Language Diarization

EC498: Major Project Report

Submitted by

Anupama Kolsur(191EC107)

Hritika Bhavsar(191EC209)

Y.S.Ch. Gowtham (191EC158)

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

In

Electronics and Communication Engineering

Under the guidance of

Dr. Deepu Vijayasenan

Department of Electronics and Communication Engineering

National Institute of Technology Surathkal

Mangalore, Karnataka, India - 575025

April 2023

DECLARATION

We hereby declare that the Project entitled **Language Diarization** which is being submitted to **National Institute of Technology Karnataka, Surathkal** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Electronics and Communication Engineering** is a bonafide report of the project work carried out by me. The material contained in this Project has not been submitted to any other university or institution for the award of any degree.

Anupama Kolsur
Reg. No:191EC107

Hritika Bhavsar
Reg. No:191EC209

Y.S.Ch. Gowtham
Reg. No:191EC158

Department of Electronics and Communication Engineering

Place: NITK, Surathkal.
Date: 2023

CERTIFICATE

This is to certify that the Major Project entitled **Language Diarization** submitted by **Anupama Kolsur**(Roll No:191EC107), **Hritika Bhavsar**(Roll No:191EC209), **Y.S.Ch.Gowtham,** (Reg. No:191EC158) as the record of the project work carried out by him, is accepted as the Project Report submission in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology**.

Dr. Deepu Vijayasenan
Project Guide
Assistant Professor
Dept. of E & C Engineering
NITK Surathkal

Dr. N S V Shet
Head of the Department
Professor
Dept. of E & C Engineering
NITK Surathkal

Acknowledgement

We would like to express our sincere gratitude to our esteemed project guide, Dr. Deepu Vijayasenan, Associate Professor, Department of Electronics and Communication Engineering, National Institute of Technology Surathkal, for his expert guidance and invaluable suggestions. We are thankful to Dr. N S V Shet, Head, Department of Electronics and Communication Engineering, National Institute of Technology Surathkal, for the provision of industry-level resources and for encouraging informative learning.

Anupama Kolsur(191EC107)
Hritika Bhavsar(191EC209)
Y.S.Ch. Gowtham (191EC158)

Abstract

While the age of the internet booms, and there are terabytes of data that are being transferred on a daily basis, it has become very important to be able to perform analysis on these data. One of the main problems that researchers face is to analyse audio. We need to successfully convert an audio to text and the machines can analyse the same. While this may sound easy, there are cases, where there are multiple speakers and the machines have to separate the speakers, to be able to make logical conclusions. There have been many studies on the same topic, while many conform to language diarization, we have taken a step further to inculcate language diarization into the same.

Contents

List of Figures 9
List of Tables 9
1 **INTRODUCTION** 10
1.1 Overview 10
1.2 Speaker diarization 11
1.3 Problem statement 11
1.4 Problem Objectives 11
2 **LITERATURE SURVEY** 12
2.1 Literature Review 12
2.2 Dataset 13
3 **METHODOLOGY** 13
3.1 Language Detection 15
3.2 Collecting embeddings. 15
3.3 Processing Embeddings. 15
3.4 Extracting subsegments 15
3.5 Clustering 16
4 **ANALYSIS AND RESULTS** 17
5 **CONCLUSION** 19
6 **FUTURE SCOPE** 20
Bibliography 20

List of Tables

1. Literature review. 12
2. CNN + LSTM Model Architecture 14

List of figures

1. Embeddings extraction. 15
2. Model training 17
3. Training and Validation accuracy curve for 150 epochs 18
4. Training and validation loss curves for 150 epochs. 18
5. Clustering subsegments 19

Chapter 1: INTRODUCTION

1.1 Overview

Spoken language diarization (SLD) is a task to perform automatic segmentation and labelling of the languages present in a given code-switched speech utterance. Inspiring from the way humans perform SLD (i.e capturing the language specific long term information), this work has proposed an acoustic-phonetic approach to perform SLD. This acoustic phonetic approach consists of an attention based neural network modelling to capture the language specific information and a Gaussian smoothing approach to locate the language changepoints. From the experimental study, it has been observed that the proposed approach performs better when dealing with code-switched segments containing monolingual segments of longer duration. However, the performance of the approach decreases with decrease in the monolingual segment duration. This issue poses a challenge in the further exploration of the proposed approach

The preliminary purpose of the project is to create a model to separate audio input according to the speaker's identity and the language they are speaking. Spontaneous language switches in a single conversation, also known as code-switching (CS), are prominent in multilingual societies. The impact of CS and other kinds of language switches on speech-to-text systems has recently received research interest and has led to several robust acoustic modelling for multilingual speech data.

