TensorFlow Speech Recognition Challenge

09.04.2021

Yash Aggarwal
IIIT-Delhi
CSAI undergraduate

Overview

We have to use Speech Command Dataset to build an algorithm that understands simple spoken commands.

General Steps

Visualising Data from Audio Signal

Data visualisation was done by analysing the data and trying to find the features required for the model.

Feature Extraction from Audio Signal

We realize that our data is in the audio_file.wav format. To read the audio data, we will have to use the librosa library.

For any audio file, we will have to factor the following parameters :

a. Mel Frequency Cepstral Coefficient(MFCC)

b. Log Based Mel Spectrogram

I have trained for these features separately.

Training Model

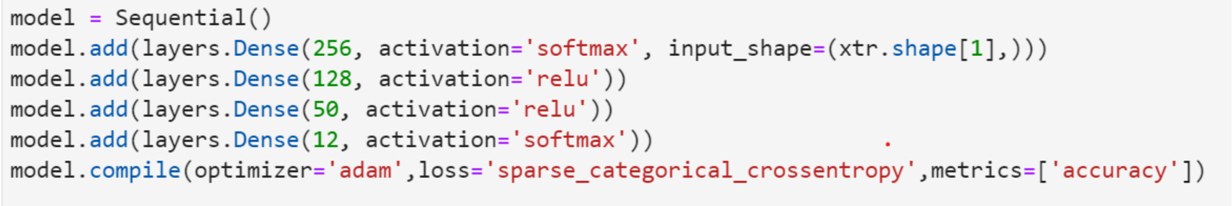
I have trained two models for this project:

1. A simple neural network (for MFCC data)
2. A convolutional neural network (for Mel Spectrogram)

In the cases of MFCC data, the following steps were followed :

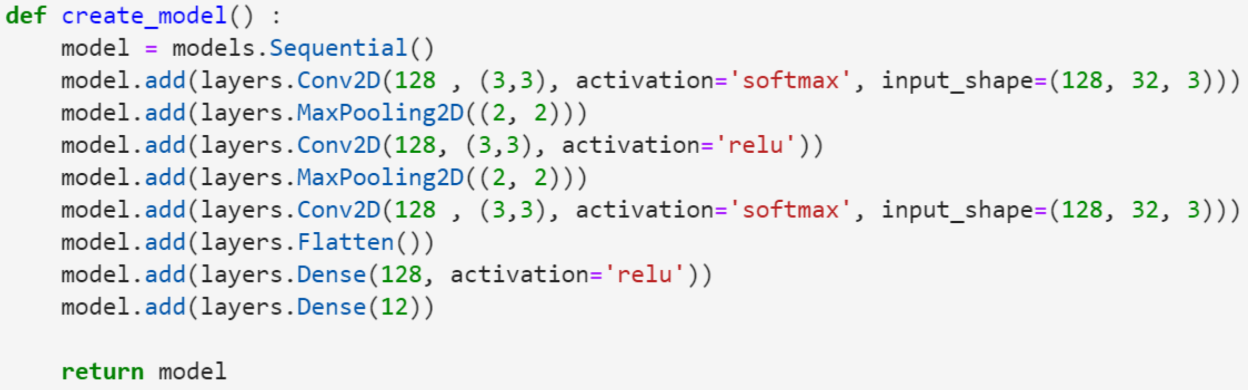
1. Finding MFCC for all audio files and storing them in a dataset.
2. Using the PCA, 12 major components were selected from these arrays.

The neural network used was as follows :



```
model = Sequential()
model.add(layers.Dense(256, activation='softmax', input_shape=(xtr.shape[1],)))
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dense(50, activation='relu'))
model.add(layers.Dense(12, activation='softmax'))
model.compile(optimizer='adam',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

In the case of Spectrogram, I generated images of size (128, 32, 3) [height, width and channel] and used them in my convolutional neural network. The model used for CNN was :



```
def create_model() :
    model = models.Sequential()
    model.add(layers.Conv2D(128 , (3,3), activation='softmax', input_shape=(128, 32, 3)))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(128, (3,3), activation='relu'))
    model.add(layers.MaxPooling2D((2, 2)))
    model.add(layers.Conv2D(128 , (3,3), activation='softmax', input_shape=(128, 32, 3)))
    model.add(layers.Flatten())
    model.add(layers.Dense(128, activation='relu'))
    model.add(layers.Dense(12))

