Speech to Text Model (Marathi Language)

# Introduction:

This document outlines the implementation and evaluation of language transcription model for speech in Marathi language utilizing “Aditya02/Vistar\_Marathi\_Model” based on Whisper architecture. The primary objective of this model is to transcribe the Marathi audio into text format.

# Goals:

1. To develop an NLP model to accurately transcribe the speech to text in Marathi language.
2. Evaluating the model with the given set of audio files and their corresponding transcripts.

# Model Selection:

Several pre\_trained models are readily available in the hugging face library to solve diverse problems. We can choose any model based on our problem. Transformers library is used to load the required models.

In this assignment, “Aditya02/Vistar\_Marathi\_Model” was chosen for its Whisper architecture. This model provides a good accuracy in predicting the Marathi words.

# Model Implementation:

The model has whisper-based architecture and it was fine tuned to get a better performance.

1. Created the model using AutoModelForSpeechSeq2Seq class and a specified processor for that model using AutoProcessor class from transformers library.
2. Created a pipeline, so that the pipeline can take multiple audio files and convert them into text.
3. Used exception handling to raise required errors during the execution of program.

# Model Evaluation:

The model was evaluated on a set of 1816 Marathi audio file to assess its transcription capabilities.

The predicted texts and reference texts are appended into 2 separate lists.

*predicted\_texts=[]*

*for file\_name in file\_names:*

*file\_path = os.path.join(directory,file\_name)*

*result = pipe(file\_path)*

*predicted\_texts.append(result['text'])*

*reference\_file = '/content/drive/MyDrive/common\_voice\_test/trans.txt'*

*reference\_texts = []*

*with open(reference\_file,'r', encoding='utf-8') as f:*

*for line in f:*

*text = line.strip().split('\t')*

Utilized the jiwer module to calculate the word error rate between the predicted texts and reference texts from the given file.

The word error rate of each sentence is then appended to another list. Calculated the mean for the list to get the average word error rate for our test dataset.

*errors = []*

*for i in range(len(predicted\_texts)):*

*error = jiwer.wer(reference\_texts[i], predicted\_texts[i])*

*errors.append(error)*

*print(f"The average  word error rate is: {np.mean(errors)}")*

# Results:

Word Error Rate: 0.22390942390942387

# Conclusion:

This whisper-based architecture has done a good job and get a word error rate of 22%. Anyone can fine tune the model using dataset in order to improve the performance.