Speaker Diarization for Single Channel Audio

Undergraduate Thesis

Submitted in partial fulfillment of the requirements of
BITS F421T Thesis

By

Simran Sehgal
ID No. 2017A8TS0405P

Under the supervision of:

Dr Eng Siong Chng
&
Dr. Pham Van Tung

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, PILANI CAMPUS
NANYANG TECHNICAL UNIVERSITY, SINGAPORE

December 2020

Declaration of Authorship

I, Simran SEHGAL, declare that this Undergraduate Thesis titled, 'Speaker Diarization for Single Channel Audio' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Date: 4/12/2020

"It's not that I'm so smart, it's just that I stay with problems longer"

Albert Einstein

Abstract

Speaker Diarization for Single Channel Audio

by Simran Sehgal

Speaker Diarization is the process of annotating audio files with multiple speakers (telephone conversations, meetings, party) in time frame labels as per who is speaking when. It is an important step in preprocessing speech and has many applications- in the transcription of a video, recording a meeting, taking notes.

This thesis explores the components that make up a typical Speaker Diarization pipeline, see how they have adapted over time, and looks out for future trends. Various state of the art systems are implemented and their results have been verified with the published results. Minor improvements have also been suggested and implemented as part of this thesis.

In particular, two systems are reviewed and implemented- the RPNSD, which is a computer vision inspired network applied to speech for diarization, and an LSTM Similarity Matrix, which replaces the classic PLDA to score 2 segments based on cosine similarity.

This thesis proposes two novel improvements. The first on top of RPNSD, it attempts to make the clustering process more friendly to neural network generated embeddings. This is done using SOMs which are neural networks that learn how certain input vectors are distributed in the neighbourhood. The results are positive for conversations with two speakers but not optimal for more number of speakers.

The second improvement is done to account for overlapped speech in conversations and tackle those segments by labeling them with a second speaker. This is implemented on top of a standard baseline in the Speaker Diarization community- Kaldi's diarization recipe with x-vectors. The chosen overlap detection algorithm labels a segment with the speakers nearest to it in time. The results show a reduced DER for overlapped speech and is a promising future direction.

Acknowledgements

I would like to express my sincere gratitude to Dr. Chng Eng Siong, Dr Pham Van Tung of NTU, Singapore and Dr Xionghu Zhong of Hunan University, China for their guidance and patience with me during our weekly meetings. I have learnt a lot from your methodical approach to research, and hope to hone this quality further in my life.

I would like to thank Dr KK Gupta at BITS Pilani, India for his co-supervision and readiness to help. Your cooperation made the remote thesis a smooth experience.

I would like to tip a hat to my fellow teammate Huan Yu for sharing the excitement and difficulties of trying out new things!

Contents

Declaration of Authorship i

Abstract iii

Acknowledgements iv

Contents v

List of Figures vii

List of Tables viii

Abbreviations ix

1 Introduction 1
- 1.1 What is Speaker Diarization . . . 1
- 1.2 Motivation . . . 2
- 1.3 Contributions . . . 2
- 1.4 Organization . . . 3

2 Overview of a Speaker Diarization System 4
- 2.1 Background . . . 4
  - 2.1.1 Problem . . . 4
  - 2.1.2 Applications . . . 5
  - 2.1.3 Evaluation . . . 5
  - 2.1.4 Metric . . . 6
  - 2.1.5 Data . . . 7
- 2.2 Components . . . 7
  - 2.2.1 SAD . . . 7
  - 2.2.2 Embeddings . . . 8
  - 2.2.3 Similarity Measurement . . . 11
  - 2.2.4 Clustering . . . 11
  - 2.2.5 Resegmentation . . . 13
- 2.3 Open Source Toolkits . . . 14
  - 2.3.1 LIUM . . . 14

2.3.2 Kaldi . . . 14
2.3.3 Pyannote-Audio . . . 15
2.4 Neural Network Models . . . 17
2.5 Summary . . . 17

3 The Region Proposal Network Model **19**
3.1 Background . . . 19
3.1.1 Faster RCNN . . . 19
3.1.2 Non-Maximum Supression . . . 19
3.2 Architecture . . . 20
3.3 Evaluation . . . 21
3.4 Limitations and Improvements . . . 22
3.5 Summary . . . 24

4 LSTM based Similarity Matrix **25**
4.1 Pipeline . . . 25
4.2 Methodology . . . 26
4.3 Training . . . 26
4.4 Evaluation . . . 26
4.5 Summary . . . 27

