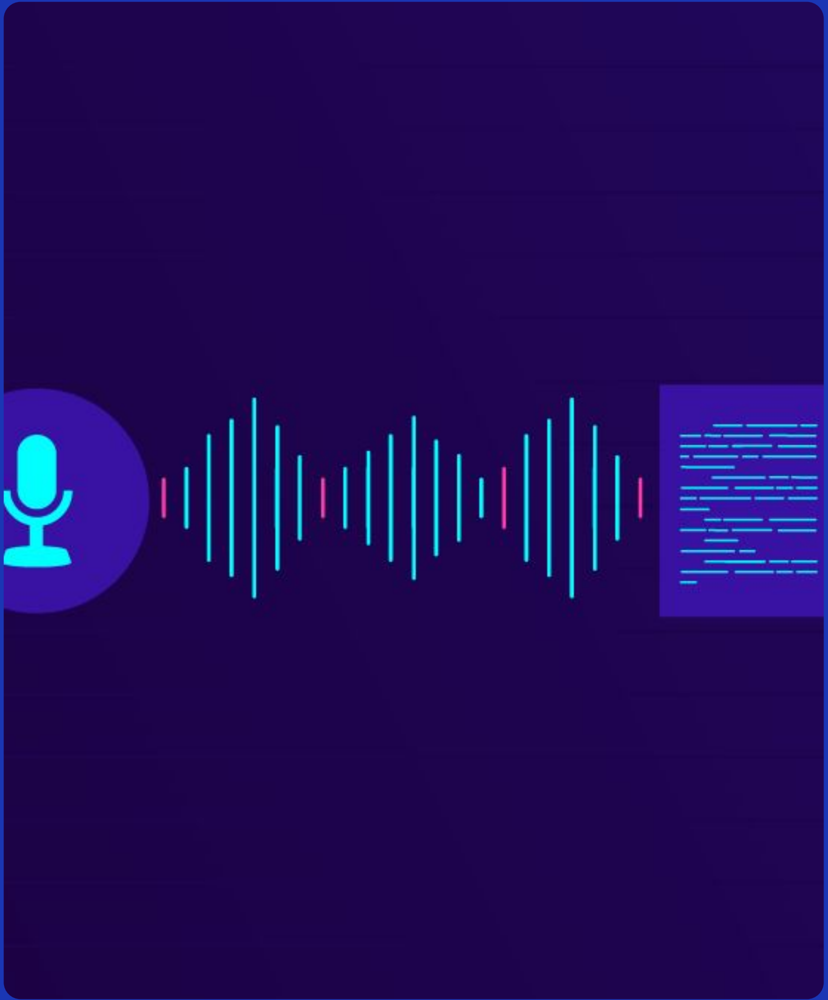
CSE445.6

AI Speech Recognition System

Mehrab-E-Rashid Khan	1931071642
Nowshin Nawal Mim	1911054042



Dataset split

Test

10%

1310 lines

Val

10%

1310 lines

Train

10%

10480 Lines

Data Acquisition:

Collected 1310 data points from Keithio.co

Model Training:

Trained the model using the training set

Hyperparameter Tuning:

Optimized hyperparameters based on the validation set performance

Model Evaluation:

Evaluated the final model's performance on the test set

Pre Processing

- First create a list of character from the alphabet and a single of quote.
- map them to numerical indices in keras layer, and specify the vocabulary of the characters used
- Create another keras layer that maps numerical indices back to character

Reading Wav File

- First we read the wav file and decode the audio to float tensor.
- Then we remove any extra channel dimension to simplify the audio signal into a mono audio
- Another Step is taken to confirm the audio data is represented in 32 bit floating point number
- A Short-time Fourier transform is performed over the audio waveform that results in a complex valued tensor representing magnitude of the frequencies across the time segment.
- Plot the waveform into a spectogram

Creating Dataset Objective

- First we define batch size to a manageable chunk that can be handled at once by the available resources. Since we are using free version of Google colab, we will use a batch size of 32 that fits perfectly with the available VRAM

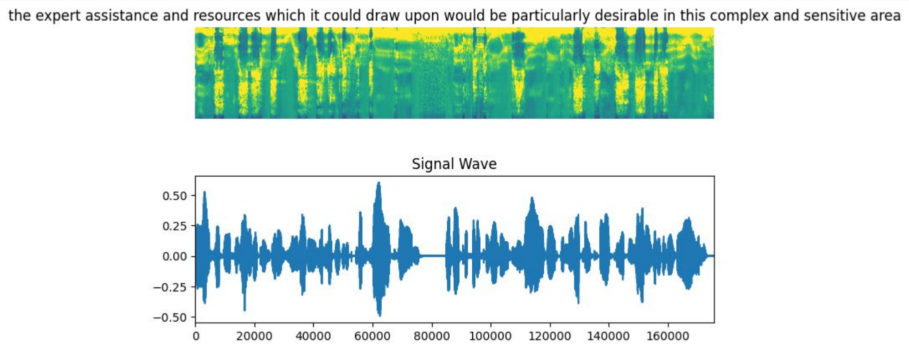
- The pre processes sample files are then made into 2 different datasets, for training and for validation

- The first element in the tuple is the name of the file and the second is the transcript of the audio file

- Then we determine the optimal number of threads to use for parallel execution of function, in hopes of speeding up the process through num_parallel_calls

