Deep Learning For Audio

Sai Priyamka Kotha [1], Sravani Nallagari [1], and Jinan Fiaidhi [1]

[1]Affiliation not available

October 30, 2023

Abstract

Speech is the most efficient and convenient way of communication. The learning capabilities of the deep learning architecture can be used to develop the sound classification system to overcome the efficiency issues of the traditional systems. We propose to develop a model that classifies the audio of the speaker.

Deep Learning For Audio

Guided By: Dr.Fiadhi

Kotha Sai Priyanka- 1111984
Masters in Computer Science
Lakehead University
skotha@lakeheadu.ca

Nallagari Sravani- 1108001
Masters in Computer Science
Lakehead University
snallaga@lakeheadu.ca

***Abstract*—Speech is the most efficient and convenient way of communication. The learning capabilities of the deep learning architecture can be used to develop the sound classification system to overcome the efficiency issues of the traditional systems. We propose to develop a model that classifies the audio of the speaker.**

***Index Terms*—Deep learning, Audio recognition, Neural networks, Accuracy.**

I. Introduction

Waveform is a general way to represent an audio signal. Signal Processing is the art and science of changing the data obtained from the time series for analysis or enhancement purposes. Audio signals are the three-dimensional signals that represent time, amplitude and frequency. Audio wave

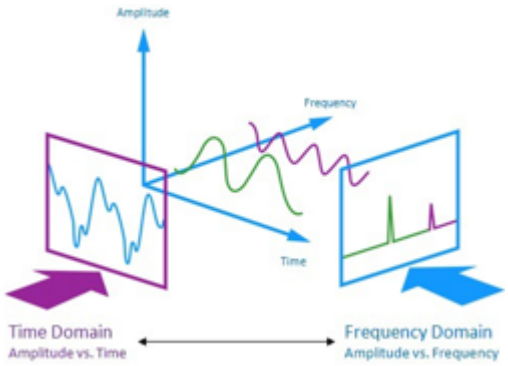

Fig. 1. Audio Signal.

Sampling: Sampling is a method of converting an analogue audio signal into a digital signal. While sampling a sound wave, the computer takes measurement of this sound wave at a regular interval called sampling interval. Each measurement is then saved as a number in binary format.

Speech recognition can be done at a sampling rate of 16 kHz (16000 times per second).

As the neural networks will have to do the speech classification, it is very important to feed the network inputs with relevant data. An appropriate preprocessing is necessary in order to be sure that the input of the neural networks is

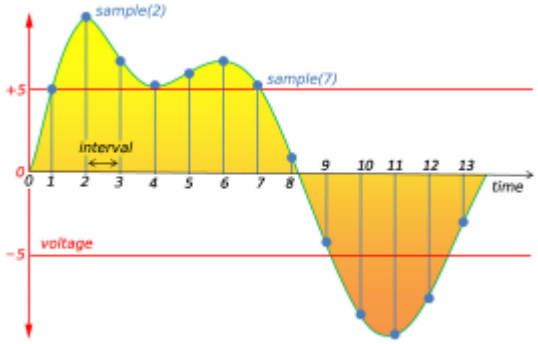

Fig. 2. Audio Sampling- converting wave to numbers.

characteristic for every word, while having a small spread amongst samples of the same word. Noise and difference in amplitude of the signal can distort the integrity of a word while timing variations can cause a large spread amongst samples of the same word. All these problems are dealt with signal processing.

For example, we have Alexa, a smart speaker that recognises the voice of a particular speaker and responds to the voice commands. Similarly, we have siri in iphone. We can use this sound classification model for biometric authentication purposes in service centres.

II. Literature Survey

We have to transform the data to features, which can then be fed into the algorithm. Most important features that are used to build the model for our audio classification in our project are:

RMSE (Root Mean Square Error)

Energy: The energy of a signal corresponds to the overall signal amplitude and is approximately associated with how intense the signal is.

n—x(n)—2 describes the energy of the signal. RMSE is the root mean square of this formulation.

Zero Crossing Rate

By looking at different speech and audio waveforms, we can

see that depending on the content, they vary a lot in their smoothness. For example, voiced speech sounds are more smooth than unvoiced ones.
To calculate the zero-crossing rate of a signal, we need to compare the sign of each pair of consecutive samples. In other words, for a length N signal, you need 0(n) operations. Such calculations are also extremely simple to implement, which makes zero crossing rate an attractive measure for low complexity applications.

Spectral Centroid
Spectral features are the frequency-based features, which are obtained by converting the time signal into the frequency domain. In the context of speech recognition or classification of a speaker, frequency information can be represented in the form of spectral centroids.

Spectral roll off
It is the frequency below which lies a given percentage of total spectral energy e.g, 85%

Chroma
Chroma refers to 12 different pitch classes. It is a 12-element feature vector indicating how much energy is present in the signal of each pitch class.