1.2 Speaker diarization

Speaker diarisation (or diarization) is the process of partitioning an audio stream containing human speech into homogeneous segments according to the identity of each speaker. It can enhance the readability of an automatic speech transcription by structuring the audio stream into speaker turns and, when used together with speaker recognition systems, by providing the speaker's true identity. Speaker diarization is a combination of speaker segmentation and speaker clustering. The first aims at finding speaker change points in an audio stream. The second aims at grouping together speech segments on the basis of speaker characteristics.

1.3 Problem statement

To create a model that separates audio input according to the speaker's identity and the language they are speaking.
Implement Spoken Language Diarization and label the monolingual segments, present in a given multilingual speech segment. Apply the models for the majority of Indian language speech data available and evaluate and analyse the contribution of different languages in day-to-day conversations.

1.4 Problem Objectives

The main objectives of this project are divided into sub-parts as follows:
(i) Language detection for small-length audio clips and collecting embeddings.
(ii) Form Similarity matrices by using different techniques on those embeddings.
(iii) Then we get the results by forming clusters.

Chapter 2: LITERATURE SURVEY

2.1 Literature Review

Four research works were deeply studied. The detailed analysis is tabulated below

Author and Title of Paper	Summary	Advantages	Limitations
Snyder, David & Garcia-Romero, Daniel & McCree, Alan & Sell, Gregory & Povey, Daniel & Khudanpur, Sanjeev. (2018). **Spoken Language Recognition using X-vectors.** 105-111. 10.21437/Odyssey.2018-15.	Feed forward DNN to extract variable length speech segments known as x-vectors and then training the x vectors using GMM	Use of x-vectors instead of mfcc features which tend to match the trend of training	Hefty data augmentation
D. -C. Lyu, E. -S. Chng and H. Li, **"Language diarization for code-switch conversational speech,"** 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 2013, pp. 7314-7318, doi: 10.1109/ICASSP.2013.6639083.	Language diarization to separate two languages on code switch utterances	Implementation using several models viz. GMM4096+SVM, P-PRLM, PR+CRF	Emphasis on diarization of only two languages
E. Yilmaz, M. McLaren, H. van den Heuvel and D. A. van Leeuwen, **"Language diarization for semi-supervised bilingual acoustic model training,"** 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), Okinawa, Japan, 2017, pp. 91-96, doi: 10.1109/ASRU.2017.8268921.	Implementation of automatic Language Identification for Automatic Speech Recognition of CS Speech	Used Gaussian Mixed Models (GMM - HMM) applying Hamming windowing with frame length of 25ms and frame shift of 10ms	Less amount of dataset used
J. Mishra, A. Agarwal and S. R. M. Prasanna, **"Spoken Language Diarization Using an Attention based Neural Network,"** 2021 National Conference on Communications (NCC), Kanpur, India, 2021, pp. 1-6, doi: 10.1109/NCC52529.2021.9530035.	Acoustic- phonetic approach for Spoken Language Diarization	Great accuracy of both training and testing(99.82% and 98.4%) using attention based neural network	The SLD is implemented on larger blocks and deters in performance for testing with smaller blocks (<200ms)

Table 1: Literature review

Dataset

The dataset we intend to use is VoxlingUA107, with a training set of over 6628 hours over 107 languages. The dataset consists of short speech segments automatically extracted from YouTube videos and labelled according to the language of the video title and description, with some post-processing steps to filter out false positives. It is curated using extraction of audio data from YouTube videos that are retrieved using language-specific search phrases (random phrases from Wikipedia of the particular language).

