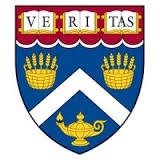
Final Project  
Identification of Foreign Language from Speech

Mr. Fakruddin Mohammed



Detailed Report

CSCI E-89 Deep Learning, Spring 2019

**Harvard University Extension School**

Prof. Zoran B. Djordjević

Contents

[1 Motivation 4](#_Toc8496547)

[2 Problem Statement 4](#_Toc8496548)

[3 Data Set 5](#_Toc8496549)

[3.1 Description 5](#_Toc8496550)

[3.2 Selected Data Set for Specific Study 5](#_Toc8496551)

[3.2.1 Preview of the Audio File 6](#_Toc8496552)

[4 Technology 7](#_Toc8496553)

[4.1 Hardware 7](#_Toc8496554)

[4.2 Software 7](#_Toc8496555)

[5 Approach 8](#_Toc8496556)

[6 Data Set Preview 8](#_Toc8496557)

[6.1 Using the Tiny Sample Data 8](#_Toc8496558)

[6.2 Step 1 – Download 9](#_Toc8496559)

[6.2.1 Download Data from Website 9](#_Toc8496560)

[6.2.2 Extract Data to Folder 9](#_Toc8496561)

[6.2.3 Extracting Selected Languages 10](#_Toc8496562)

[6.3 Spectrogram Approach 10](#_Toc8496563)

[6.3.1 Spectrogram Approach Overview 10](#_Toc8496564)

[6.3.2 Generate Spectrogram from Speech Files 11](#_Toc8496565)

[6.3.3 Run Model 12](#_Toc8496566)

[6.4 MFCC Approach 13](#_Toc8496567)

[6.4.1 Step-1: Re-structure the directory structure of data folders 13](#_Toc8496568)

[6.4.2 Step=2: Calculate MFCC coefficients from speech files 14](#_Toc8496569)

[6.4.3 Step-3: Run Model 15](#_Toc8496570)

[7 Experiments Set-1 (without Data Augmentation) 17](#_Toc8496571)

[8 Data Augmentation Techniques for Audio Files 17](#_Toc8496572)

[8.1 Motivation 17](#_Toc8496573)

[8.2 Data Augmentation Techniques 18](#_Toc8496574)

[9 Experiments 19](#_Toc8496575)

[9.1 Details 19](#_Toc8496576)

[10 Results 20](#_Toc8496577)

[10.1 Spectrogram Approach without Data Augmentation 20](#_Toc8496578)

[10.2 MFCC Approach without Data Augmentation 21](#_Toc8496579)

[10.3 Spectrogram Approach with Data Augmentation 21](#_Toc8496580)

[10.4 MFCC Approach with Data Augmentation 21](#_Toc8496581)

[10.5 Results Summary 22](#_Toc8496582)

[11 Conclusions 22](#_Toc8496583)

[12 Lessons Learnt and Next Steps 23](#_Toc8496584)

[12.1 Lessons Learned 23](#_Toc8496585)

[12.2 Next Steps 23](#_Toc8496586)

[13 References 23](#_Toc8496587)

[14 YouTube Tutorial URLs 24](#_Toc8496588)

[14.1 Summary – 2 Minutes Video 24](#_Toc8496589)

[14.2 Detailed – 15 Minutes Video 24](#_Toc8496590)

# Motivation

* Demand – for Voice and Speech Recognition is coming from every industry vertical with Automotive and Healthcare leading the market, followed by Legal, Education, Government, Military, Consumer and Retail Industries
* Regulations – such as handsfree driving
* Competition – Enterprises competing towards market edge and Global presence
* Trend – PC with Key board, Laptop with integrated key board, Palm computers with qwerty keyboard, touch screen and now the latest trend is voice recognition.
* Market Share – statistics shows the market share of global language services industry in 2018 is 45 billion and is forecasted to rise to 56.2 billion by 2021. The market share of just voice and speech recognition market size is valued at 9.12 billion in 2017.

# Problem Statement

* In my not so extensive literature survey [1][2][3][4], I found that, using the CNN model and using the data set comprises of Chinese, German, French, Spanish and English language speech files, the spoken language is successfully predicted with success rate of 87-92%. In one of the studies [3] where the focus was on just using the English & French language, the reported accuracy was 98%.
* The languages that are chosen in these studies [1][2][3][4] were disparate both in terms of geography, dialect and accent.
* Therefore, the goal of the study is to find out, if similar sounding languages in terms of geography/regional, dialect and accent are fed to the CNN model, can it predict the foreign language being spoken with similar accuracy levels that are reported in the literature.
* In this study, the chosen foreign languages are:
  + Arabic
  + Arabic Egyptian Spoken
  + Arabic Sudanese Spoken



* Using the data set available at the Top Coder website [5] and using Tensorflow/Keras API the CNN model was implemented to test the hypothesis.

# Data Set

## Description

The data set used for this study is downloaded from the TopCoder website and the link is provided below.