MFCC (Mel-frequency Cepstral Coefficients).
The idea of MFCC is to compress information about the vocal tract i.e. smoothed spectrum into a small number of coefficients. The mel-frequency cepstral coefficients(MFCC) of a signal are a small set of features usually about 10 to 20, which describes the overall shape of the spectral envelope.

By using fourier transforms, we split the complex sound wave into simple sound waves and then adds up to sum of energy.

III. Model

We are extracting the audio features from the audio files. Then we are preprocessing the data. Then, we are applying the neural networks and predicting the speaker. Finally, we are calculating how accurately our model is predicting the speaker.
In this, we have 5 steps.
Step 1: Audio extraction features.
Step 2: Preprocess and transform the dataset
Step 3: Apply the neural network.
Step 4: Prediction
Step 5: Calculate the accuracy.

We have already collected the audio samples and converted them into csv files to extract information from the files.

A. Audio extraction features

Librosa is a python package used to analyse music and audio. It provides the building blocks required for the development of music knowledge recovery systems. We use Librosa's in-built functions such as mfcc, chroma, etc which generate an MFCC from audio data from the time series. All the extracted features are kept in the csv file.

B. Preprocess and transform the dataset

The filenameArray contains all audio files. For speakers a different array is produced. The respective speakers are given a weight. For example, if the speaker name begins with j, the speaker is assigned a "0." We add the respective speaker's audio file to the assigned number from the list. We remove the filename, number, chroma stft and unnecessary columns.

We have a total of 1500 recordings at the repository. We split the dataset into training and validation sets using sklearn.model selection.train test split.70% of the dataset is used for training the model and the remaining 30% is used for validation. There are total 1050 recordings in training data and validation data has 450 recordings.

Data Normalization:
Data normalization gets rid of many irregularities which can hinder the data analysis. The information inside the data can typically be structured in such a way that it can be visualised and analysed by data normalisation.
We are using sklearn StandardScaler for normalizing the data. It transforms the data, so it has mean as 0 and standard deviation as 1. In short, the data is standardised. Standardization is useful for negative-value results. The data is structured in a typical standard distribution. Classification is more useful.
Each value in the dataset is subtracted by the mean and divided by the standard deviation.

C. Apply the Neural Network

CNN: Convolutional Neural Networks (CNNs) is the most popular neural network model being used for image classification problem. Instead of a fully connected network of weights from each, a CNN has just enough weights to look at a small patch. The main advantage of CNN compared to its predecessors is that it automatically detects the important features without any human supervision.

There are total 13 layers and we are using the dense layer and the dropout layer, where the dense layer uses the back-propaganda function. Dropout is used to make the model to forget each time after it learns, then overfitting might happen which can be prevented.
So, softmax is used as the classifier classification as we have multiple classes so when we pass through the model it will show the probability distribution.
Here, we are getting ready the model then we should pass the data. The optimizer is the adam and the loss function is the sparce _categorical_crossentropy. In model.fit we will pass

the data to the model with epochs 50 and batch size 128.

LSTM: Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning.

These networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series.

First, we are taking the embedding matrix with size of 5000 and dropout of 0.2.**For all the models the dropout is 0.2 because to create a uniform standard among all the model's foe comparison we can change if we want to increase, he accuracy of the model.**

Here, the max pooling is 2 so we will generate a 2*2 matrix. We are using LSM here and in dense layer we are using softmax for classification. We are training with early stopping to avoid the overfitting, epochs=50 and batch size=128.

MLP: Multilayer perceptrons are often applied to supervised learning problems. They train on a set of input-output pairs and learn to model the correlation (or dependencies) between those inputs and outputs. Training involves adjusting the parameters, or the weights and biases, of the model in order to minimize error.
We are using two layers i.e dense layer followed by relu for activation and for the classification we are using softmax. The optimizer is "adam"and the loss is "sparce_categorical_crossentropy". Training by early stopping to avoid overfitting epochs=50 and batch size=128.

RNN: A recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. An RNN remembers each information through time. It is useful in time series prediction only because of the feature to remember previous inputs as well. This is called Long Short-Term Memory.

First layer is Embedding layer which is similar to LSTM, but here zfirst we are taking the embedding matrix and we are passing data and multiplying.

Second, we are using LSTM, the differences between this LSTM in RNN and actual LSTM is the recurrent drop for back propagation. After that we used "relu" and "dropout" layer. softmax for classification with epochs 50 and batch size=128.

