



# AUTOMATIC SPEECH RECOGNITION

Capstone Project - IIITH AIML Course 2023

---

## Group 22

Prerna Rohatgi (NPCI)  
Gandi Tejaswini (NPCI)  
Kaparthi Sathwika (NPCI)  
Guduru Sai Deepthi (NPCI)



## 1. Introduction

Automatic speech recognition is the application of Machine learning or AI where human speech is processed and converted into readable text. We can find numerous applications such as Instagram for real-time captions, Spotify for podcast transcriptions, Youtube video transcription, Zoom meeting transcriptions, etc. The field has grown exponentially over the last few years. An explosion of applications taking advantage of ASR technology in their products to make audio and video data more accessible. In this paper, we demonstrate an end-to-end process of building an Automatic Speech Recognition (ASR) model for converting speech to text using HMM/GMM. We have highlighted the various stages during the incremental process and outcomes at each stage from development to deployment.

## 2. Literature Review

A few decades ago, computers were limited to basic functions and had no capability to understand human language. However, with the rise of natural language processing (NLP), computers can now recognize and translate spoken words into text. One of the most prominent applications of NLP is speech recognition, which allows computers to decipher the sounds we make when we speak. Interesting history behind the growth in this domain; the statistical ASR systems like Hidden Markov Models (HMM, with static patterns) and Gaussian Mixture Models (GMM, with sequential patterns) that were trained on less hours of data with medium performance levels. Followed by the Deep learning Acoustic models, HMM-DNN, that incorporated neural networks and finally the E2E networks. The modern End-to-end networks use deep neural networks which possess the capability to produce letters directly from acoustics. They use tokenizers like sentencepiece, wordpiece (spe, wpe respectively) in their speech pipeline. Some commonly used toolkits are ESPNet and FAIRSEQ.

As we go on increasing the amount of training data, the model size increases thereby improving the quality of the transcriptions and lower WER rate (the lower the better). This however, requires larger compute to either train or inference from such a model.

## 3. Applications

There are many uses and applications of ASRs. They vary from self-servicing call centers and self-ordering machines to mobile devices operated by voice commands. Siri, Google assistant and Alexa are the most famous applications of ASR, where you talk to your

mobile asking some questions or giving some voice commands, live captioning and transcriptions, voice biometrics, language translation and so many more.

## 4. Approach

There are different approaches to Automatic Speech Recognition, viz. conventional HMM (Hidden Markov Models) and GMM (Gaussian Mixture Models) and end-to-end deep learning models. In this project, we aim to build and deploy a model that can generate the written text from the speech with a decent accuracy.

### 4.1 Background study

ASR algorithms work through three types of modeling: acoustic modeling, language modeling, and pronunciation modeling (Fig.1).

1. **Acoustic modeling in ASR** deals with the relationship between linguistic units of speech (e.g., phonemes) and audio signals.
2. **Language modeling in ASR** looks for patterns in sequences of words and therefore helps to distinguish between different words with the same sound.
3. **Pronunciation modeling in ASR** provides a mapping between a conventional symbolic transcript of speech, which can exhibit varying degrees of arbitrariness, and an acoustically/phonetically motivated one.