The characteristics of the data set is described below.

|  |  |  |
| --- | --- | --- |
| **Description** | **Training Data** | **Testing Data** |
| No of Languages | 176 | 176 (possibly) |
| No of Files | 66,176 | 12,320 |
| No of Files/Language | 376 | Unknown (official labels not made available yet) |
| File Format | MP3 | MP3 |
| Duration of Voice Recording | 10 seconds | 10 seconds |
| Output Labels | Yes | Unknown (official labels not made available yet) |
| Total Data Set Size | 4.5GB | 0.9GB |

## Selected Data Set for Specific Study

For the purpose of this study, only audio recordings for the following languages are considered: Arabic, Arabic Egyptian Spoken and Arabic Sudanese Spoken. The data set size is described below.

|  |  |
| --- | --- |
| **Description** | **Input Data** |
| No of Languages | 3 |
| No of Files/Language | 376 |
| No of Files | 1128 |
| File Format | MP3 |
| Duration of Voice Recording | 10 seconds |
| Train/Test Split | 80/20 (902/226) |
| Chosen Languages | Arabic  Arabic Egyptian  Arabic Sudanese |

### Preview of the Audio File

The intensity or frequency plot, fast fourier transform (FFT), spectrogram and mel frequency cepstral coefficients (MFCC) plots are shown below for one of the sample audio file.

|  |  |
| --- | --- |
| **Frequency/Intensity Plot** | **FFT Plot** |
| **Spectrogram Plot** | **MFCC Plot** |

#### Spectrogram

A spectrogram is a visual way of representing the signal strength or loudness of a signal over time at various frequencies present in a particular waveform. Spectrograms are basically two-dimensional graphs, with a third dimension represented by colours. Time runs from left (oldest) to right (youngest) along with the horizontal axis. The vertical axis represents frequency, which can also be thought as pitch or tone. The amplitude (or energy or loudness) of a particular frequency at a particular time is represented by the third dimension colour, with dark blueness corresponding to low amplitudes and brighter colours up through red corresponding to progressively stronger (or louder) amplitude.

#### Mel Frequency Cepstral Coefficients (MFCC)

In sound processing, the mel-frequency cepstrum (MFC) is representation of the short-term power spectrum of a sound, based on linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency.

# Technology

## Hardware

Since the size of the dataset is small, all models including the data pre-processing is carried out on personal laptop of the following configuration.

* Mac Air (2017)
* 2 CPU (quad core)
* 8GB RAM

## Software

The main backend to implement the deep learning model is Tensorflow 1.12 and along with this the following python library are user.

Python System Libraries

* os
* shutil
* glob
* pathlib

Python Audio Processing Library

* librosa

Python Data and Plotting Libraries

* pandas
* numpy
* scipy
* matplotlib
* imageio
* seaborn
* IPython
* plotly
* tqdm

Python Machine Learning Libraries

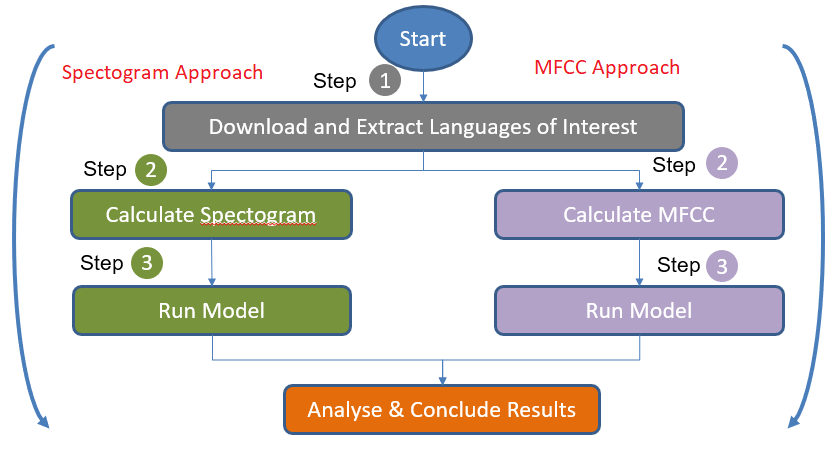
* sklearn
* keras
* skimage
* tensorflow

Among all the above library, the librosa is key python library to process the speech files and it is covered in more detail in the pre-processing section.

# Approach

The end to end approach to study the problem under investigation is presented below. As shown in the diagram, essentially, there are two techniques to address this issue.

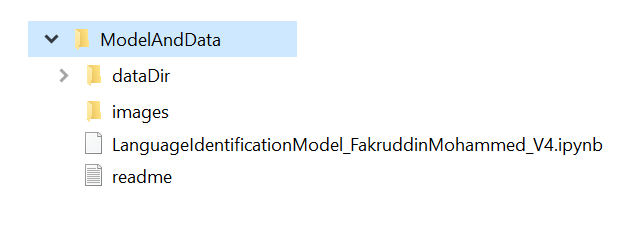
1. Spectrogram technique: which involves converting the speech file to spectrogram image, then using the standard CNN model
2. MFCC technique: which involves calculating the mel frequency cestral coefficients vestors and then using those vectors as input to CNN model.



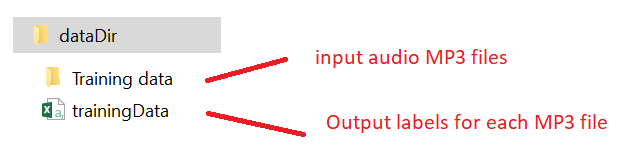
# Data Set Preview

## Using the Tiny Sample Data

* If you decided to use the tiny sample data that comes with this jupyter notebook, then simply extract the contents of the “ModelAndZip” to a folder.
* After the extract your directory structure should look like the below.



* The ‘dataDir’ folder has the input audio speech files along with the output labels.



* From within this folder, start the jupyter notebook
* Open the “LanguageIndentificationModel\_FakruddinMohammed\_V4.ipynb”
* Run all cells to run the spectrogram and MFCC approach for feature vector sizes of 32x32 for Spectrogram and 11 for MFCC.