Chapter 3: METHODOLOGY

3.1 Language Detection

- A dataset consisting of audio files of 8 languages was used to build a model to detect the languages spoken.
- 1000 files were taken from each language and feature extraction was done by extracting mfcc(13 features), delta, double delta, as features from the audio files and concatenating them and padding them.
- Initially, a Feed-forward neural network was used consisting of 4 dense layers. After training for 50 epochs, an accuracy of **86%** and a validation accuracy of **51.25%** was achieved.
- Using a Convolutional neural network consisting of 4 convolutional layers and 3 pooling layers, after training for 30 epochs, an accuracy of **90%** and a validation accuracy **75%** was achieved.
- Finally a model consisting of 4 convolutional layers followed by an LSTM layer was built. After training for 80 epochs, a final accuracy of **97.78%** and a validation accuracy of **91.75%** was achieved.
- The same model, when trained for 150 epochs for 16 languages, gave an accuracy of **99.29%** and a validation accuracy **87.69%**.
- The final CNN+LSTM model reduced the overfitting as compared to the previous models as the difference between the accuracy and validation accuracy is reduced significantly.

The model architecture is tabulated below:

Layer(type)	**Output shape**	**Total trainable parameters**
Convolution (2D)	(None, 27,860,16)	160
Max Pooling (2D)	(None,18,430,16)	0
Convolution (2D)	(None,16,428,32)	4640
Max Pooling (2D)	(None,8,214,64)	0
Dropout (0.2)	(None,8,214,32)	0
Convolution (2D)	(None,6,212,64)	18496
Max Pooling (2D)	(None,3,106,64)	0
Convolution	(None,1,104,128)	73856
Dropout (0.2)	(None,1,104,128)	0
Reshape	(None,104,128)	0
LSTM	(None,500)	1258000
Dense	(None,5)	2505
Total		1,357,657

Table 2: CNN+LSTM Model Architecture

This is a Keras model definition in Python for a convolutional neural network (CNN) followed by a long short-term memory (LSTM) model for sequence processing, with a final dense layer with softmax activation for classification. Here's a brief overview of the layers:

Conv2D: This layer applies convolutional filters to the input data, with the number of filters and their sizes specified by the 16 and (3,3) arguments, respectively. The activation argument specifies the activation function to apply after the convolution. The input_shape argument specifies the shape of the input data, which is assumed to be a 3D tensor with the first dimension being the number of samples, the second and third dimensions being the width and height of each sample, and the final dimension being the number of channels (e.g., RGB color channels).

MaxPooling2D: This layer performs max pooling on the output of the previous convolutional layer, with the pooling window size specified by (2,2). Max pooling reduces the spatial dimensions of the data while preserving the most important features.

The pattern of Conv2D and MaxPooling2D layers is repeated two more times, with increasing numbers of filters, to extract higher-level features from the data.
Reshape: This layer reshapes the output of the final convolutional layer into a 2D tensor with shape (batch_size, 103, 128), where batch_size is the number of samples in each batch.
LSTM: This layer applies an LSTM model to the input sequence of shape (batch_size, 103, 128), with 500 units in the LSTM layer.
Dense: This layer is a fully connected layer with softmax activation, which is used for multi-class classification. The number of units in the layer is specified by encodedOutputLabelList.shape[1], which is assumed to be the number of classes in the classification task.

3.2 Collect embeddings

```
from keras.models import Model
from keras.models import load_model
import os
os.chdir('/content/drive/My Drive/Major Project/Feature Files')
model = load_model("trained_model.h5")
embedding_model = Model(inputs=model.input, outputs=model.layers[-2].output)
speech_embeddings = embedding_model.predict(x_test)
np.shape(speech_embeddings)
```

Figure 1: Embedding extraction

The code creates a new model called "embedding_model" that takes the same inputs as the loaded model but only outputs the activations of the second-to-last layer. This is done using the Model class from Keras, which allows you to specify the inputs and outputs of a model.