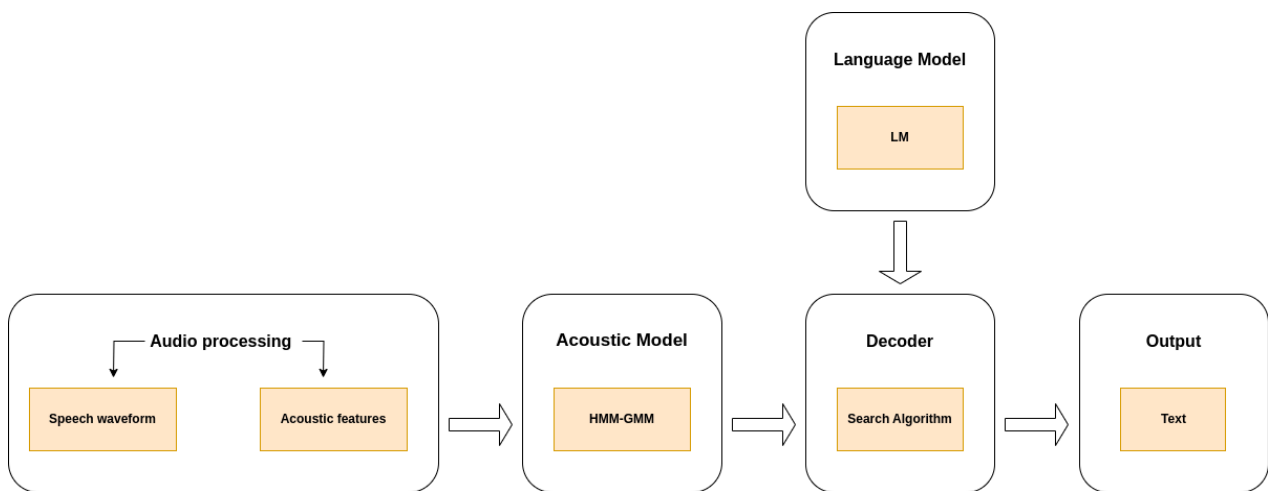




Fig 1. General architecture of ASR systems



Like any other machine learning or deep learning project, the speech pipeline generally begins from feature extraction; in this case the input data being audio files. Acoustic elements are extracted from raw speech for each speech frame. These features are known as Mel frequency Cepstral Coefficients or MFCCs. These features are then passed to the acoustic model, like HMM-GMM, for mapping acoustic features to sequences of phonemes. The language model picks up the occurrence of words given in recent context and comes in handy when dealing with identical acoustics. Further, the pronunciation model (if added) derives words from phonemes. The decoder block contains search algorithms like Viterbi, beam search or other decoding mechanisms to predict and produce text transcripts.

## 5. Project implementation and workflow

To build an efficient ASR system, we require certain hours of labeled data and categorize the data as 'dev-clean' and 'test-clean' datasets for building the model. Once modeling is done with these smaller data sets, we will proceed with 'train-clean'/'train-other' data sets of larger sizes as a training set. Now, 'dev-clean', 'test-clean', and 'test-other' datasets are used for validation/testing purposes only.

For the current scope of the project, we use a monolingual dataset and tools like Kaldi toolkit, Pytorch and other open source audio processing libraries and packages and opt to train the HMM-GMM acoustic model (Fig 2). Consequently, fine tuning on more data to observe the results. After training our network, we evaluate our model performance with . Word Error Rate (WER) metric. WER compares the predicted output and the target transcript, word by word (or character by character) to figure out the number of differences between them. A difference could be a word that is present in the transcript but missing from the prediction (counted as a Deletion), a word that is not in the transcript but has been added into the prediction (an Insertion), or a word that is altered between the prediction and the transcript (a Substitution). Lower the WER, higher the accuracy.