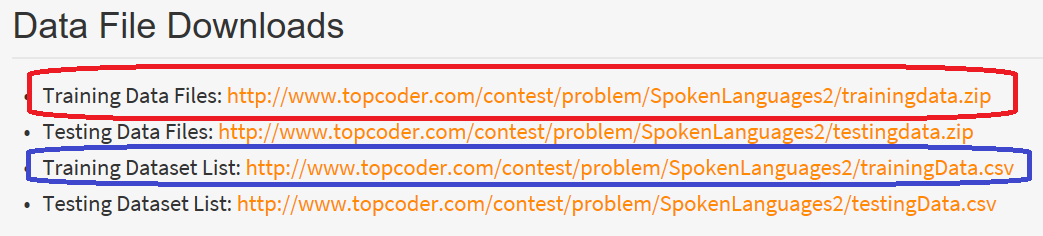
## Step 1 – Download

Instead of using the tiny sample data that comes with this jupyter if you decide to download the data from the website, then follow the steps described below

### Download Data from Website

Download the training data from the link provided below and refer to the screen print for brevity.

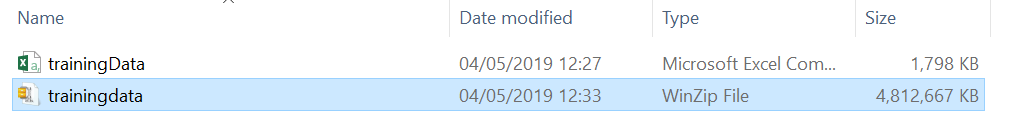
<https://community.topcoder.com/longcontest/?module=ViewProblemStatement&rd=16555&compid=49304>



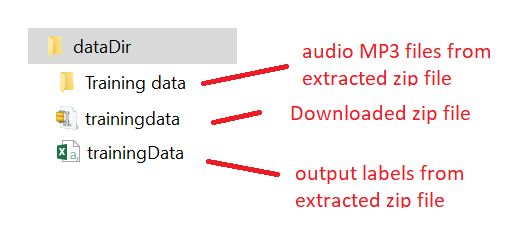
**Note:** Please note that the, test data set is unlabelled, therefore, for the purpose of this study only the training data is downloaded (1st & 3rd link above) and the training data is further split into train/test to test the model performance.

### Extract Data to Folder

* Copy the downloaded data files to a folder called ‘Data’ as shown in the figure below



* Extract the contents of trainingdata.zip into the same folder, after the extract your folder should like the following. The “Training data” has speech files and traningData.csv file has the labels for each recording.

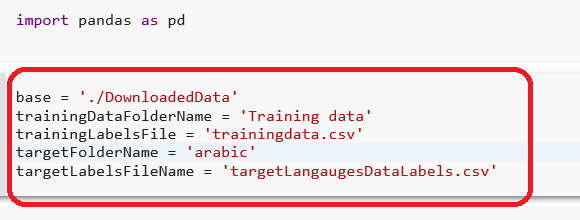


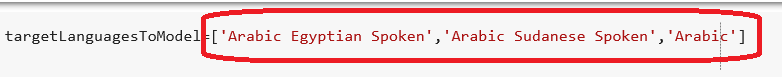
### Extracting Selected Languages

The downloaded data set has 373 languages but our study focuses on only three languages (Arabic, Arabic Egyptian and Arabic Sudanese), therefore, using the labels CSV file we will be extracting the speech files belonging to the above three languages into a new sub folder called ‘arabic’ within the same folder.



The jupyter notebook, “Step\_1:SelectLangaugeOfInterest.ipynb”automatically extracts the languages of interest into the destination folder. If you want to extract different set of languages and different folder, only amend the code as highlighted below.

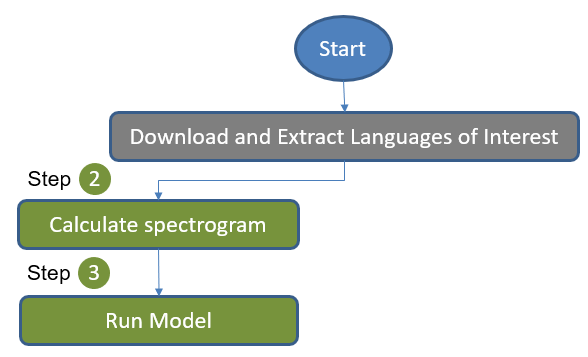




Now, the jupyter notebook, creates the required data folders and extracts the languages of interest and also creates a labels CSV file for the extracted languages. The ‘mp3’ folder will have the speech files, ‘jpg’ folder must be empty.

## Spectrogram Approach

### Spectrogram Approach Overview

spec

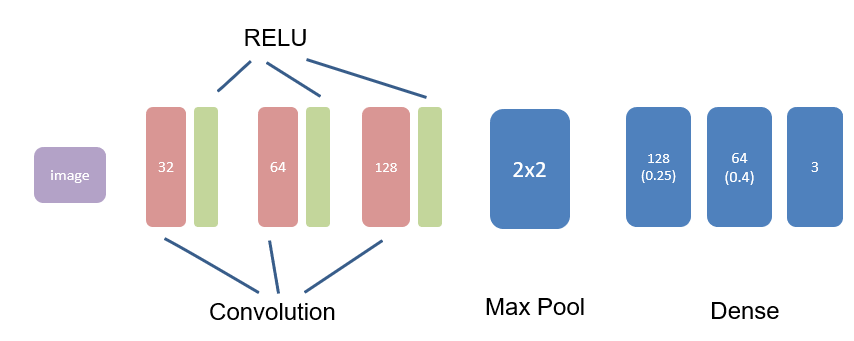
### Generate Spectrogram from Speech Files

|  |  |
| --- | --- |
| 1. The code snippet shows the process involved in generating the spectrogram images. Essentially, it loops through each MP3 recording file and then using the librosa library function ‘load’, ‘melspectrogram’.  2. Each generated image and the corresponding output label is appended into arrays. |  |
|  | 3. The labels are transformed into numerical encodings using the ‘LabelBinarizer’  4. Then the output labels encodings and image arrays are written to the numpy array files so that they can be reused again and again for experimentation. |

