Speech Recognition

# Introduction

## Prologue

it has been always the case that new ways to communicate with our computers are appearing everyday every hour. In the past people used to interact with their computers mainly through keyboards and text-based systems Zen came about the revolution of graphical user interfaces and the advent of computer mice, after that the Interactions With our computing devices became dominated with touch interfaces. Now to accommodate people with disabilities and to devise new ways to interact with our devices in scenario in scenarios when Traditional methods cannot be visible there has been a rise in Voice interactions with all sorts of electronic devices with the proliferation of digital assistants such as Apple’s Siri and Google Assistant it's only evident that this mode of interaction Is popular with clients.

## Why Speech-To-Text

Before popularisation of voice commands there have been developments in inferring meanings from text through Natural Language Processing (NLP) and Natural Language Understanding (NLU) and those were primarily text based and to build upon these technologies, it only made sense to convert voice or speech to take it to its text representation and run the established algorithms to infer what a user wants to make a better service. There have been other methodologies that work on the Voice or the wavy nature of speech, if so to speak, but they haven't been as successful as a text -based methods so converting speech into text then processing the resulting text makes much more sense in this regard.

## History of Speech-To-Text

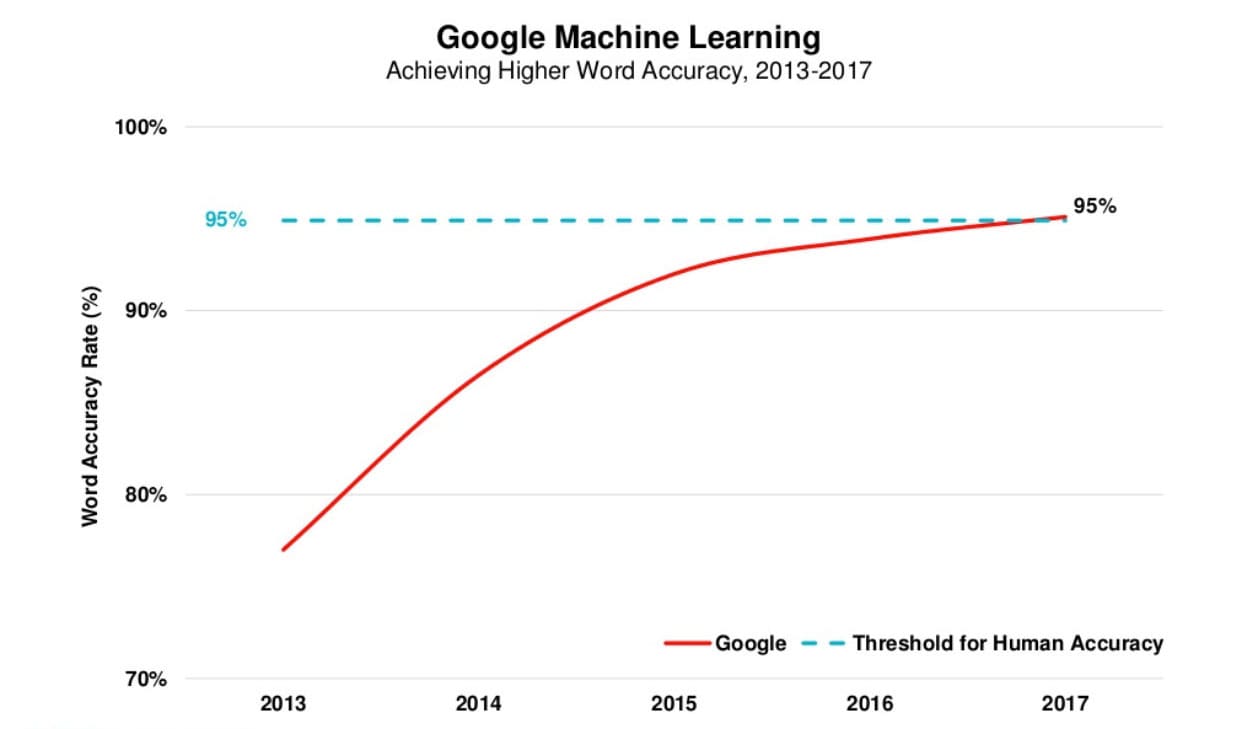
With the ever-increasing power to perform computations, the advances in artificial intelligence, especially in the Machine Learning subfield, and the abundance of speech data, many of which accompanied with its text representation, there has been a huge boon to the capabilities of computers to recognise human speech, and recently they have become as efficient as humans in this regard.