    return model
```

Reference : https://www.tensorflow.org/tutorials/images/cnn

Challenges

The major challenges faced during the process were following:

1. Classifying “Unknown” : I decided to randomly pick some files from each of these labels and train them as “unknown”.

   Problems : The data was not very robust and did not increase accuracy.

   Other Approaches : The model could have been made better by training it with better data using techniques such as reversing words.

2. Silence : To train the silence data set, I decided to use the background noise label.

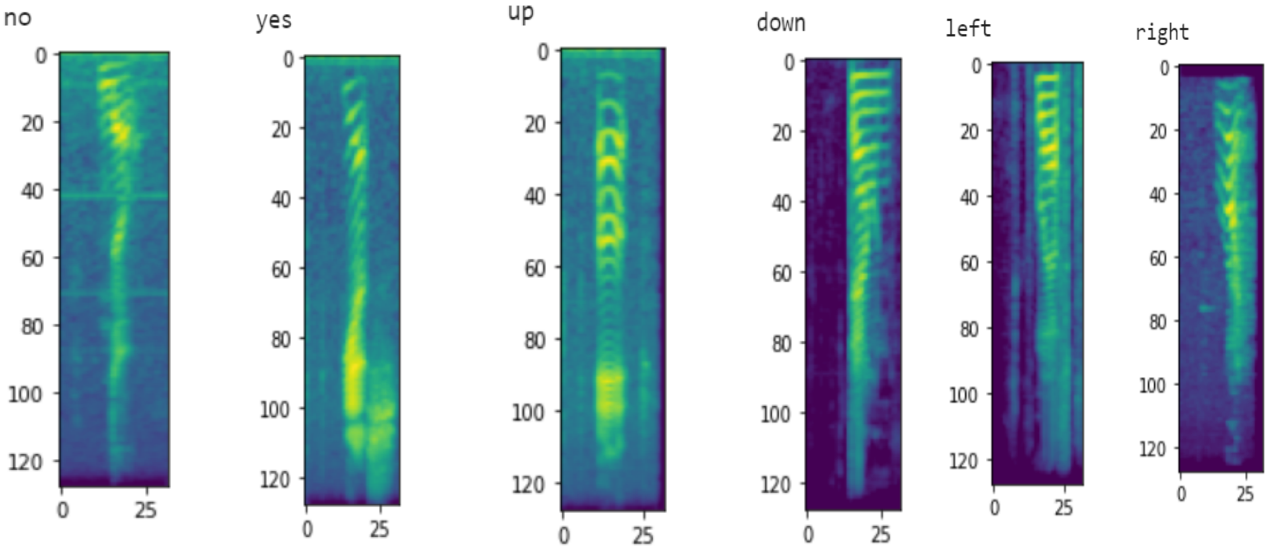
   For this I decided to split the training set of 7 files into 1 second splices to train from.

Experiments Performed

1. Finding the Log Mel Spectrogram :

While trying to find the relevant features for speech recognition, I came across log based mel spectrogram and decided to test it for this project.

The plot of spectrogram for various labels was as follows :

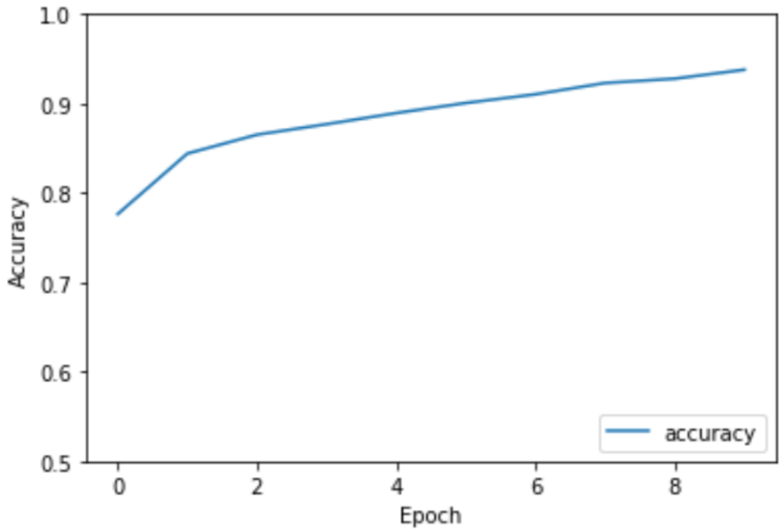


I decided to use this feature and train a 2D CNN for my model because these features looked distinguishable

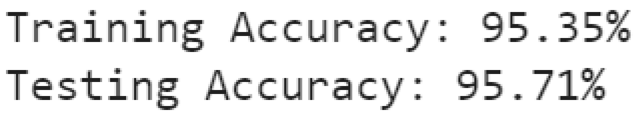
2. MFCC : The MFCC matrix did not show good results and also gave an accuracy of only 55% on the validation set and hence, was not used.

Training Model and Accuracy Observation

For the mel spectrogram model, the accuracy for the training and the validation set was observed to be the following (on a test split of 0.2):



And the accuracies were:



```
Training Accuracy: 95.35%
Testing Accuracy: 95.71%
```

Further Experiments

Due to limitation of time, I would like to perform more experiments on the data. A few of them would be :

1. Adding noise to the training data and its effect on accuracy of the model

2. Sampling the data at different frequencies and better experimentation with data augmentation (such as shifting, scaling).

References

1. https://towardsdatascience.com/constructing-manipulating-classifying-and-generating-audio-with-digital-signal-processing-and-2c5a252dbab9
2. https://www.kdnuggets.com/2020/02/audio-data-analysis-deep-learning-python-part-1.html
3. https://www.tensorflow.org/tutorials/images/cnn
4. https://www.infoq.com/presentations/dl-audio-signal/