Title: Speech to Sign-Language(with emotions) for the Hearing-Impaired

Members:		
	Abhinav Goud Bingi	180050002
	Mohit Agarwala	19307R004
	Soham Naha	193079003

Project Description:

1) **Input**: Audio, Output: Emotion, Sign-language Images
2) **Evaluation Metric**: Word Error Rate for Speech2Text, and Accuracy for Emotion Detection
3) **Existing Approaches**:
    - DeepSpeech(https://github.com/mozilla/DeepSpeech),wav2letter(https://github.com/facebookresearch/wav2letter) for Speech2Text
    - https://towardsdatascience.com/speech-emotion-recognition-with-convolution-neural-network-1e6bb7130ce3 and https://github.com/MITESHPUTHRANNEU/Speech-Emotion-Analyzer for Speech to Emotion Detection.

Progress (I) -- Methodology:

We divide this project to two subtasks, one for Speech to Text and then to the corresponding sign language, and the other the Emotion Detection from the Audio.
So, for the two tasks we would use the same features and pass through two models for the subtasks.

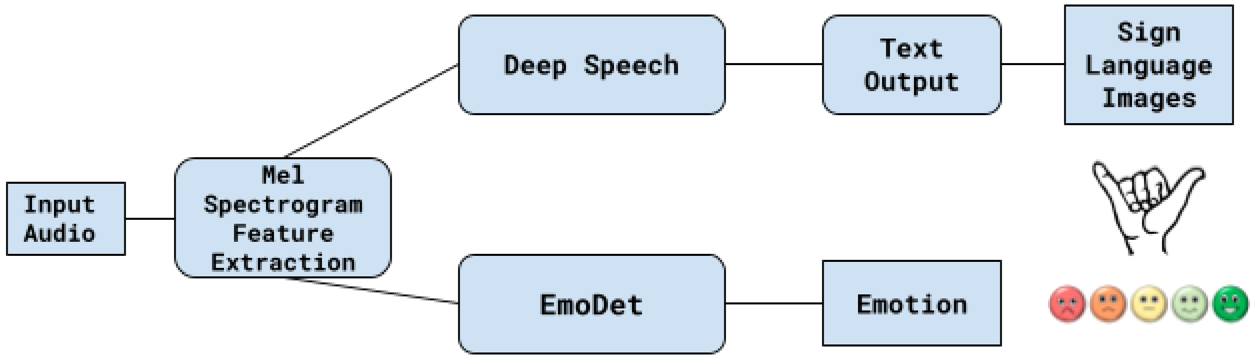

Fig. WorkFlow

We first extract the Mel Spectrogram features (128) from the Input Audio and then pass the extracted features through the models for the two subtasks.

The **DeepSpeech** model for Speech to Text task, would take into account the temporal variability of the features via Convolution Layers, followed by Bi-Directional GRUs and finally passing through a Softmax Classification layer to detect letters

The Convolution Layers are used to extract high level features from the mel-spectrogram and then pass it through the GRU layers, that help in extracting temporal relationships between the data, as it is necessary to learn the context of words and phones in detecting text utterance. The final classifier is used to detect the most probable label, using BeamDecoder from the CTCDecode library, and the criterion used for this task is the CTCLoss function.

The **EmoDet** model for Emotion Detection, the Mel Spectrogram features are passed through a Conv Layer -> ReLU -> Conv Layer -> Batch Norm -> ReLU and Dropout -> MaxPool ->
-> 3 x (Conv Layer -> ReLu) -> Conv Layer -> Batch Norm -> ReLU and Dropout -> MaxPool ->
2 x (Conv Layer -> ReLu) -> Dense Layer -> Softmax Probabilities.
We are using Conv Layers for this emotion classification, because it works best to figure out the required combinations on different levels of the Spectrogram features for robust classification as we already know the effectiveness of them in classifying data. The metric used is the accuracy score on the different categories of emotion [happy, sad, neutral, calm, angry, fearful, disgust, surprised].

<u>Progress (II) -- Implementation Details</u>:
For both the models we are using PyTorch and PyTorch Audio.

- **<u>Speech2Text</u>**:
  For the **DeepSpeech** model, we first load the Librispeech Data (*train-clean-100* for training and *test-clean* for testing), using the PyTorch DataLoader. The number of Conv Layers (with residual connection) chosen were 4 with Dropout(p=0.2) and stride=2, number of Bi-directional GRUs chosen were 3, each with hidden dimensions 384, followed by the Linear Layer with GeLU, and then the Softmax Layer with the label probabilities, the criterion used being CTCLoss.
  Most of the implementation is borrowed from https://colab.research.google.com/drive/1IPpwx4rX32rqHKpLz7dc8sOKspUa-YKO with some changes in the training routines fitting our workflow.

  But as both the model and the dataset is quite large, we haven’t been able to train the complete model yet.

- **<u>Emotion Detection</u>:**
  The implementation is done upto loading the data and extracting it’s features. We have already run the reference implementations upto completion but now we need to tweak the parameters and explore some more models(if possible) for any better performances.
  The reference implementation model details are as follows:
  (https://towardsdatascience.com/speech-emotion-recognition-with-convolution-neural-network-1e6bb7130ce3)

```
Conv1D(256, 8, padding='same',input_shape=(X_train.shape[1],1)))
Activation('relu')
```

```
Conv1D(256, 8, padding='same')
BatchNormalization()
Activation('relu')
Dropout(0.25)
MaxPooling1D(pool_size=(8))
Conv1D(128, 8, padding='same')
Activation('relu')
Conv1D(128, 8, padding='same')
Activation('relu')
Conv1D(128, 8, padding='same')
Activation('relu')
Conv1D(128, 8, padding='same')
BatchNormalization()
Activation('relu')
Dropout(0.25)
MaxPooling1D(pool_size=(8))
Conv1D(64, 8, padding='same')
Activation('relu')
Conv1D(64, 8, padding='same')
Activation('relu')
Flatten()
Dense(target_class)
Activation('softmax')
```

Progress (III) --- Experiments:

- **Speech2Text**:
  For the **DeepSpeech** implementations, we haven't yet been able to completely train the model, because of the huge number of parameters in the model and the large dataset(it takes 58+ hrs to train on Librispeech *train-clean-100*, with batch-size 32 and 100 epochs). Also Google Colab now has an additional feature where it detects the idle time, and if the idle time is greater than 3 hours, with no mouse activity on the colab notebook, it issues a checkbox asking whether I'm a bot as of March,2021. So, the previous javascript to auto-trigger the run button fails now.
  Given these constraints, we look forward to using a smaller model with lesser parameters and hopefully a smaller subset of the Librispeech data, given the time and resource constraints.
- **Emotion Detection:**
  We ran the reference implementations and saw how much they were varying and what caused the change. The differences were mainly due to some extra conv layers in one model than the other, but the model with lesser conv layers had 72% accuracy due to data leaks, meaning some of the data used in training somehow slipped into test. We need to tweak the parameters and explore some more models(if possible) for any better performances.

  For the two links provided, the accuracies were 72% and 67.5%.