5 The Overlap Detection Module **28**
5.1 Motivation . . . 28
5.2 Algorithm . . . 28
5.3 Evaluation . . . 28
5.4 Summary . . . 29

6 Conclusions and Future Work **30**
6.1 Conclusions . . . 30
6.2 Future Work . . . 31
6.2.1 RPNSD . . . 31
6.2.2 LSTM Similarity Matrix . . . 31
6.2.3 Overlap Detection . . . 32

Bibliography **33**

List of Figures

2.1 A visual depiction of Speaker Diarization 4
2.2 The Speaker Diarization pipeline 5
2.3 Fitting training data onto a UBM 8
2.4 Extraction of d-vectors 9
2.5 Extraction of X-vectors 10
2.6 A HMM modelling a Speaker Diarization System 14
2.7 Kaldi's Speaker Diarization Pipeline 15
2.8 The Pyannote Pipeline 16
2.9 Model Architecture of Pyannote's x-vector 16
2.10 The EEND Pipeline 17

3.1 A Visual Depiction of Non-Maximum Suppression 20
3.2 Regular Speaker Diarization Pipeline vs RPNSD Pipeline 20
3.3 Embedding Extraction from Proposals of Speech 21
3.4 A Self Organising Map learning neighbourhood of input vectors. 23

4.1 Pipeline with LSTM similarity scoring matrix 25
4.2 Training the BLSTM model 26

List of Tables

2.1 Evaluation Dataset Features 7
2.2 DER using Kaldi's Diarization Pipeline 15

3.1 RPNSD evaluated on Callhome using 5-fold cross validation 22
3.2 RPNSD+ SOM evaluated on a subset of Callhome 23

4.1 LSTM pipeline Evaluted on Callhome 27

5.1 Overlap Detection Algorithm Evaluated on Callhome 29

Abbreviations

ASR	Automatic Speech Recognition
ANN	Artifical Neural Network
BIC	Bayesian Information Criterion
BLSTM	Bidirectional Long short-term memory
CNN	Convolutional Neural Network
DER	Diarization Error Rate
GMM	Gaussian Mixture Models
EM	Expectation Maximization
HMM	Hidden Markov Models
JER	Jaccard Error Rate
JFA	Joint Factor Analysis
LSTM	Long short-term memory
MAP	Maximum a Posteriori
MFCC	Mel-Frequency Cepstrum Coefficients
NIST	National Institute of Standards and Technology
NMS	Non Maximum Suppresion
RCNN	Region based Convolutional Neural Networks
RPN	Region Proposal Network
RPNSD	Region Proposal Network based Speaker Diarization
SAD	Speech Activity Detection
SOM	Self Organizing Maps
STFT	Short-time Fourier transform (STFT)
TDNN	Time Delay Neural Network
UBM	Universal Background Model
VAD	Voice Activity Detection

To 2020.

Chapter 1

Introduction

1.1 What is Speaker Diarization

Speech is the primary form of interaction between humans and in recent times, of humans with machines. It is perhaps the most important tool we have to understand each other. There has been considerable research in understanding speech, identifying and verifying speakers, analyzing speech for events and emotions, and processing who is speaking when.

Speaker Diarization is one such component of speech processing which involves annotating an unprocessed audio recording at the times a speaker speaks with the speaker label. It answers the question of 'who is speaking when'. Speaker Diarization is very closely related to Speaker Verification since they both involve distinguishing between speakers, but the challenge in a Speaker Diarization system lies because it has not encountered the speakers beforehand. Speaker Diarization aims to distinguish between multiple speakers ideally in one pass.

In recent times, when there is a lot of audio content online- podcasts, news broadcasts, films, Speaker Diarization helps in making a robust transcribing module, working in tandem with ASR. Speaker Diarization can also help in psychoanalysing the nature of conversation by looking at order of turns taken by speaker or dominant and quiet speakers.

For diarizing an audio, we might have one of 2 different forms of audio- single-channel or multi-channel speech. Single-channel audio files are common in interviews and recordings of spontaneous speech where the entire conversation is recorded using single device. In contrast, multi-channel speech has more than 1 channels of speech, and this can play an important role while designing the diarizing system. Examples of this type are found in video meetings where the channel of audio directly depends on the speaker, or a meeting recording using an array of mics, which holds some spatial information about the speaker. For the purpose of this thesis, we

will focus only on single-channel diarization since it is present more widely and multi-channel can also be solved as a single-channel diarization problem.