Figure Speech Accuracy vs Years

### Milestones

Figure Recognition Accuracy vs Years

#### 1950s to 1960s

Bell labs first designed Audrey Systems: a speech recognition device focused on recognising numbers. Ten later, in 1960s, IBM came with the Shoebox system; it could understand and respond to 16 words in English.

#### 1970s

Several advancements in this field were made in this decade courtesy of none other than US Department of Defence’s (DoD) DARPA, the original progenitors of Internet. They created a system called Speech Understanding Research (SUR); its child system called Harpy at Carnegie-Mellon University was able to understand 1,000 English words, the equivalent of a 3-year-old’s vocabulary.

#### 1980s

A great growth in the vocabulary understood by recognition systems from thousands of words to tens of thousands of words. A breakthrough in statistical algorithms known as Hidden Markov Models were of tantamount importance towards the realisation of these advancements. HMMs work by estimating the probability of a sound utterance being a word instead of using fixed sound patterns to map to words directly in a deterministic way.

#### 1990s

Speech Recognition became popular with the masses in the 90s because Personal Computers had faster processors capable of running recognition systems like Dragon Dictate which became widely used. Also, a accompany called BellSouth was pivotal in popularising such systems through the introduction of Voice Portal (VAL) which was a dial-in phone service based on Speech Recognition. This system gave to myriad of phone tree services currently in use today.

#### 2000s

By the time it was 2001, Speech Recognition could attain around 80% in accuracy; however, there was not much development in the field till the arrival of Google Voice Search in the latter end of the decade. Due to the popularity of Google, this service was accessed by millions of people, which gave valuable datasets to be worked on and the needed processing power to serve the users’ requests were offloaded to its datacentres, therefore allowing a higher accuracy, thus quality, service.

#### 2010s

In 2011 Apple launched its Siri App, much like Google Voice Search, now known as Google Assistant, it came like an App that is a point that helped in making both popular in the age of handheld devices. Then came Amazon’s Alexa, its competitor Google Home, and Microsoft’s Cortana. With the rise of Machine Learning techniques and ever-greater datasets, accuracy became higher and higher. In 2016 IBM achieved a word error rate of 6.9%; Microsoft beat it in 2017 and achieved an error rate of 5.9%; however, Google now reigns supreme with its error rate of just 5.5%, which is the rate of human error in recognising speech.

|  |  |  |
| --- | --- | --- |
| 1952 | Invention | A team at Bell Labs designs the Audrey, a machine capable of understanding spoken digits.[[1]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-PCWorld-1) |
| 1962 | Demonstration | [IBM](https://en.wikipedia.org/wiki/IBM) demonstrates [the Shoebox](https://en.wikipedia.org/wiki/IBM_Shoebox), a machine that can understand up to 16 spoken words in English, at the [1962 Seattle World's Fair](https://en.wikipedia.org/wiki/1962_Seattle_World%27s_Fair).[[4]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-4) |
| 1971 | Invention | IBM invents the Automatic Call Identification system, enabling engineers to talk to and receive spoken answers from a device.[[5]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-Pioneering-5) |
| 1971–1976 | Program | [DARPA](https://en.wikipedia.org/wiki/DARPA) funds five years of speech recognition research with the goal of ending up with a machine capable of understanding a minimum of 1,000 words. The program led to the creation of the [Harpy](https://en.wikipedia.org/wiki/Harpy) by [Carnegie Mellon](https://en.wikipedia.org/wiki/Carnegie_Mellon), a machine capable of understanding 1,011 words.[[1]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-PCWorld-1) |
| Early 1980s | Technique | The [hidden Markov model](https://en.wikipedia.org/wiki/Hidden_Markov_model) begins to be used in speech recognition systems, allowing machines to more accurately recognize speech by predicting the probability of unknown sounds being words.[[1]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-PCWorld-1) |
| Mid 1980s | Invention | IBM begins work on the Tangora, a machine that would be able to recognize 20,000 spoken words by the mid 1980s.[[5]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-Pioneering-5) |
| 1987 | Invention | The invention of the World of Wonder's Julie Doll, a toy children could train to respond to their voice, brings speech recognition technology to the home.[[1]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-PCWorld-1) |
| 1990 | Invention | Dragon launches [Dragon Dictate](https://en.wikipedia.org/wiki/Dragon_Dictate), the first speech recognition product for consumers.[[1]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-PCWorld-1) |
| 1993 | Invention | [Speakable items](https://en.wikipedia.org/wiki/Speakable_items), the first built-in speech recognition and voice enabled control software for [Apple](https://en.wikipedia.org/wiki/Apple_Inc.) computers. |
| 1993 | Invention | [Sphinx-II](https://en.wikipedia.org/wiki/CMU_Sphinx), the first large-vocabulary continuous speech recognition system, is invented by [Xuedong Huang](https://en.wikipedia.org/wiki/Xuedong_Huang).[[6]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-6) |
| 1996 | Invention | IBM launches the MedSpeak, the first commercial product capable of recognizing continuous speech.[[5]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-Pioneering-5) |
| 2002 | Application | [Microsoft](https://en.wikipedia.org/wiki/Microsoft) integrates speech recognition into their Office products.[[7]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-7) |
| 2006 | Application | The [National Security Agency](https://en.wikipedia.org/wiki/National_Security_Agency) begins using speech recognition to isolate keywords when analyzing recorded conversations.[[8]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-8) |
| 2007 | Application | Microsoft releases Windows Vista, the first version of Windows to incorporate speech recognition.[[9]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-9) |
| 2007 | Invention | [Google](https://en.wikipedia.org/wiki/Google) introduces [GOOG-411](https://en.wikipedia.org/wiki/GOOG-411), a telephone-based directory service. This will serve as a foundation for the company's future Voice Search product.[[10]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-10) |
| 2008 | Application | Google launches the Voice Search app for the [iPhone](https://en.wikipedia.org/wiki/IPhone), bringing speech recognition technology to mobile devices.[[11]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-11) |
| 2011 | Invention | Apple announces [Siri](https://en.wikipedia.org/wiki/Siri), a digital personal assistant. In addition to being able to recognize speech, Siri is able to understand the meaning of what it is told and take appropriate action.[[12]](https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition#cite_note-12) |
| 2014 | Application | Microsoft announces Cortana, a digital personal assistant similar to Siri. |
| 2014 | Invention | [Amazon](https://en.wikipedia.org/wiki/Amazon.com) announces the Echo, a voice-controlled speaker. The Echo is powered by Alexa, a digital personal assistant similar to Siri and Cortana. While Siri and Cortana are not the most important features of the devices on which they run, the Echo is dedicated to Alexa. |