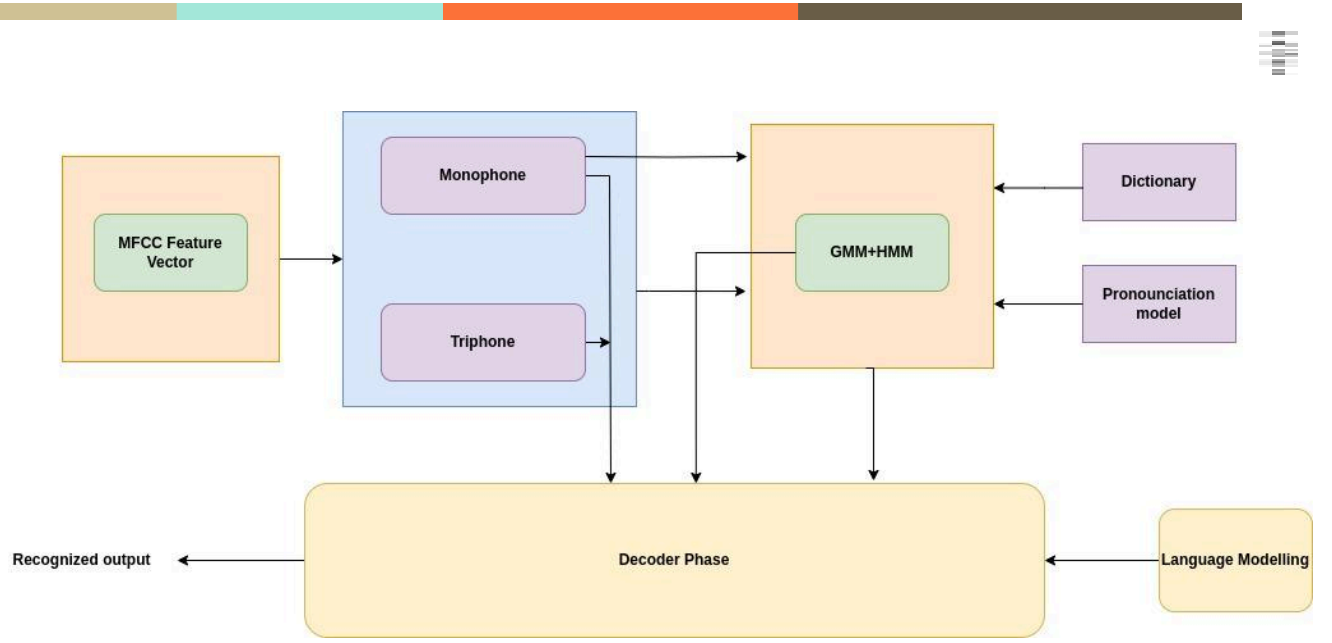


Fig 2. End-to-End ASR building pipeline

## 5.1 Data and Data pre-processing

Our data is Telugu language and the dataset comprises 50 hours of audio data out of which train and test split was 45 hours and 5 hours respectively with a sampling rate of 16000 Hz. The initial data preparation includes creation of four: text, wav.scp, utt2spk and lexicon.txt. The text file contains the audio file id and the text transcription; wav.scp file contains audio file name and its corresponding path on the system; utt2spk includes the utterance to speaker id wherein we assume each utterance is spoken by an individual speaker. Further preparation of non\_silence\_phones.txt, silence\_phones.txt, optional\_silence\_phones.txt using lexicon.txt. As the name indicates, non\_silence\_phones.txt file contains a list of all the phones that are not silent; silence\_phones.txt will contain a 'SIL' (silence) and 'oov' (out of vocabulary) phones.

Example :

- 1) text - 002140077 చేరుకోవాల్సిన చివరి లైను అల్లంత దూరంలో కనిపిస్తూనే వుంది
- 2) wav.scp-000780122  
/home/Desktop/telugu\_dataset/Telugu\_Interspeech\_Task1\_Valid\_labelled\_13-08-20/21\_07-20/000780122.wav
- 3) utt2spk - 000600034 000600034
- 4) lexicon.txt- word phonetic\_sequence



## 5.2 Feature extraction

Speech signals are converted into a sequence of feature vectors known as Mel Frequency Cepstral Coefficients (MFCCs) or filterbanks.

## 5.3 Experimental Stages and Outcomes

### Training process

Monophone training is a crucial step in building acoustic models for speech recognition systems. Monophones model phonemes in isolation without considering context. They serve as a foundation for complex models. Forced alignment is performed using a monophone model to align the acoustic features (such as MFCCs) with the corresponding phonetic transcriptions at the monophone level. This step ensures that the training data is properly aligned for training monophone models. Triphone training in Kaldi ASR enhances the accuracy of acoustic models by capturing context-dependent phonetic variations, leading to improved speech recognition performance. Unlike monophones, which represent phonemes in isolation, triphones consider the context in which phonemes occur. This means that instead of modeling each phoneme independently, triphones model phonemes in the context of both preceding and following phonemes. Forced alignment is then performed again, but this time using the triphone models, which take into account the context in which phonemes occur. By training both monophone and triphone models, a speech recognition system can effectively capture both phonetic characteristics and contextual dependencies in speech signals, leading to improved overall performance.