1.2 Motivation

The task of Speaker Diarization has been widely solved as subproblems-detecting speech regions, extracting features and, finding similar segments to represent speakers. Solving these problems individually is easier than optimizing the whole pipeline to distinguish speakers accurately.

Over the years, there have been research works that optimize a certain module of the pipeline. Extracting features and clustering have been the most popular parts to innovate upon. Recent times have observed a shift from traditional speech theory to neural network approaches, which serves especially well for Speaker Diarization systems, since neural networks can be trained for the whole pipeline, in an end to end manner, or individual neural network blocks can be jointly optimised together.

Issues also arise due to the variability in speakers encountered with different audio types, the recording conditions, the distribution of time each speaker speaks etc. Thus Speaker Diarization systems need to be adapted for specific domain data before use for best results.

Speaker Diarization is a vast field with very different research directions. The motivation of this thesis is to analyse and implement the state of the art research directions, and figure out the trend for the future directions.

1.3 Contributions

The main contribution of this thesis lies in understanding and analysing the recent works of Speaker Diarization. The vast problem of speaker diarization has been narrowed down to single channel audio and recent research directions have been implemented, verified and analysed. At the end, possible future directions have been proposed.

While reproducing the state of the art results, minor improvements have been tried upon the existing architecture and the results have been analysed. One such improvement shows promising results for audio files with only 2 speakers. Another improvement resulted in approximately 2% decrease in error for overlapped speech by including an overlap detection module. The implementations and results of this thesis can be verified at https://github.com/sehgal-simran/Spk-Dzn

1.4 Organization

This thesis is organised in 6 chapters. The first 2 chapters are introductory, with the first chapter introducing the problem statement of the thesis, the motivation felt by the author and the contributions made. The second chapter gives an overview of the speaker diarization system, lists the open source toolkits and their technology and reviews recent research neural network inspired advances in the field.

The next 3 chapters constitute the systems which were analysed and improved upon. The third chapter introduces the RPNSD, its key concepts, architecture, evaluated results and improvement. The fourth chapter reviews the LSTM similarity matrix, its methodology, training and evaluation. The fifth chapter focuses on the need for an overlap detection system in the module and evaluates the results after implementing one.

The last chapter concludes the thesis and offers some future work for all the major research works introduced.

Chapter 2

Overview of a Speaker Diarization System

2.1 Background

As described in Chapter 1, the aim of this thesis would be to identify, review and modify current approaches to speaker diarization. In this chapter, necessary background information is described to understand the trend of current research.

2.1.1 Problem

As mentioned before, the goal of the system is to partition an audio stream firstly into homogeneous regions of speech, and then attribute to each segment a particular speaker. In some applications, the number of speakers is known, but often the system has no prior information and only receives the audio file as input. The system outputs speaker wise time annotations and a brief working can be seen in figure 2.1

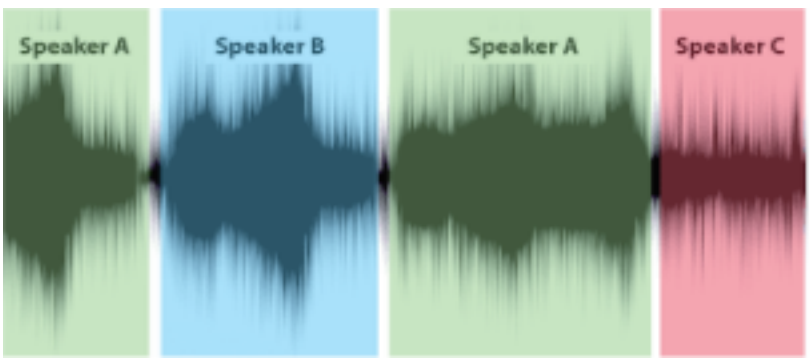

Figure 2.1: A visual depiction of Speaker Diarization

The entire system broadly performs 3 steps:

1. Speech Activity Detection
2. Embedding Extraction
3. Clustering

A more detailed step-by-step breakdown can be seen in figure 2.2, where after detection of speech regions, the audio stream is segmented into small-size frames. Embeddings are extracted from these frames which are clustered and then merged to give an output file annotated time-wise per speaker. There is often an implicit assumption that 1 embedding belongs to 1 speaker while clustering.