Table 1 Showing The timeline of advancements in Speech Recognition Systems

# How Speech Recognition Systems Work

The first step in designing any algorithm or processing pipeline, whether it was deterministic or statistical, is the choice of the input features. In the field of Recognition two systems are mainly used Linear Prediction Cepstral Coefficients (LPCC) and Mel Frequency Cepstral Coefficients (MFCC). The other important aspect is the choice of the algorithm itself that will process the input, and we have plenty of choices in this regard: the aforementioned HMMs, Artificial Neural Networks (ANNs) and LSTM Networks. In practice a combination of HMMs and ANNs works well and achieves high accuracy.

## MFCC

When a human speaks, the sounds he makes are filtered through the shape of his larynx, teeth and tongue. This leads two iPhone M being represented is a short time scales by a set of Cesptra, a changing in the frequency representation of the voice, Answer isn't much difference between How the human brain perceives neighboring frequencies as the pitch of the sound increases. These features are replicated by MFCC to get the phoneme In a four form suitable for a computer algorithm this exploits the fact that a word just string of phonemes and by getting those we infer a word to a high degree of confidence. MFCC was developed in a set of experiments to understand how humans speak and understand speech.

Steps to Compute MFCC:

* Cut the sound into tiny frames each typically is 20ms to 40ms.
* Compute the Fourier or rather the Short Time Fourier Transform.
* Transform the Spectrum to the Mel-Scale.
* Calculate the Log of the result
* Do another Fourier Transform or a Discrete Cosine Transform.

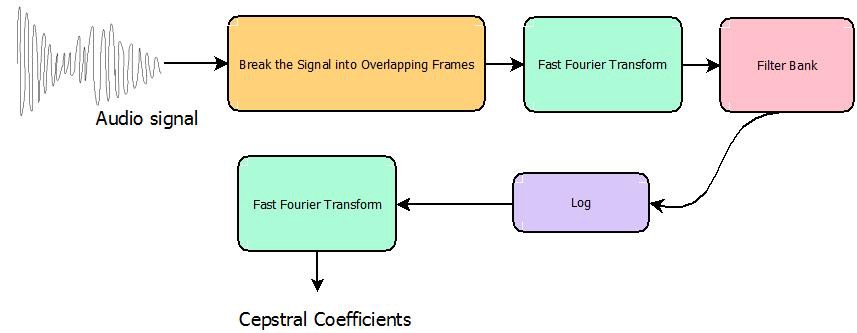


Figure 2 A block diagram for an algorithm to compute MFCC

## Hidden Markov Models (HMMs)

### Markov Chains

Let us begin with first identifying what is a Markov State and a Markov Chain. In a random process in which the random variable exists in a certain value or a *state* if it changes states and the probability it settles in a certain state is only given by the previous state *irrespective of its history or how it ended up there* is called a Markov State. A Markov Chain is just a series of successive states