### Run Model

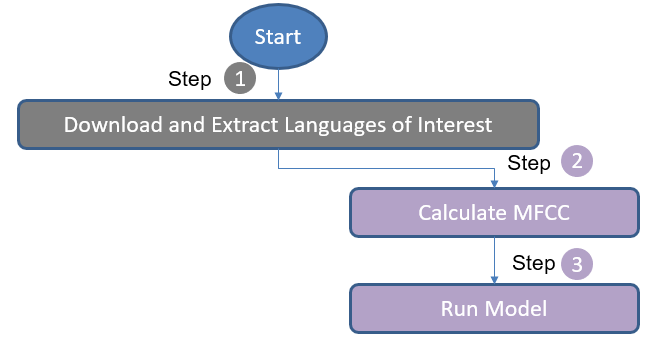
The CNN model code is implemented. The model architecture consists of:

* Three convolution layers consist of 32, 64 and 128 nodes with kernel size of 3x3
* Max pooling
* Dropout (0.25)
* Flatten
* Dense layers (128 & 64)
* Final output layer with softmax





## MFCC Approach



### Step-1: Re-structure the directory structure of data folders

The MFCC model is expecting the input data to be in the following directory structure

./data

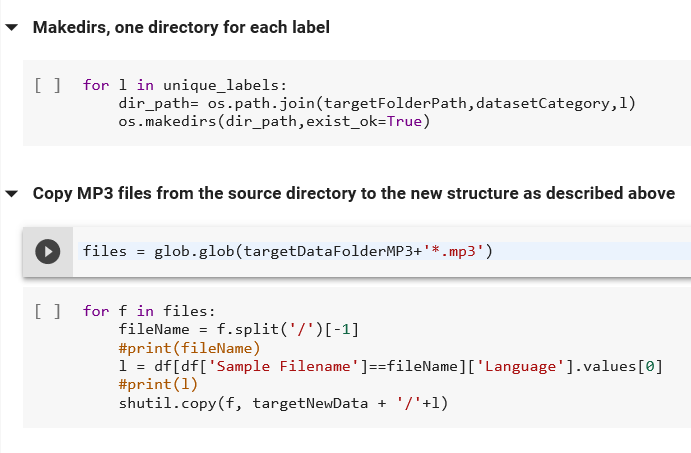
./data/label\_1

./data/label\_2

./data/label\_3



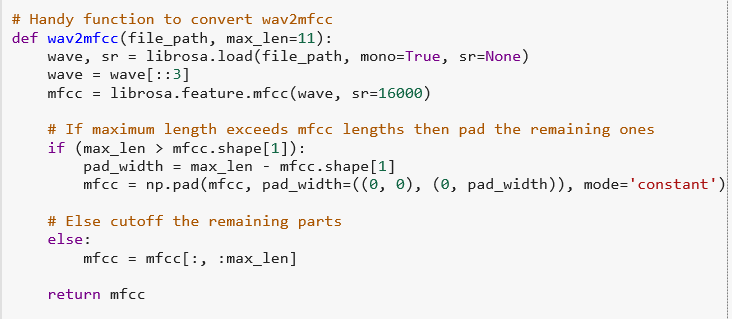
Therefore, the input data downloaded from the website needs to be rearranged in the above fashion. The code snippet below achieves this.



### Step=2: Calculate MFCC coefficients from speech files

The code snippet to calculate the MFCC is shown below. The core logic is as follows:

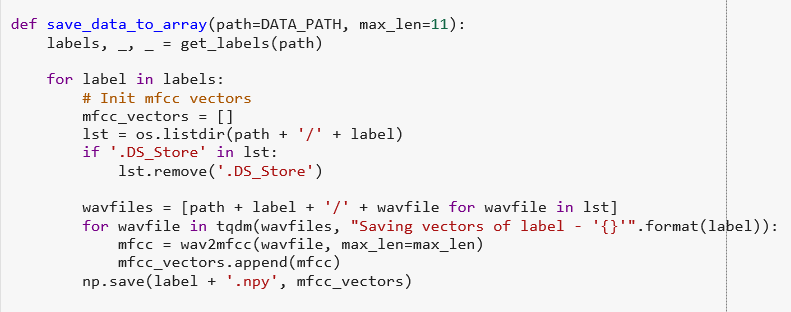
* For a given audio recording file, the code uses librosa library functions: ‘load’ to load audio recording and then using the function ‘feature.mfcc’ to get the MFCC vectors. The function below return a 2D array of specified maximum length. If none specified, then default size of 11 is assumed.



The end to end process of calculating the MFCC vestors consists of –

* Loop through each audio recording file
* Load the audio file
* Calculate the MFCC
* Append to the array
* Finally, write to numpy array file

The code snippet which does the above is shown in the Figure below.