- Stage 1  
Training with a subset of train data (10 hours). Resulting Word Error rate was 29%.
- Stage 2  
Fine Tuning with an entire train data (50 hours). Resulting Word Error rate was 31%.

## 5.4 Deployment

We deployed our trained model on two frameworks for user-friendly interactions with the model. The REST API endpoint using Fastapi framework Fig 3(b) and in-built Swagger UI, enables the user to upload an audio and get the text transcription. whereas, the Gradio Fig 3 (a) interface allows recording a live audio for transcription.

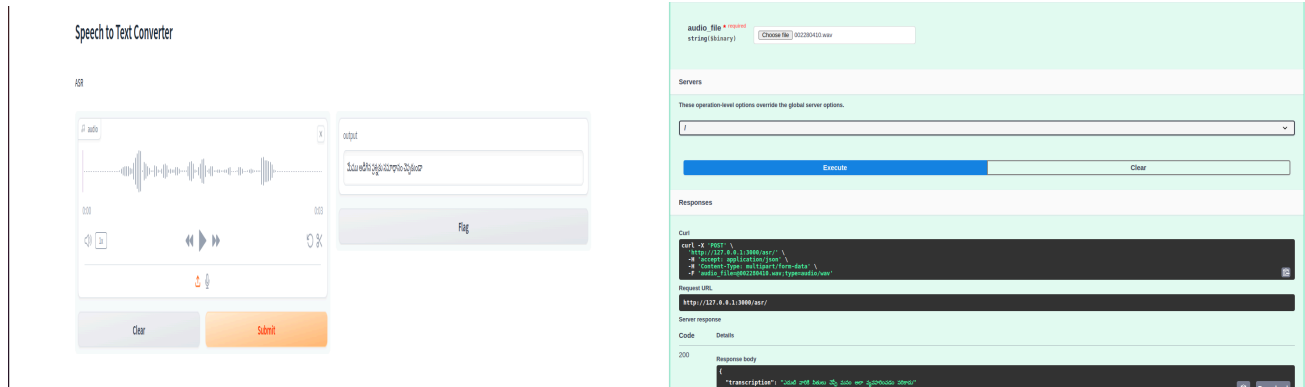


Fig 3. (a) Gradio. Fig 3. (b) FastApi

## 6. Challenges and Limitations

While speech recognition technology has come a long way, there are still challenges that need to be addressed to achieve even greater accuracy and versatility.

Fine tuning processes took long hours limiting our asynchronous experiments as a team. Also, our model struggled with robustness to variabilities such as:

- Environment : Type of noise, echo and reverberations, interfering speakers
- Transducer : Speech generated from telephone, microphone have varying sampling rates
- Speaker characteristics: Speaker age, gender
- Speech styles : Accent and speed of speech, continuous or isolated words in speech or tone

## 7. Conclusion

While HMM/GMM models have been foundational in ASR research, they represent an older paradigm compared to newer neural network-based approaches like deep learning. As a result, they may not fully leverage the advancements in modeling techniques and architectures. Recent times have seen continuous improvement towards making an ASR Model robust, there are constraints such as high computation power , unavailability of data for spoken languages (or low-resources languages) are some challenging scenarios. Models can be challenging to train or optimize if there are a lot of trainable parameters. However, modern techniques like self-supervised pre-training and finetuning can boost the speech recognition domain and open up a world of possibilities for voice assistants and chatbots.

## 8. Citations and References

- 1) Automatic Speech Recognition: Systematic Literature Review:  
<https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9536732>
- 2) Speech Recognition by Machine: A Review <https://arxiv.org/pdf/1001.2267>
- 3) Word Error Rate [https://en.wikipedia.org/wiki/Word\\_error\\_rate#References](https://en.wikipedia.org/wiki/Word_error_rate#References)
- 4) <https://towardsdatascience.com/audio-deep-learning-made-simple-automatic-speech-recognition-asr-how-it-works-716cfc>
- 5) <https://www.toolify.ai/ai-news/demystifying-speech-recognition-speech-to-text-voice-recognition-and-speech-synthesis-15889>
- 6) Automatic speech recognition - An overview (Microsoft)  
<https://www.youtube.com/watch?v=q67z7PTGR>