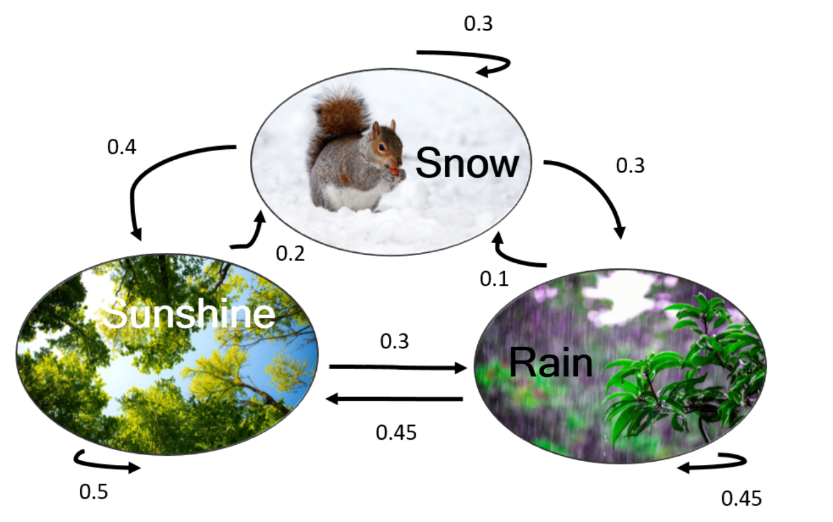
Image for post

Figure 3 Markov Chain example

### Hidden Markov Models

Hidden Markov Models are just random or probabilistic processes which its underlaying Markov Chain is *hidden* from us and we must come up with a way to reconstruct them from the observations we make. There exist many algorithms to reconstruct them one of note is **Viterbi algorithm**.

#### HMMs for the Task of Speech Recognition

In Speech Recognition we want to estimate the word **W** given utterances **O**

Using Bayes Theorem, we can reformulate the problem as

This splits the task into two components **P(O|W)**, which is called the acoustic model i.e. how likely an utterance given a word as an input, and **P(W)**, which is called the language model, how likely a word to occur in a text of the language overall. In most speech recognition systems, the acoustic model is represented by Hidden Markov Models (HMMs), which are a generative model of a linguistic unit of speech (phone, word). Each HMM is a finite state machine with n states whereby each state, besides the first and last, has specific output probabilities and each state transition between states is associated with a transition probability. Each state represents an utterance and the end of a chain represents a word.

A picture containing object

Description automatically generated

Figure 4 A Markov Model for Speech Recognition

## Artificial Neural Networks

It is a class of Machine Learning algorithms in which the biological neural pathways in the brain a mimcked through computational graphs. Each node in the graph is analogues to a neuron, and the connections between the graphs are analogues to synapses in which their thickness is represented by a weight on the connection which gets multiplied by the input. Each node produces a non-linear output from a linear combination of the inputs. where y is the output vector, ***w*** is the weight vector, ***x*** is the input vector and ***b*** a bias coeffecient; *f* is the non-linear activation function. This non-linearity in the output of nodes allows the neural networks to produce highly complex functions that estimates or approximates the outputs to a high degree of accuracy.