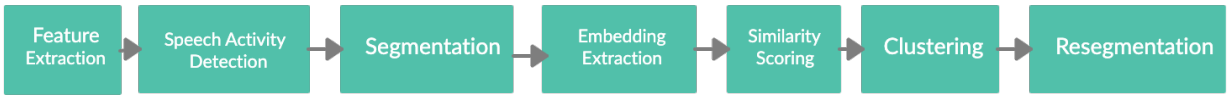

Figure 2.2: The Speaker Diarization pipeline

2.1.2 Applications

Speaker Diarization systems are used for a host of applications such as analysing meeting data or transcribing podcasts and videos. Annotating audio file according to who spoke when greatly increases the information present in the file, and enables new methods of retrieval. A user can now search an audio, say a meeting, for utterances of a particular speaker, say the teacher.

Apart from working as a standalone system, it also increases the accuracy of ASR systems in multi-speaker environments by acting as a back-end processing system

2.1.3 Evaluation

This section introduces the metrics commonly used for evaluation and some of the popular datasets utilised for comparing Speaker Diarization systems.

2.1.4 Metric

Diarization Error Rate

The most popular evaluation metric is the DER which is described by NIST. In a nutshell, it is the proportion of time that is imperfectly labelled- as a wrong speaker, or a wrong speech region. The DER is computed by comparing the output hypothesis file with a reference file as below:

$$DER = \frac{\sum_{s=1}^{S} \text{dur}(s) \cdot (\max(N_{ref}(s), N_{hyp}(s)) - N_{corrcct}(s))}{\sum_{s=1}^{S} \text{dur}(s) \cdot N_{ref}}$$

The above equation is similar to a weighted average of segments according to their duration. S is the total number of segments, $N_{ref}(s)$ is the number of speakers in a segment s in the reference file, $N_{hyp}(s)$ is the number of speakers in a segment s in the output hypothesis file and $N_{correct}(s)$ are the number of speakers in segment s which have been correctly matched between reference and hypothesis.

Diarization is generally calculated with an accepted collar of 0.25 seconds, which is the acceptable error tolerance.

The DER can be broken down into 3 components: Speaker Error, False alarm, Missed Speech. Speaker error is the percentage error that crops up due to wrong identification of speaker. False alarm error and Missed Speech are essentially errors of the SAD, when a non-speech/speech region is erroneously labelled speech/non-speech.

$$DER = E_{spkr} + E_{MISS} + E_{FA} + E_{ovl}$$

Jaccard Error Rate

JER was introduced for the Dihard II Challenge and is based on the Jaccard index. The jaccard index is a similarity metric which measures the similarity between 2 datasets by finding the ratio between their interection to their union.

For N reference speaker and M system speakers, an optimal one to one mapping is produced. Then for each reference speaker, the speaker specific JER is calculated as:

$$JER = \frac{FA + MISS}{TOTAL}$$

The total JER is the average of the JER of all reference speakers and is often highly correlated with the DER of the system.

2.1.5 Data

Data in diarization tasks is not easy to get by since manual annotation at timestamps by speaker is required, which is a labour intensive issue. Thus, there are not as many Speaker Diarization datasets as for other speech processing tasks.

However since modules in Speaker Diarization such as the embedding extraction are trained with the task of distinguishing speaker, datasets used in speaker recognition and verification can be used here. Some of these include:

- NIST Speaker Recognition Evaluation datasets (2004, 2005, 2006, 2008)
- Switchboard Cellular (Part 1, Part2)
- Dihard
- ICSI meeting corpus
- AMI training meeting corpus

For evaluation, there are 2 popular datasets- The AMI meeting corpus and the Callhome corpus, the statistics of which are listed in Table 2.1

Features	Callhome	AMI Meeting Corpus (Train)
Sampling Frequency	8 KHz	16 KHz
Number of Recordings	500	147
Number of Hours	17	100
Language	Multlingual (English, Spanish etc)	English
Average duration of file	124.5 secs	37.9 mins

Table 2.1: Evaluation Dataset Features

2.2 Components

2.2.1 SAD

SAD/VAD is the first step in processing the raw audio input. It separates the stream into segments of speech and non-speech (silence). An SAD is essentially a classifier which classifies a segment based on its features (MFCCs, STFTs). SADs have evolved from GMMs to TDNNs and are now being combined with the later modules to form a single neural network.