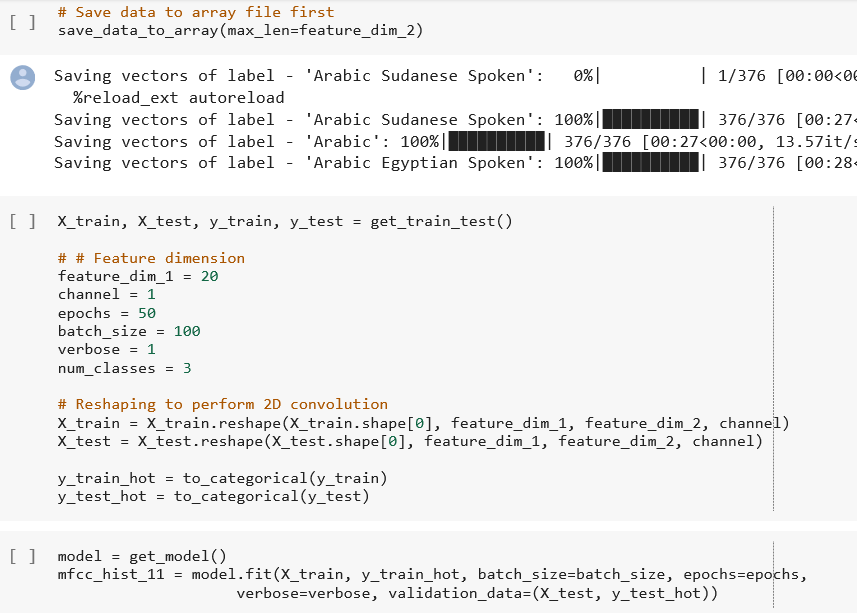


### Step-3: Run Model

Once the MFCC vectors are calculated for each audio recorded file, the rest of the process is essentially,

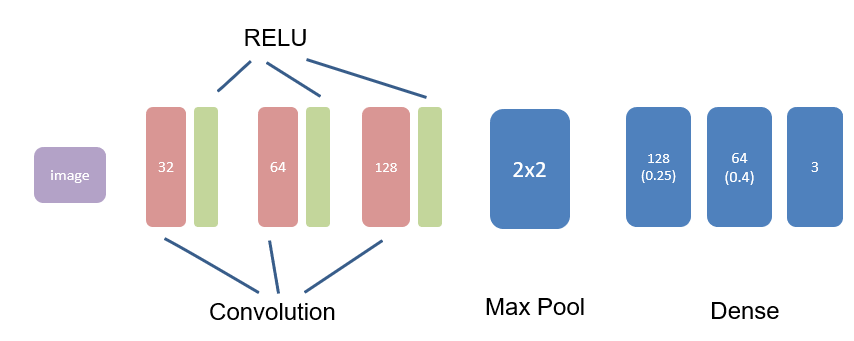
* Load the numpy array file
* Split into train/test
* Define the model
* Fit the model
* And plot the loss & accuracy

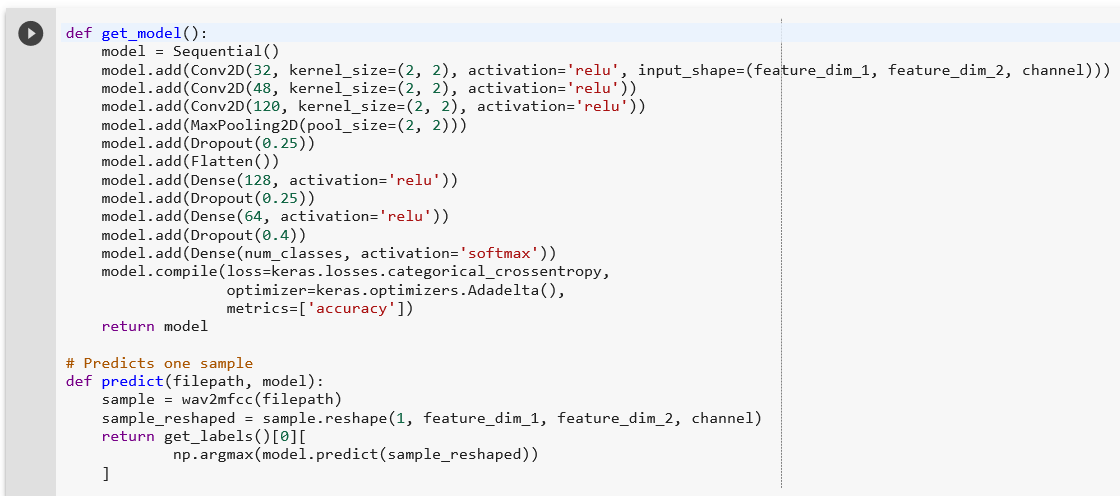
The code snippet doing the above is shown in the Figure below.