The predict() function is then used to obtain the embeddings of the test data (x_test) using the embedding_model. The resulting embeddings are stored in the "speech_embeddings" variable.

3.3 Processing Embeddings

This part extracts embeddings from an audio file using a pre-trained model.

The script reads the wave file using the "soundfile" library, and the segments file using the "kaldi_io" library. It then extracts embeddings using the above Language detection model. The embeddings are stored in the "embeddings" folder within the specified "out_path".

The script takes as input several arguments, including the path to the pre-trained model, the path to the input audio file, the path to the output file where the embeddings will be saved, and several other parameters such as the batch size and whether or not to use a GPU.

The script first loads the pre-trained model and sets it to evaluation mode. Then, it reads the audio file and segments it into smaller chunks. These chunks are then processed by the pre-trained model to produce embeddings. The embeddings are then saved to the output file.

3.4 Extracting subsegments

This is the part that takes an input segments file, and creates a subsegments file from it. The output file contains subsegments that are created by dividing the input segments into smaller segments. The output format is: <subsegment-id> <utterance-id> <start-time> <end-time> where the timings are relative to the start-time of the <utterance-id> in the input segments file.

The script uses the argparse module to parse command line arguments. The argparse module makes it easy to write user-friendly command-line interfaces. The script takes the following command-line arguments:

--max-segment-duration: Maximum duration of the subsegments (in seconds)

--overlap-duration: Overlap between adjacent segments (in seconds)

--max-remaining-duration: Segment is not split if the left-over duration is more than this many seconds

--constant-duration: Final segment is given a start time max-segment-duration before the end to force a constant segment duration. This overrides the max-remaining-duration parameter

--vad_segments_file: Input kaldi segments file

--out_subsegments_file: Output subsegments file

The script reads the input segments file line by line, and creates subsegments from each segment based on the specified parameters. It writes the subsegments to the output file.

3.5 Clustering

Then we further implement Agglomerative Hierarchical Clustering (AHC) using the embeddings. AHC is a method for clustering hierarchical clusters. First, the cosine similarity matrix is computed from the embeddings, and then the matrix is converted into a distance matrix. Then, a linkage matrix is computed using the fastcluster package, and finally, clusters are formed using a threshold derived from the two-Gaussian GMM calibration.

Here's a brief summary of each function:

twoGMMcalib_lin(s, niters=10): This function trains a two-Gaussian GMM (Gaussian Mixture Model) with shared variance for calibration of scores s. It returns a threshold for

original scores s that "separates" the two Gaussians and an array of linearly calibrated log odds ratio scores.

cos_similarity(x): This function computes a cosine similarity matrix in a CPU & memory-sensitive way. It takes embeddings as input, which is a 2D array where embeddings are in rows, and returns a cosine similarity matrix.

AHC(x): This function performs hierarchical agglomerative clustering on the embeddings x using cosine similarity as a distance metric. It returns the cluster labels.

Chapter 4: Analysis of Results

Training the model:

```
In [28]: 1 history = model3.fit(x_train, y_train, epochs = 150, batch_size = 500, validation_data=(x_test, y_test))
0.8841
Epoch 145/150
26/26 [==============================] - 305s 12s/step - loss: 0.0229 - accuracy: 0.9935 - val_loss: 0.5973 - val_accuracy:
0.8850
Epoch 146/150
26/26 [==============================] - 306s 12s/step - loss: 0.0372 - accuracy: 0.9880 - val_loss: 0.6965 - val_accuracy:
0.8628
Epoch 147/150
26/26 [==============================] - 311s 12s/step - loss: 0.0412 - accuracy: 0.9874 - val_loss: 0.6247 - val_accuracy:
0.8834
Epoch 148/150
26/26 [==============================] - 317s 12s/step - loss: 0.0269 - accuracy: 0.9932 - val_loss: 0.5852 - val_accuracy:
0.8844
Epoch 149/150
26/26 [==============================] - 301s 12s/step - loss: 0.0253 - accuracy: 0.9931 - val_loss: 0.6059 - val_accuracy:
0.8816
Epoch 150/150
26/26 [==============================] - 302s 12s/step - loss: 0.0250 - accuracy: 0.9929 - val_loss: 0.6193 - val_accuracy:
0.8769
```