A robust SAD can make a huge difference to the system since it is the primary contributor of missed speech and false alarm errors. Most SADs only detect regions of speech and not overlap, so overlapped speech is also labelled as speech attributed to Single speaker.

2.2.2 Embeddings

This module produces a fixed size embedding for small chunks of segment of 1.5-2 secs input to it. Embeddings are calculated from speech features like MFCC, STFT extracted from each frame of the audio chunk. These embeddings optimally capture all the distinctive speaker information present in the audio segment.

In earlier years, the features from each frame were taken to model GMMs.[7] Najim Dehak introduced i-vectors[19], meant to be fixed size embeddings that could capture all the information of the speaker from the segment in a compact manner. Since then, i-vectors were the state of the art for a lot of time until the neural network generated embeddings arrived, namely x-vectors and d-vectors.

i-vectors

Early breakthroughs in speaker verification and diarization tasks used GMM-UBM on speech features like mfcc. These models captured speaker and channel variability both inseparably. A GMM-UBM was constructed from all the training dataset to model all speakers as mixture of Gaussian Models.

The means μ, covariance Σ and coefficents π_k were adjusted so that the GMM could approximate the training data. However often there was a shortage of training samples which led to suboptimal training of the GMM. Thus,a UBM had to be used here, which was essentially another GMM pretrained on a large corpus, and rather than training from scratch, the training data finetunes the UBM to estimate a GMM-UBM. The process is shown in figure 2.3

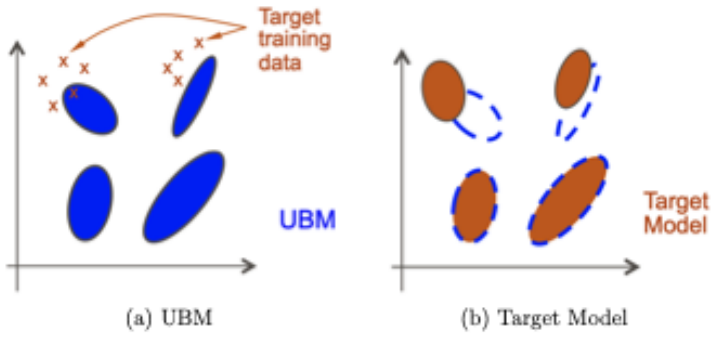

Figure 2.3: Fitting training data onto a UBM

To get the best fitted UBM, EM was used to get the optimal paramters. And to adapt from UBM to target GMM, MAP was used, which was similar to UBM.

To model speaker and channel variability separately, Joint Factor Analysis[10] was proposed by Kenny, which decomposes the supervector generated by the GMM (which depends on speaker and channel) into two subspaces which represent the speaker and channel variabilities individually.

$$M = m + Tw$$

In the above equation, M is the supervector extracted from the UBM-GMM based on the audio, m is the speaker and channel independent supervector, which is simply the supervector mean from the UBM in this case, to provide a starting point. T is a low rank total variability matrix which models the total variance of both speakers and channels and w represents the total variability factors. The posterior mean of w, estimated using MAP are the i-vectors.

d-vectors

d-vectors are embeddings extracted from deep neural networks. The input to these could be any audio feature- MFCC, frame energy, STFT or others.

d-vector extraction is shown in the figure 2.4 where stacked framewise features are input to the neural network. The neural network consists of several fully connected layers with maxout and dropouts in between to prevent overfitting. d-vectors are extracted from the last layer's activation values. If multiple d-vectors are available, then a pooling layer is added and the average value is taken.[21]

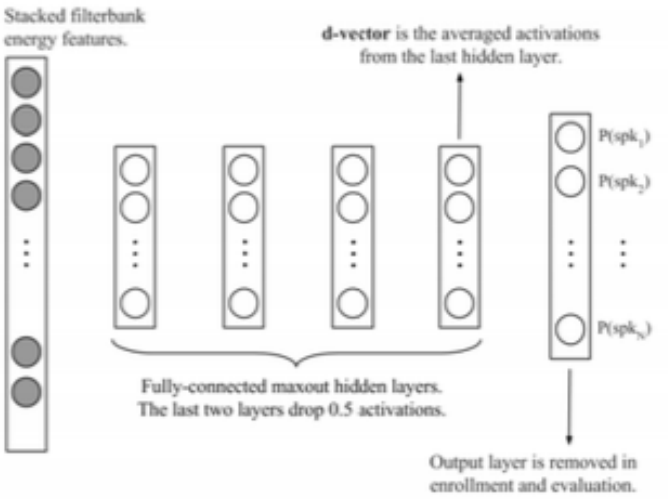

Figure 2.4: Extraction of d-vectors

The final output layer shown in the neural network is removed while evaluating and testing the system. This layer is only added for training purposes where each node represents one speaker of the dataset represented in a one hot labelling format. We do not require it while testing since our task is of embedding generation and not speaker verification.