The model definition is shown below. It is essentially, a CNN model containing the following layers:

* Three convolution layers consist of 32, 48 and 120 nodes with kernel size of 2x2
* Max pooling
* Dropout (0.25)
* Flatten
* Dense layers (128 & 64)
* Final output layer with softmax





# Experiments Set-1 (without Data Augmentation)

The following set of experiments are on data without any augmentation. The results show the model accuracy was 0.7 for Spectrogram and 0.6 for MFCC. The less accuracy was probable due to very small sample of data, therefore, data augmentation techniques are explored below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Set** | **Train/Test Samples** | **Technique** | **Accuracy** |
| Without Augmentation | 902/226 | Spectrogram(32x32) | 0.7 |
| Without Augmentation | 902/226 | Spectrogram(64x64) | 0.7 |
| Without Augmentation | 902/226 | MFCC (11) | 0.6 |
| Without Augmentation | 902/226 | MFCC (24) | 0.6 |
| Without Augmentation | 902/226 | MFCC (32) | 0.6 |

# Data Augmentation Techniques for Audio Files

## Motivation

Since the number of samples for each language are only 376 and with these samples the model is predicting the language with accuracy of only 60%. Therefore, the data augmentation techniques listed below are used to generate more samples. By using these techniques, number of samples are increased from 376 to 1880 files (4 times the original dataset).

* Adding noise
* Shifting or rolling the audio
* Increasing the pitch
* Decreasing the pitch

## Data Augmentation Techniques

A class is created called ‘AudioAugmentation’ and the code snippet of this class is shown below. The usage of this class is described below.

aa = AudioAugmentation()

data = aa.read\_audio\_file(one\_sample\_file)

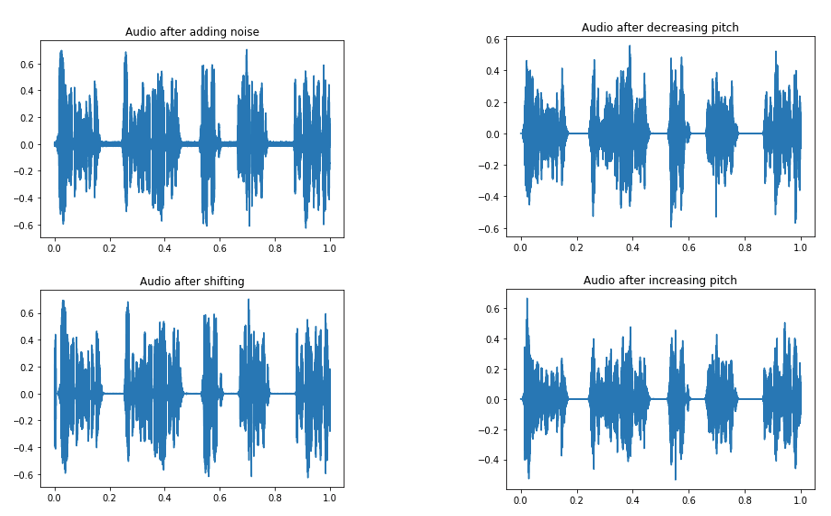
aa.plot\_time\_series(data)

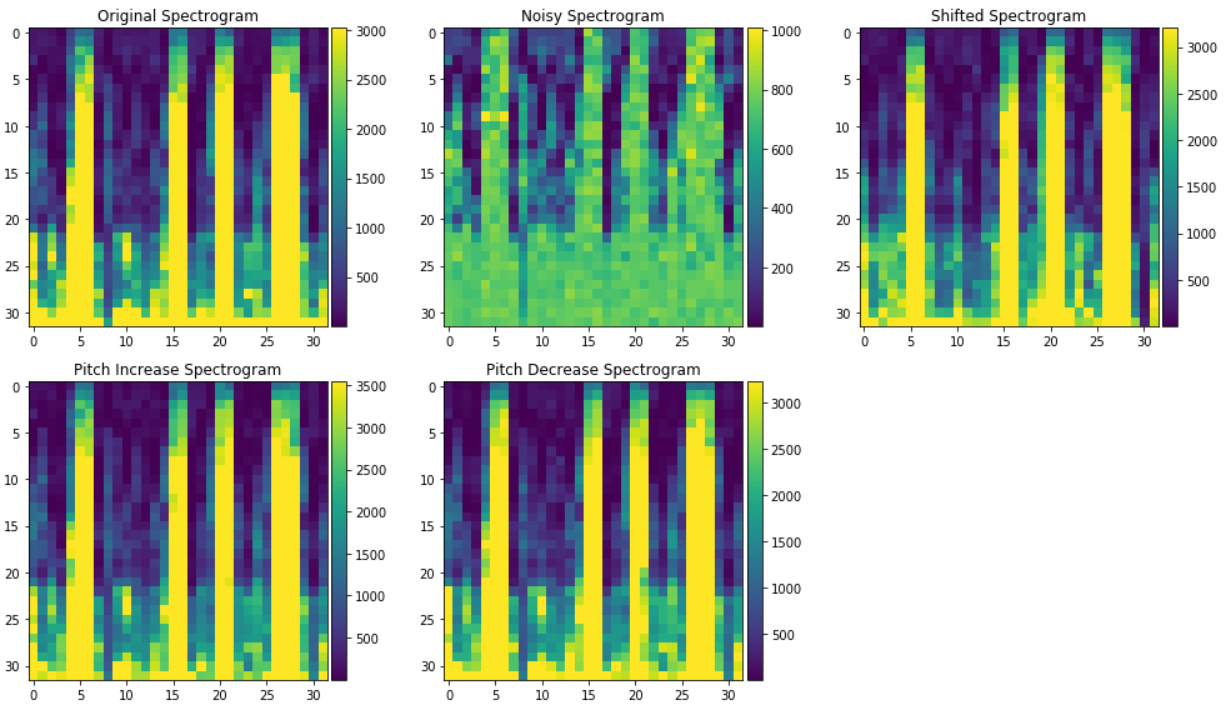
data\_noise = aa.add\_noise(data)

aa.plot\_time\_series(data\_noise)



The intensity plots and spectrogram plots for one of the sample audio recording file is shown below.