Figure 2: Model training

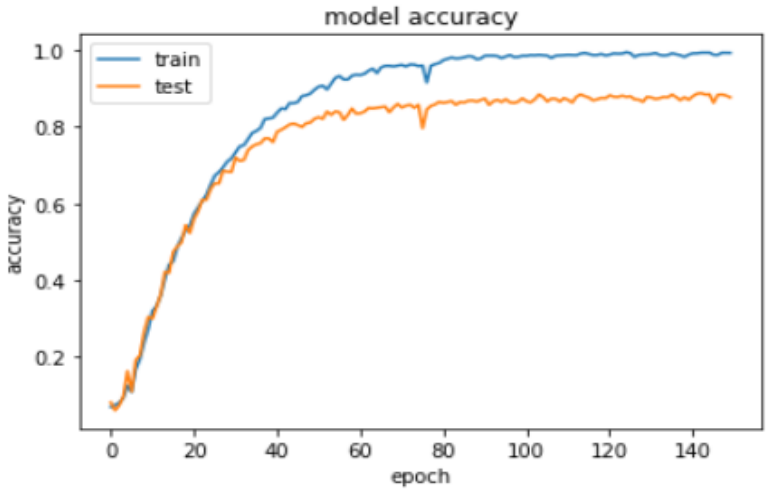

Figure 3: Training and validation accuracy curve of model for 150 epochs

The curve has a steady growth rate with very less overfitting and gets saturated after a particular number of epochs

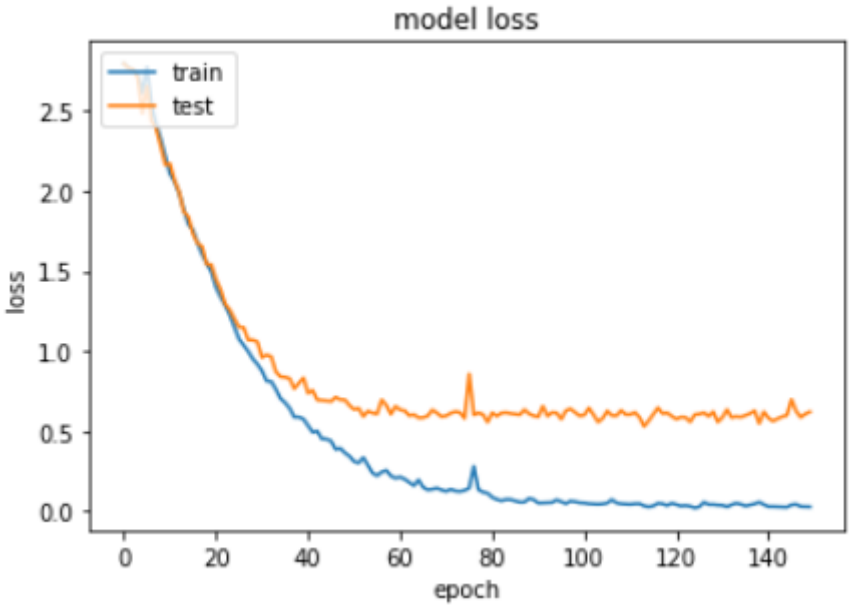

Figure 4: Training and validation loss curve of model for 150 epochs

Clustering of embeddings:

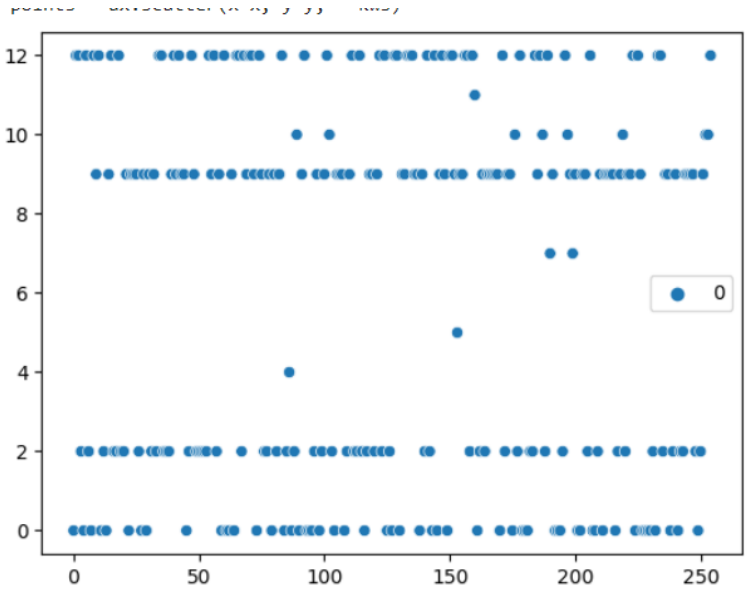

Figure 5: Clustering output for speech segment

Here, the embeddings extracted with a window length of 256, gives us 256 distinct embeddings which are labelled with a particular language code and then clustered to realise which embeddings correspond to which cluster. Here we can observe consistency over certain embedding segment intervals and a few code switching instances between languages.

Chapter 5

CONCLUSION

1. In this study, we successfully implemented a language diarization system to automatically identify and separate speech segments in an audio file containing code-switching between multiple languages. The proposed approach, which utilised CNN+LSTM architecture, achieved 91.75% validation accuracy in language identification, demonstrating its effectiveness for handling code-switching scenarios
2. Our study contributes to the growing field of language diarization research, providing a practical approach for automatically identifying and separating speech segments in code-switching audio data. The results obtained from this project can be used to further improve the accuracy and robustness of language diarization systems in real-world scenarios

Chapter 6

FUTURE SCOPE

1. Improving accuracy and efficiency of the language identification model to make it more robust by training for more languages and more files for each language
2. Implementing other clustering algorithms (eg. K-Means, knn, latent class) and compare which algorithm yields the best results.
3. Implementing the model for code switching for comparatively smaller window length to analyse the real world scenario of the speed of code switching.

Bibliography

[1]D. -C. Lyu, E. -S. Chng and H. Li, "Language diarization for conversational code-switch speech with pronunciation dictionary adaptation," *2013 IEEE China Summit and International Conference on Signal and Information Processing*, Beijing, China, 2013, pp. 147-150, doi: 10.1109/ChinaSIP.2013.6625316.
[2] Spoken Language Recognition using X-vectors David Snyder, Daniel Garcia-Romero, Alan McCree, Gregory Sell, Daniel Povey, Sanjeev Khudanpur. https://www.isca-speech.org/archive/odyssey_2018/snyder18_odyssey.html
[3] E. Yilmaz, M. McLaren, H. van den Heuvel and D. A. van Leeuwen, "Language diarization for semi-supervised bilingual acoustic model training," *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, Okinawa, Japan, 2017, pp. 91-96, doi: 10.1109/ASRU.2017.8268921.
[4] Spoken Language Diarization Using an Attention based Neural Network Jagabandhu Mishra, Ayush Agarwal and S. R. Mahadeva Prasanna Department of Electrical Engineering https://www.researchgate.net/publication/352223715_Spoken_Language_Diarization_Using_an_Attention_based_Neural_Network.
[5] M. H. Moattar and M. M. Homayounpour, “A review on speaker diarization systems and approaches,” Speech Communication, vol. 54, no. 10, pp. 1065–1103, 2012.
[6] L. v. d. Maaten and G. Hinton, “Visualizing data using t-SNE,” Journal of machine learning research, vol. 9, no. Nov, pp. 2579–2605, 2008.
[7] J. Braun and H. Levkowitz, “Automatic language identification with perceptually guided training and recurrent neural networks,” in Fifth International Conference on Spoken Language Processing, 1998.
[8] J. Gonzalez-Dominguez, I. Lopez-Moreno, H. Sak, J. Gonzalez-Rodriguez, and P. J. Moreno,“Automatic language identification using long short-term memory recurrent neural networks,” in Fifteenth Annual Conference of the International Speech Communication Association, 2014.