X-vectors

X-vectors are the current state of the art discriminatively generated speaker embeddings. They are generated from time delay neural networks which convert high dimensional speech features into a low dimensional representation, preserving context for distinction.[20] A typical x-vector TDNN is shown in Figure 2.5.

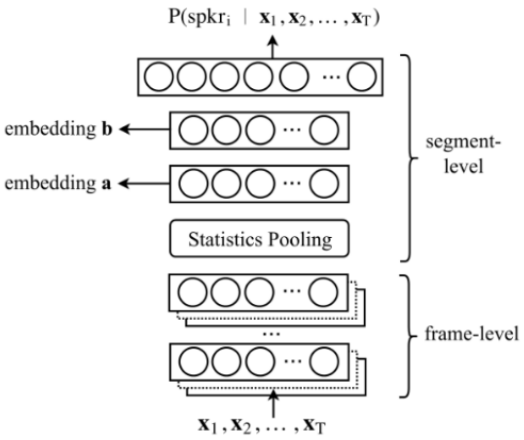

Figure 2.5: Extraction of X-vectors

Speech features such as MFCCs, STFTs are input into the cascaded frame-level layers. This step captures the temporal information in the speech signal as you could specify the time delay, say x, which would mean x+1 frames will be considered. The pooling layer serves as an aggregator by converting the frame level information into a recording level information, which represents the original segment utterance. The size of the vector is decided by these recording layers and is most commonly in multiples of 128.

The final output layer is only relevant for training purposes and is removed at evaluation and testing times. The layer, like in d-vectors, represents a one-hot label to denote each speaker in the training set and is trained to output the probability of each speaker. It is removed and instead, the last hidden recording layer's activations are used since diarization systems may encounter speakers which the model has not seen.

X-vectors have been known to perform significantly better than i-vectors as they are able to make use of larger training data, whereas i-vectors get constrained after a certain point. This

also means we can train x-vectors with data augmentation to increase diversity of training data and make them more robust.

2.2.3 Similarity Measurement

Given 2 vectors, their similarity needs to be quantified before we can move on to the clustering step. Traditional methods to measure similarity are Cosine Similarity and PLDA. A simple heuristic approach could be to classify 2 segments to the same speaker if the score is greater than a certain threshold, or further cluster them using algorithms.

Cosine Similarity

Cosine similarity measures the similarity between 2 vectors by computing their dot product. It is 1 when the vectors are same, and 0 when they are entirely different.

As cosine similarity is a simple vector operation, it is often used to train similarity scoring matrices, an example of which we will see in Chapter 4.

PLDA

Probablistic Linear Discriminative Analysis projects the input data into a lower dimensional space preserving their discriminative aspects. Mathematically, it maximizes inter variations and minimizes intra-class variations.

PLDA is built on LDA, a supervised dimensional reduction technique. PLDA is a generative model which assumes that a training sample X is generated from a Gaussian distribution.[18]

2.2.4 Clustering

After segmenting the speech regions, producing a representative embedding and generating similarity scores, the pipeline has a clustering module which clusters the embeddings. Common clustering algorithms are explained as follows:

K Means Clustering

K means is a popular clustering algorithm that aims to partition given set of n observations of d dimensions into K clusters by minimizing the within cluster sum of squares. K means generally works well for speech data, though it sometimes might take time to converge to an

optimum value since it is an iterative algorithm. Hence, settings of operation need to be clearly experimented with and defined. K-means is non-deterministic and time complexity increases with number of speakers or clusters.

GMMS

Gaussian mixture models are a bottom-up probabilistic approach to clustering. As input we have many small segments of speech that are assumed to contain only 1 speaker. A universal background Gaussian mixture model (UBM) is trained on all the input segments using EM algorithm.