# Experiments

## Details

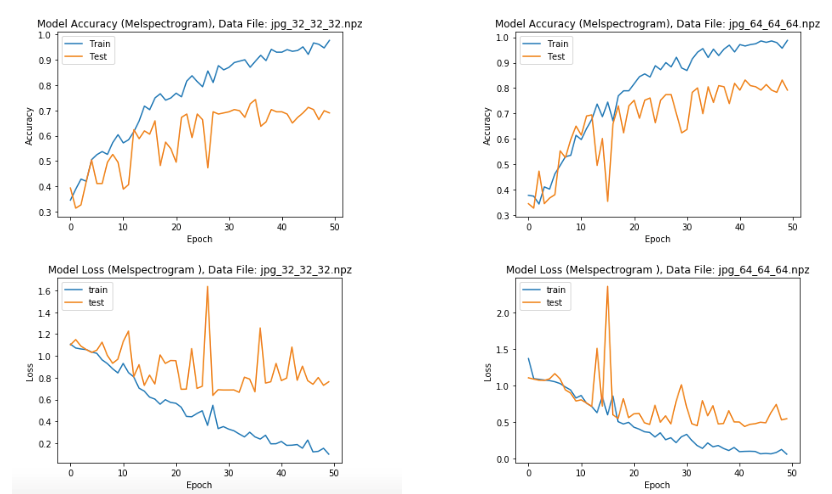
To validate the hypothesis, the following 8 experiments are conducted: 4 experiments using the Melspectrogram and 4 experiments using the MFCC vectors.

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Set** | **Train/Test Samples** | **Technique** | **Accuracy** |
| Without Augmentation | 902/226 | Spectrogram(32x32) | 0.7 |
| Without Augmentation | 902/226 | Spectrogram(64x64) | 0.7 |
| Without Augmentation | 902/226 | MFCC (11) | 0.6 |
| Without Augmentation | 902/226 | MFCC (24) | 0.6 |
| Without Augmentation | 902/226 | MFCC (32) | 0.6 |
| With Augmentation | 4512/1128 | Spectrogram(32x32) | 0.9 |
| With Augmentation | 4512/1128 | Spectrogram(64x64) | 0.9 |
| With Augmentation | 4512/1128 | MFCC (11) | 0.8 |
| With Augmentation | 4512/1128 | MFCC (24) | 0.8 |
| With Augmentation | 4512/1128 | MFCC (32) | 0.8 |

# Results

## Spectrogram Approach without Data Augmentation

The CNN model was run on the original data set of size 1128 input samples for the spectrogram sizes of 32x32 and 64x64; the loss & accuracy plots are presented below. As the plots shows, the accuracy of the model is around with 70% and the model starts overfitting from around 15 epochs.



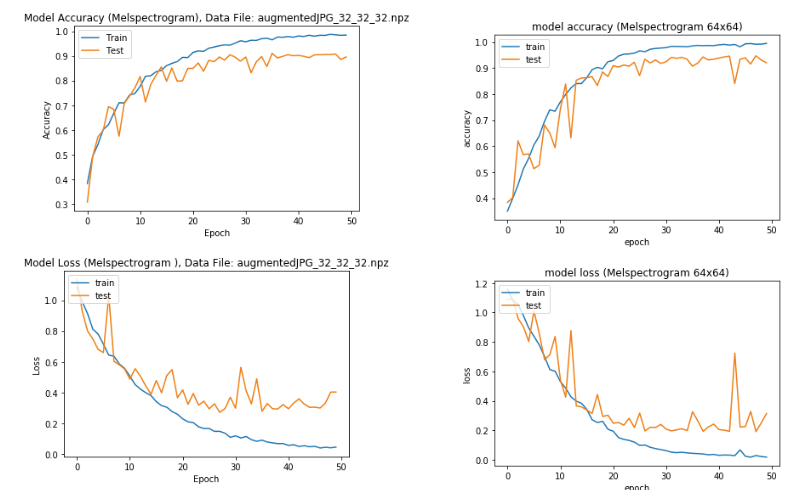
## MFCC Approach without Data Augmentation

The CNN model was run on the original data set of size 1128 input samples for the MFCC vector sizes of 11, 24 and 32. The loss & accuracy plots are presented below. As the plots shows, the accuracy of the model is around with 70% and the model starts overfitting from around 20 epochs.



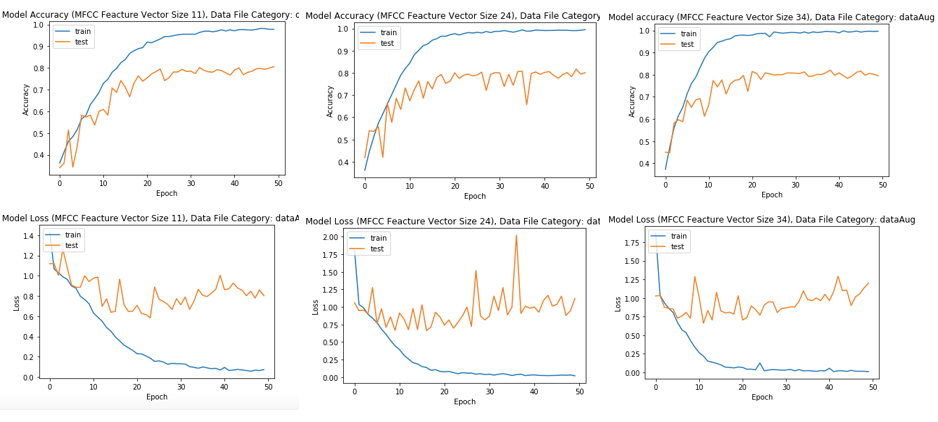
## Spectrogram Approach with Data Augmentation

The CNN model was run on the original data set of size 5640 input samples for the spectrogram sizes of 32x32 and 64x64; the loss & accuracy plots are presented below. As the plots shows, the accuracy of the model is around with 70% and the model starts overfitting from around 15 epochs.