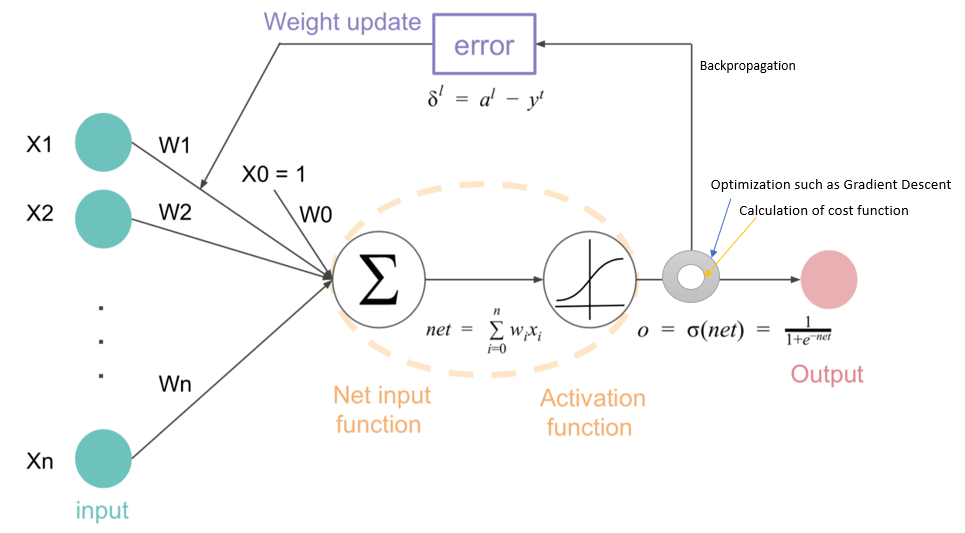
The problem with neural networks that prevented itswidespread adoption before recent is that it requires huge resources to train, by training it is meant that a suitable set of weight vectors is calculated for the approximating function, using the most known common algorthms such as Gradient Descent, RMSprob, AdaGrad and Adam. This is done with aid of a *Loss function* ***L****,* a function that determines how much is the actual output far away from the desired output. The minmisation of this function is the objective of the aforementioned algorithms by tuning weights ***w*** and biases ***b***. All make use of the derivative of the loss function with the respective weight .

Figure A graphical representation of an Artificial Neural Network

Figure 5 A graph showing the process of Backprobagation

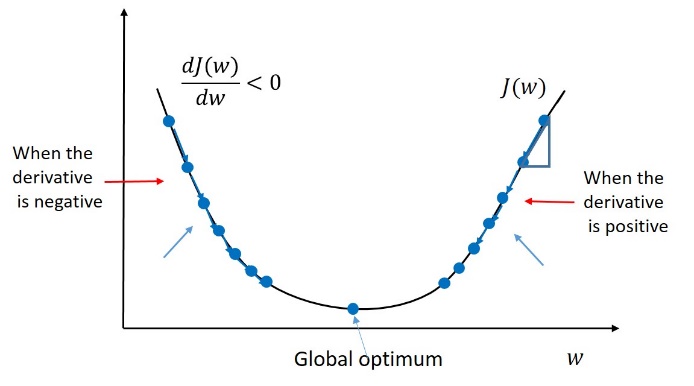


Figure Gradient descent, the father of most of the mentioned algorithms

## The Google Speech-To-Text API

Google Speech-To-Text API was used because it offers many of features that will be ease the development of the project. For starters it offers state-of-the-art accuracy as mentioned before and it has very low latency due to the distribution of Google’s datacentres worldwide. Most importantly it is an easy-to-use API that integrates well with platforms worked on Unity using C# and python.

Key features

* Speech adaptation
  + It can use domain-specific languages and convert spoken numbers into currencies, dates and addresses using classes.
* Streaming speech recognition
  + It offered streaming the voice from the VR set to its servers to do real-time speech recognition.
* Noise robustness
* Content filtering
  + It can filter profanity if the situation arises.
* It is very affordable for our use
* Well-documented

## Processing of the generated text

The text received from the API is tokenised into words then appended to the end of an array tat gets emptied every minute. Using a timer that runs on a separate thread in the code that fires every minute it will calculate the number of words uttered in this minute by the speaker using a variable that contains all the words spoken on or after the last call for timer’s handling method till this call; based on it the method will decide whether the speaker was speaking fast or slow or about right.

# References

* <https://cloud.google.com/speech-to-text>
* <https://cloud.google.com/speech-to-text/docs/basics>
* <https://www.youtube.com/watch?v=mNSQ-prhgsw>
* <https://medium.com/@jonathan_hui/speech-recognition-gmm-hmm-8bb5eff8b196>
* <https://en.wikipedia.org/wiki/Hidden_Markov_model#Applications>
* <https://en.wikipedia.org/wiki/Hidden_Markov_model#:~:text=Hidden%20Markov%20Model%20(HMM)%20is,to%20learn%20about%20by%20observing%20>.
* <https://en.wikipedia.org/wiki/Mel-frequency_cepstrum>
* <https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd>
* <https://www.quora.com/What-are-the-best-algorithms-for-speech-recognition>
* <https://sonix.ai/history-of-speech-recognition#:~:text=1950s%20and%2060s,to%2016%20words%20in%20English>.
* <https://en.wikipedia.org/wiki/Timeline_of_speech_and_voice_recognition>