Initially each segment is treated as a single cluster and is represented by a GMM adapted from the UBM using MAP. A single GMM is represented by its mean m, weight w and variance σ. The distance between 2 GMMs P and $\hat{P}$ on the basis of which they are clustered is

$$D(P, \hat{P}) = \sqrt{\sum_{k=1}^{K} \sum_{d=1}^{D} \sigma_k \cdot \frac{(m_{k,d} - \hat{m}_{k,d})^2}{\sigma_{k,d}^2}}$$

where K is the number of clusters m and $\hat{m}$ are the respective means, and σ_k is the variance.

For clustering purposes, K is initially the total number of segments. After calculating pairwise distances, clusters with least distance between them are grouped together and a new GMM is estimated. The clustering stops once the distance is above a certain threshold.

Agglomerative Hierarchical Clustering

AHC, like GMMs is also a bottom-up hierarchical clustering approach. At the beginning, each segment is seen as an individual cluster, and clusters are merged based on their similarity scores. AHC generally uses similarity scores generated from previous modules, like cosine similarity or PLDA.

The progressive clusterings result in a dendogram structure and offer a unique advantage to using AHC. For the case that we do not know the number of speakers in the input stream, we could perform AHC till all segments represent one structure, and from the dendrogram, choose the most suitable stage which represents K classes. This is especially useful in Speaker Diarization systems when this is often the case, and thus AHC is widely used in this industry.

Spectral Clustering

SC transforms the problem of unsupervised clustering into a graph partitioning problem. Spectral Clustering works very well for non-convex data and it is a popular choice for Speaker Diarization systems since computation is easy as it is based on linear algebra.[13]

SC transforms the data points (in our case, speech segments) as nodes of the graphs connected by edges which represents similarity between the segments. The adjacency matrix, degree matrix and Graph Laplacian matrix are calculated.

The Graph Laplacian matrix has several properties of our interest which aid in spectral clustering. The first non-zero eigenvalue of this matrix represents the 'spectral gap' of the data, representing the density. The second eigenvalue, called the Fiedler value, and the corresponding vector, called the Fiedler vector give us information on partitioning the graph into 2 connected components.

2.2.5 Resegmentation

This is an additional refinement step which makes a second pass over the audio stream with the calculated results from the first pass, and aims to align frames accurately. The state of the art resegmentation is the Variational Bayes resegmentation introduced by JHU as a major improvement over the previous system of Viterbi resegmentation. The Variational Bayes HMM based system is actually a standalone diarization system, but when initialised with the resulting labels of each segment, we can consider it to be a resegmentation system. [17]

The VB HMM diarization system tries to model the entire audio stream using an HMM. The HMM has speaker specific states, which are modelled using GMMs. Overlapped speech is not considered with each state representing only 1 speaker, and transitions between states being governed by learnt transition probabilities. The model might have multiple states for a single speaker, with the same speaker specific distribution.[4]

One such system is represented in Figure 2.6, with 3 speakers and 1 state per speaker. Model can enter the state of speaker 1,2,3 with probability π_1, π_2 or π_3 respectively, and on reaching the state, can choose to remain in the state with probability P_{loop} or exit it with probability $1 - P_{loop}$.[3]

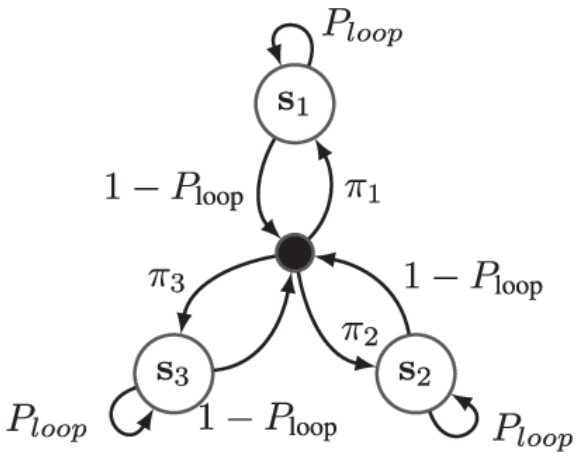

Figure 2.6: A HMM modelling a Speaker Diarization System

2.3 Open Source Toolkits

A lot of open source toolkits based on various techniques are available for speaker diarization and described below.

