize_google() requires an argument audio_data otherwise it will return an error.

US English is the default language. If your speech or audio file isn't in US English, you can change the language with the language argument. A list of language codes can be seen here. [3]

4) Audio Preprocessing:

While passing the audio data if you get an error it is due to the wrong data type format for the audio file. To avoid this kind of situation preprocessing of audio data is a must there is a class especially for preprocessing the audio file which is called AudioFile. [3]

5) Hidden Markov Model(HMM):

A Hidden Markov Model (HMM) is a statistical model which is also used in machine learning. It can be used to describe the evolution of observable events that depend on internal factors, which are not directly observable. [4]

The first component of speech recognition is, of course, speech. Speech must be converted from physical sound to an electrical signal with a microphone, and then to digital data with an analog-to-digital converter. Once digitized, several models can be used to transcribe the audio to text. [4]

Most modern speech recognition systems rely on what is known as a Hidden Markov Model (HMM). This approach works on the assumption that a speech signal, when viewed on a short enough timescale (say, ten milliseconds), can be reasonably approximated as a stationary process—that is, a process in which statistical properties do not change over time. [4]

In a typical HMM, the speech signal is divided into 10-millisecond fragments. The power spectrum of each fragment, which is essentially a plot of the signal's power as a function of frequency, is mapped to a vector of real numbers known as cepstral coefficients. The dimension of this vector is usually small—sometimes as low as 10, although more accurate systems may have dimension 32 or more. The final output of the HMM is a sequence of these vectors. [4]

To decode the speech into text, groups of vectors are matched to one or more phonemes—a fundamental unit of speech. This calculation requires training, since the sound of a phoneme varies from speaker to speaker, and even varies from one utterance to another by the same speaker. A special algorithm is then applied to determine the most likely word (or words) that produce the given sequence of phonemes. [4]

One can imagine that this whole process may be computationally expensive. In many modern speech recognition systems, neural networks are used to simplify the speech signal using techniques for feature transformation and dimensionality reduction *before* HMM recognition. Voice activity detectors (VADs) are also used to reduce an audio signal to only the portions that are likely to contain speech. This prevents the recognizer from wasting time analyzing unnecessary parts of the signal. [4]

6) Hidden Markov Models in NLP:

Our main focus is on those applications of NLP where we can use the HMM for better performance of the model, we can see that one of the applications of the HMM is that we can use it in the Part-of-Speech tagging. [4]

7) What is POS-tagging?.

The part of speech indicates the function of any word, like what it means in any sentence. There are commonly nine parts of speeches; noun, pronoun, verb, adverb, article, adjective, preposition, conjunction, interjection, and a word need to be fit into the proper part of speech to make sense in the sentence. [4]

POS tagging is a very useful part of text preprocessing in NLP as we know that NLP is a task where we make a machine able to communicate with a human or with a different machine. So it becomes compulsory for a machine to understand the part of speech. [4]

Classifying words in their part of speech and providing their labels according to their part of speech is called part of speech tagging or POS tagging OR POST. Hence the set of labels/tags is called a tagset. [4]

8) POS Tagging With Hidden Markov Model:

We can say that in the case of HMM is a stochastic technique for POS tagging. Let's take an example to make it more clear how HMM helps in selecting an accurate POS tag for a sentence. [4]

The probability for a word to be in a particular class of part of speech is called the Emission probability.

Mary Jane can see will

The spot will see Mary

Will Jane spot Mary?

Mary will pat Spot

The below table is a counting tableau for the words with their part of speech type

Words	Noun	Modal	Verb
mary	4	0	0
jane	2	0	0
will	1	3	0
spot	2	0	1
can	0	1	0
see	0	0	2
pat	0	0	1

Let’s divide each word’s appearance by the total number of every part of speech in the set of sentences.

Words	Noun	Modal	Verb
mary	4/9	0	0
jane	2/9	0	0
will	1/9	3/4	0
spot	2/9	0	1/4
can	0	1/4	0
see	0	0	2/4
pat	0	0	1/4

Here in the table, we can see the emission probabilities of every word.