D. Implementation

We are extracting the features of the Wav files, save them into CSV files and store them into Pandas All the features are stored in train.csv file. Then this file is sent to the neural netowrks model as an input.

```
def extractWavFeatures(soundFilesFolder,
    csvFileName):

   header = 'filename chroma_stft rmse
       spectral_centroid spectral_bandwidth
       rolloff zero_crossing_rate'
   for i in range(1, 21):
      header += f' mfcc{i}'
   header += ' label'
   header = header.split()
   print('CSV Header: ', header)
   file = open(csvFileName, 'w', newline='')
   #with file:
   writer = csv.writer(file)
   writer.writerow(header)
   genres = '1 2 3 4 5 6 7 8 9 0'.split()
   for filename in
       os.listdir(soundFilesFolder):
      number =
          f'{soundFilesFolder}/{filename}'
      y, sr = librosa.load(number, mono=True,
          duration=30)
      # remove leading and trailing silence
      y, index = librosa.effects.trim(y)
      chroma_stft =
          librosa.feature.chroma_stft(y=y,
          sr=sr)
      rmse = librosa.feature.rms(y=y)
      spec_cent =
          librosa.feature.spectral_centroid(y=y,
          sr=sr)
      spec_bw =
          librosa.feature.spectral_bandwidth(y=y,
          sr=sr)
      rolloff =
          librosa.feature.spectral_rolloff(y=y,
          sr=sr)
      zcr =
          librosa.feature.zero_crossing_rate(y)
      mfcc = librosa.feature.mfcc(y=y, sr=sr)
      to_append = f'{filename}
          {np.mean(chroma_stft)}
          {np.mean(rmse)}
          {np.mean(spec_cent)}
          {np.mean(spec_bw)}
          {np.mean(rolloff)} {np.mean(zcr)}'
      for e in mfcc:
         to_append += f' {np.mean(e)}'
      writer.writerow(to_append.split())
   file.close()
```

Pre-processing the data by assigning values to each speaker and dropping unnecessary columns.

```
def preProcessData(csvFileName):
   print(csvFileName+ " will be preprocessed")
   data = pd.read_csv(csvFileName)
   # we have six speakers:
   # 0: Jason
   # 1: Nickel
   # 2: Teyami
   # 3: Ahil
   # 4: Cade
   # 5: Richard
   filenameArray = data['filename']
```

```
    speakerArray = []
    #print(filenameArray)
    for i in range(len(filenameArray)):
        speaker = filenameArray[i][2]
        #print(speaker)
        if speaker == "j":
            speaker = "0"
        elif speaker == "n":
            speaker = "1"
        elif speaker == "t":
            speaker = "2"
        elif speaker == "a":
            speaker = "3"
        elif speaker == "c":
            speaker = "4"
        elif speaker == "r":
            speaker = "5"
        else:
            speaker = "6"
        #print(speaker)
        speakerArray.append(speaker)
    data['number'] = speakerArray
```

Sample code for implementing the model to predict the speaker.

```
model = Sequential()
model.add(layers.Dense(1024,
    activation='relu',
    input_shape=(X_train.shape[1],)))
model.add(Dense(10, activation='softmax'))

model.compile(optimizer='adam',
          loss='sparse_categorical_crossentropy',
          metrics=['accuracy'])

# simple early stopping
from keras.callbacks import EarlyStopping

es = EarlyStopping(monitor='val_loss',
    mode='min', verbose=1)

#Train with early stopping to avoid
    overfitting
history = model.fit(X_train,
               y_train,
               validation_data=(X_val, y_val),
               epochs=50,
               batch_size=128,
              )
```

E. Prediction

We have plotted the graph for the training vs testing loss.

The prediction and ground truth is calculated where x-axis is prediction and y-axis is ground turth which refers to the accuracy of the training set's classification for supervised learning techniques. This is used in statistical models to prove or disprove research hypotheses.

CNN Model

The Accuracy obtained from the CNN model is 97.11%. The prediction and ground truth for the CNN model is calculated and shown in the below table.

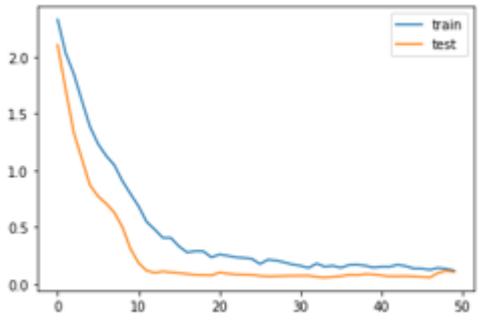

Fig. 3. CNN graph loss

	prediction	ground_truth
1116	1	1
1368	0	0
422	2	2
413	2	2
451	0	0

Fig. 4. Ground Turth for CNN

LSTM Model

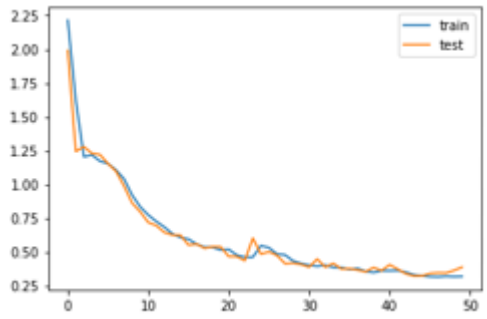

Fig. 5. LSTM graph loss

The Accuracy obtained from the LSTM model is 85.56%. The prediction and ground truth for the LSTM model is calculated and shown in the below table.