## MFCC Approach with Data Augmentation

The CNN model was run on the original data set of size 5640 input samples for the MFCC vector sizes of 11, 24 and 32. The loss & accuracy plots are presented below. As the plots shows, the accuracy of the model is around with 70% and the model starts overfitting from around 20 epochs.



## Results Summary

|  |  |  |
| --- | --- | --- |
| **Technique** | **Accuracy without Data Aug**  **(Train/Test: 902/226)** | **Accuracy with Data Aug.**  **(Train/Test: 4512/1128)** |
| Spectrogram 32x32 | 0.7 | 0.9 |
| Spectrogram 64x64 | 0.7 | 0.9 |
| MFCC Vector Size: 11 | 0.6 | 0.8 |
| MFCC Vector Size: 24 | 0.6 | 0.8 |
| MFCC Vector Size: 32 | 0.6 | 0.8 |

# Conclusions

* The CNN model found to be effective in predicting the foreign language from speech file with accuracies of around 90% even if the input audio data is similar in terms of dialect and accuracy.
* The results show foreign language identification using the Spectrogram technique is much more effective than MFCC approach
* As the sample size increases, increasing the size of the spectrogram or MFCC vector sizes had a very little impact on improving the model accuracy
* When the input samples are limited, then relying on data augmentation techniques is extremely useful and boost the model accuracy by 10%
* The CNN model found to be quite useful in identifying the foreign language from the audio speech files with accuracies of around 90%

# Lessons Learnt and Next Steps

## Lessons Learned

* The input sample size for the selected three language are 1128 files and total size of 88M. When the sample size increased with data augmentation to 5640 files and the total size of the data was 4.8GB. I had significant memory issues during the model execution, therefore, I converted each audio recording file to numpy array (spectrogram) and wrote to the disk. This approach significantly reduced the memory requirements from 4.8GB to 46MB.

## Next Steps

* Even with the data augmentation, the accuracy of the model is around 91%. Therefore, I want to explore the other architectures of CNN like increasing the number of layers and with varied sizes of nodes and kernel sizes.
* Experiment with data augmentation techniques to generate more samples, to check if the accuracy limitations are due to sample size.
* Test the model with k-fold cross validation, to test the model stability in predicting the language.
* Explore the language prediction using RNN/LSTM model, as the meaning/interpretation of the words being spoken is dependent on the contact and RNN models are best suited to time series.
* It appears the recordings are done in the controlled environment as the speaker voice is monotone not reflecting the ordinary speech or not reflecting how the languages are actually spoken. Therefore, I would like to audio recordings from variety of sources and test the models.
* The literature survey reveals that, in addition to MFCC, using their 1st and 2nd derivatives as feature vectors improves the model performance. Therefore, I would like to explore this idea further.
* I would like to conduct the language identification model testing on the whole data set of 373 languages with 66176 files to check if the model performance stays at 90% accuracy or deteriorates when the number of output class labels increases.

# References

* [1] Language Detection from Speech: Chinese or English?, Tianlong Song Tags Machine Learning Natural Language Processing; Sun 15 Oct 2017 ; https://stlong0521.github.io/20171015%20-%20Language%20Detection.html; https://github.com/stlong0521/language-detector
* [2] Automatic spoken language identification (English, German, Spanish, French) identification using voxfrge dataset,Thomas Werkmeister; https://github.com/twerkmeister/iLID
* [3] Spoken Language Classification, (German, Italian, Engligh and French) Julien De Mori, Misrab Faizullah-Khan, Cameron Holt, Shahriyar Pruisken; Autumn 1012; http://cs229.stanford.edu/proj2012/DeMoriFaizullahKhanHoltPruisken-SpokenLanguageClassification.pdf
* [4] Practical Application of Multimedia Retrieval, (English, German, Spanish, French), Tom Herold, Thomas Werkmeister, https://github.com/twerkmeister/iLID/blob/master/Deep%20Audio%20Paper%20Thomas%20Werkmeister%2C%20Tom%20Herold.pdf Data set: https://community.topcoder.com/longcontest/?module=ViewProblemStatement&rd=16555&compid=49304
* [5] Data augmentation of audio data: https://www.kaggle.com/CVxTz/audio-data-augmentation
* [6] A Review of Audio Features and Statistical Models Exploited for Voice Pattern Design, Ngoc Q. K. Duong and Hien-Thanh Duong, https://arxiv.org/abs/1502.06811, 24th Feb 2015. Giannakopoulos T (2015) pyAudioAnalysis: An Open-Source Python Library for Audio Signal Analysis. PLoS ONE 10(12): e0144610. doi:10.1371/ journal.pone.0144610
* [7] Deep Learning Lectures by Prof. Zoran B Djordjevic, Harvard University Extension School.

# YouTube Tutorial URLs

## Summary – 2 Minutes Video

<https://youtu.be/8jNt1EIKpxc>

## Detailed – 15 Minutes Video

<https://youtu.be/FFq7wMZ5fnY>