Now as we have discussed that the transition probability is the probability of the sequences we can define a table for the above set of sentences according to the sequence of part of speech.

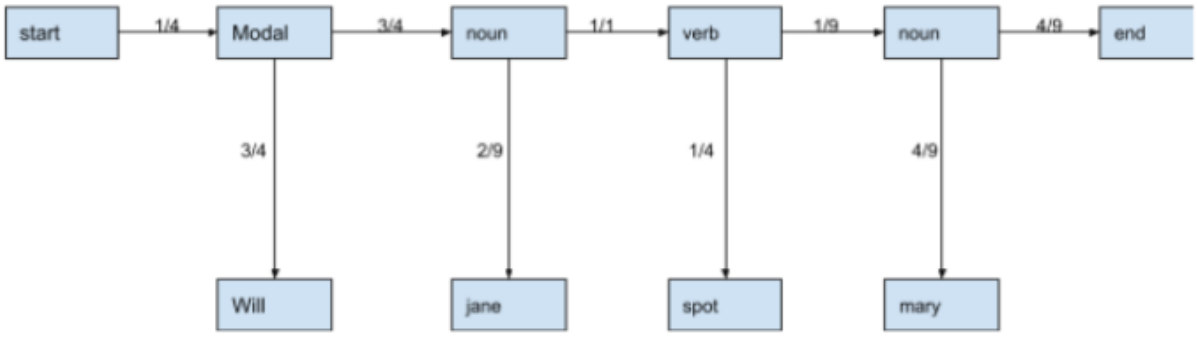
	Noun	Modal	Verb	End
Start	3	1	0	0
Noun	1	3	1	4
Model	1	0	3	0
Verb	4	0	0	0

Now in the table, we are required to check for the combination of parts of speeches for calculation of the transition probabilities. For example, we can see in the set of sentences modal before a verb has appeared 3 times and 1 time before a noun. This means it has appeared in the set for four-time and the probability of coming modal before any verb will be ¾ and before a noun will be ¼. Similarly performing this for every entity of the table:

	Noun	Modal	Verb	End
Start	3/4	1/4	0	0
Noun	1/9	3/9	1/9	4/9
Model	1/4	0	3/4	0
Verb	4/4	0	0	0

Here the above values in the table are the respective transition values for a given set of sentences.

Let's take the sentence "Will Jane spot Mary?" out from the set and now we can calculate the probabilities for every part of speech using the above calculations.



In the above image, we can see that we have emission probabilities of the words in the sentence given in the vertical lines and the horizontal lines are representing all the transition probabilities.

9) search in browser:

Syntax of search is as follows

search(searchfor, tld='.edu', lang='en', num=10, start=0, stop=None, pause=2.0)

So here you will see there are several attributes present in search function. No need to panic, they are very easy to understand. Let's take a look at this attribute one by one.

searchfor is a variable that stores the value that you need to search.
lang is nothing but the language of your search and it is optional.
num is variable that justifies the number of links in your result.
start and stop specifies the starting and ending index of search result. So if start =1 that means the link on the zeroth position will be skipped and stop = 1 that means you will get only one result. You can adjust the value of start and stop accordingly.
tld is the extension of your search that means it holds the domain you need to search like .com, .in, .edu, .mil, .co.in, etc.
pause is the time frame for the result to scrape. Remember the value of pause should not be too long or too short. In our opinion pause at 2 or 3 works fine.

That's the search function that will do all the tasks for us. Now we just need to enter the string we are searching for, loop through our search function and finally printing the result.

Reference:

1) https://www.techtarget.com/searchcustomerexperience/definition/speech-recognition

2) https://www.ibm.com/cloud/learn/speech-recognition

3) https://realpython.com/python-speech-recognition/

4) https://analyticsindiamag.com/a-guide-to-hidden-markov-model-and-its-applications-in-nlp/