MLP Model

The Accuracy obtained from the MLP model is 99.11%. The prediction and ground truth for the MLP model is calculated and shown in below table

The MLP has more accurate prediction and ground truth values when compared to other models.

RNN Model

The Accuracy obtained from the RNN model is 82.89%. The prediction and ground truth for the RNN model is calculated

	prediction	ground_truth
1116	1	1
1368	0	0
422	2	2
413	2	2
451	0	0

Fig. 6. Ground Turth for LSTM

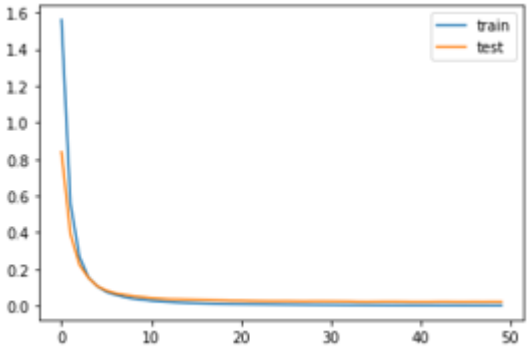

Fig. 7. MLP graph loss

F. Calculate the Accuracy

Finally, we have plotted a graph to compare all the models, in which MLP has the highest accuracy.

IV. Conclusion

In this project the heighest accuracy of identifying a particular speaker is 99.11% by using multilayer perceptron which is a class of feedforward artificial neural network.

The learning capabilities of the deep learning network model architecture was more accurate in the sound classification of the particular speaker to overcome the efficiency issues of the traditional systems.

	prediction	ground_truth
1116	1	1
1368	0	0
422	2	2
413	2	2
451	0	0
...	...	...
479	0	0
426	2	2
123	2	2
1169	2	2
394	2	2

100 rows × 2 columns

Fig. 8. Ground Turth for MLP

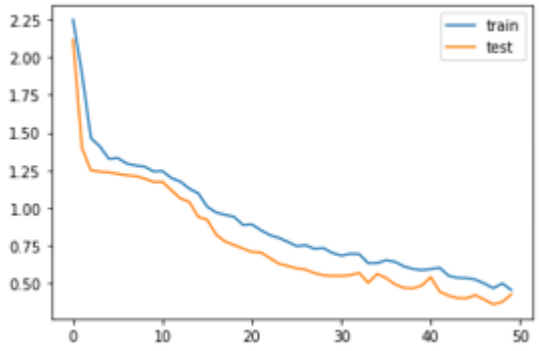

Fig. 9. RNN graph loss

	prediction	ground_truth
1116	1	1
1368	1	0
422	0	2
413	0	2
451	0	0
...	...	...
479	0	0
426	2	2
123	2	2
1169	2	2
394	0	2

100 rows × 2 columns

Fig. 10. Ground Turth for RNN

V. References

1. L. Rabiner, B.H. Juang, Fundamentals of Speech Recognition, Prentice Hall, 1993.

2. Md. R. Hasan, M. Jamil, Md. G. Rabbani, Md. S. Rahman, ”Speaker Identification using Mel Frequency

Fig. 11. Accuracy comparision Garph

Cepstral Coefficients," Third International Conference on Electrical Computer Engineering ICECE, Dhaka, 2004.

3. O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," ICASSP, 2012.

4. O. Abdel-Hamid, L. Deng, and D. Yu. "Exploring convolutional neural network structures and optimization for speech recognition," Interspeech, 2013, submitted.

5. M. C. Anzalone, L. Calandruccio, K. A. Doherty, L. H. Carney, "Determination of the potential benefit of time-frequency gain manipulation", Ear Hearing, vol. 27, pp. 480-492, 2006.

6. W. Yutai, J. Xiaoqing, L. Feng,"Speaker Recognition Based on Dynamic MFCC Parameters, "School of Information Science and Engineering, University of Jina , 2002.

7.W.Han, C.F. Chan, C.S. Choy and K.P. Pun, "An Efficient MFCC Extraction Method in Speech Recognition," Department of Electronic Engineering, The Chinese University of Hong Kong, Hong, IEEE - ISCAS, 2006.

8.V. Tiwari, " MFCC and its applications in speaker recognition", International Journal on Emerging Technologies",vol.1, pp.19-22,Feb.2010.

9.S.H. Chen and Y.R. Luo, "Speaker Verification Using MFCC and Support Vector Machine", Proceedings of the International Multi Conference of Engineers and Computer Scientists Vol.1 IMECS 2009, March 18-20, 2009, Hong Kong.