Visualizing the spectrogram and corresponding audio waveform

- We take the first element of the first batch of the training dataset as an example. The tensor is converted to NumPy for further manipulation.



the expert assistance and resources which it could draw upon would be particularly desirable in this complex and sensitive area

CTCLoss Function

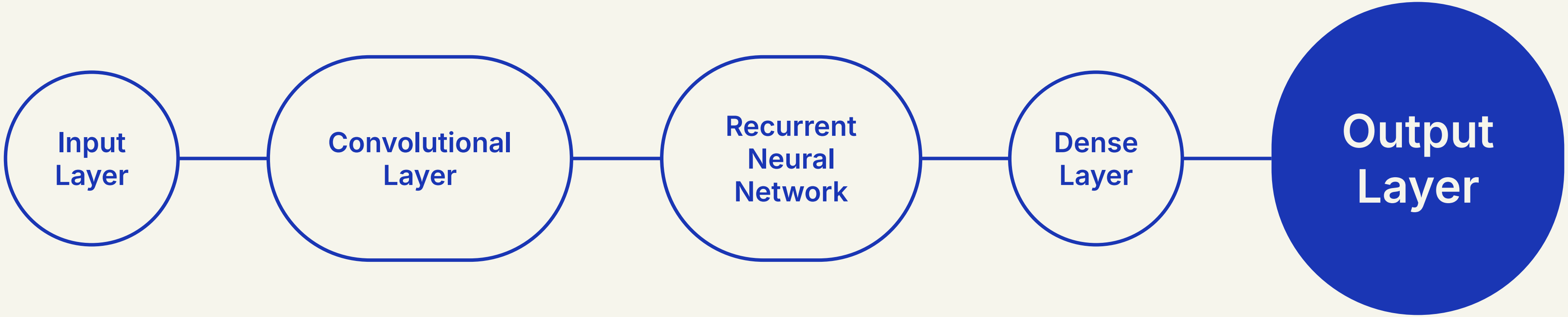
- First we define batch size to a manageable chunk that can be handled at once by the available resources. Since we are using free version of Google colab, we will use a batch size of 32 that fits perfectly with the available VRAM

- The pre processes sample files are then made into 2 different datasets, for training and for validation

- The first element in the tuple is the name of the file and the second is the transcript of the audio file

- Then we determine the optimal number of threads to use for parallel execution of function, in hopes of speeding up the process through num_parallel_calls

Model


Input Layer
Convolutional Layer
Recurrent Neural Network
Dense Layer
Output Layer

DeepSpeech2 Model

Input Layer

Takes Spectrogram As input, which is a time-frequency representation of audio

Convolutional Layer

Conv1D layer reduces the temporal dimension while increasing the number of feature maps.

This ensures that the relevant features are extracted from the spectrogram

Recurrent Neural Network

Bidirectional GRU Layers processes the input in both forward and reverse sequence thus generating a clearer image of the input sample, making it easier to come to conclusion

Dropout ensured that overfitting doesn't happen by randomly dropping out units during training

DeepSpeech2 Model

Dense Layer

This layer transforms the output from the RNN layer into a suitable representation for the final output layer

Output Layer

This layer outputs a probability distribution over the vocabulary of character, representing the likelihood of each character at each time step.

CTCLoss

Since the data we are working with is sequential, we use CTC loss function to train this model

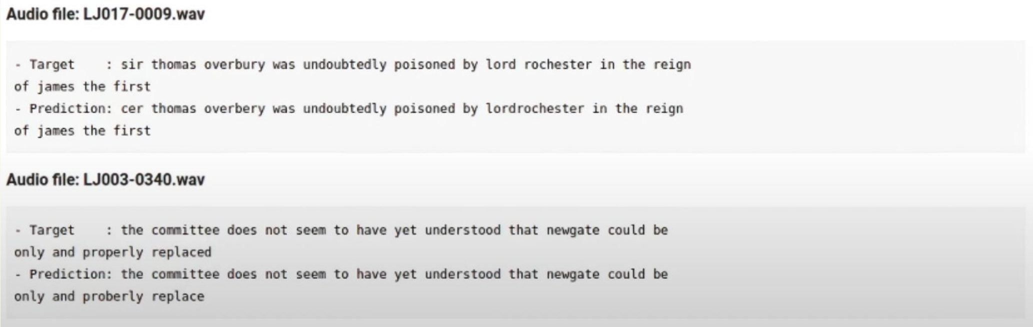
It allows for alignment-free training, making it robust to variations in speech speed and pronunciation.

Training and Result

Due to time constraints of google colab, we only had the ability to run 13 epochs before terminating the training.

We observed a steady dip in the loss value over time and expected it to be minimum after 50 epochs which could be considered

We came down to 43 loss value for training dataset and 49 loss value for validation set



Audio file: LJ017-0009.wav

```
- Target    : sir thomas overbury was undoubtedly poisoned by lord rochester in the reign
of james the first
- Prediction: cer thomas overbery was undoubtedly poisoned by lordrochester in the reign
of james the first
```

Audio file: LJ003-0340.wav

```
- Target    : the committee does not seem to have yet understood that newgate could be
only and properly replaced
- Prediction: the committee does not seem to have yet understood that newgate could be
only and proberly replace
```

Thank You