2.3.1 LIUM

Lium is a JAVA software for speaker diarization optimised for radio and TV shows. Lium was developed in 2008 and was state of the art until the advent of x-vectors. Lium mainly uses i-vectors from 13 MFCC coefficients as features and performs segmentation based on BIC. [14]

2.3.2 Kaldi

Kaldi is a popular toolkit for speech recognition and contains scripts for various speech related tasks.[15] It is written in C++, Bash and Python and has been quick to adapt to neural network approaches too. Specifically for single channel speaker diarization, it contains 2 pipelines, one with i-vectors and one with x-vectors. The steps followed by Kaldi's speaker diarization system are roughly the components described above and shown in the figure 2.7.

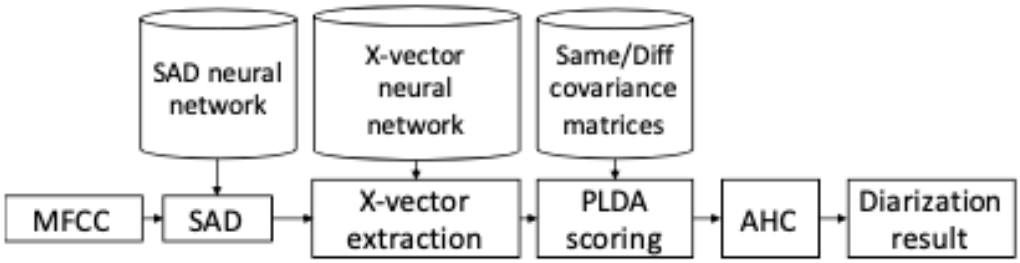

Figure 2.7: Kaldi's Speaker Diarization Pipeline

Kaldi's x-vector system is a very popular baseline which is often compared with by recent research works while evaluating their systems. This is because Kaldi provides trained models for each module of the pipeline and building or modifying on top of this pipeline is easy.

Kaldi's x-vector models and PLDA are trained on augmented Switchboard, Mixer 6, and NIST SREs. The PLDA is also adapted to callhome data for a 2-fold cross validation experiment.

This recipe of Kaldi is commonly used as a baseline to compare the results of recent research direction with. Table 2.2 lists the performance of the Kaldi baseline on the Callhome dataset.

Overlap	DER
Overlap included	16.78
No overlap %	7.09%

Table 2.2: DER using Kaldi's Diarization Pipeline

2.3.3 Pyannote-Audio

Pyannote is an entirely neural network based Speaker Diarization published online. Unlike Kaldi, it is maintained only for speaker diarization and not other tasks. It is written entirely in python and also has trained models available for each module, ready to be used.[1]

Pyannote-audio is still under release at the time of writing of this thesis, but is gaining popularity with researchers as being completely Pythonic makes it highly accessible. It is also entirely built with neural network blocks, and thus building or modifying system on top of it would be efficient.

The pyannote-audio pipeline is shown in figure 2.8.

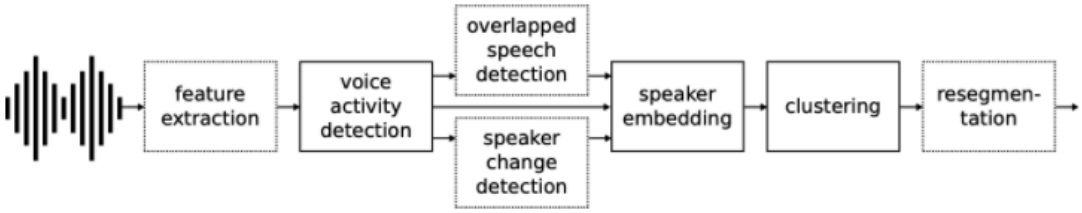

Figure 2.8: The Pyannote Pipeline

The pipeline is built using the AMI meeting corpus, which has meeting audio files with a sample rate of 16KHz. Features extracted are MFCCs with a frame size of 25ms and a frame shift of 10ms. Pyannote-audio considers 19 MFCC coefficients as features.

Speech Activity Detection as explained previously detects speech regions in a given audio stream. Pyannote-audio utilises a bidirectional LSTM for speech detection with a hidden size of 128 cells. SAD works on segments, and is trained for 2 second audio segments. Training data is also augmented with Musan noise to increase training data and make system more robust.

Pyannote also provides modules for overlapped speech detection modules and speaker change detection, though these are not present in the published pipeline. The modules are quite similar to SAD, especially overlapped speech detection. OSD labels a segment as 0 if no speech, 1 if 1 speaker, and 2 if multiple speakers.[1]

Speaker embedding follows an x-vector approach by training a model based on SincNet and TDNN. The model architecture is shown in the figure 2.9

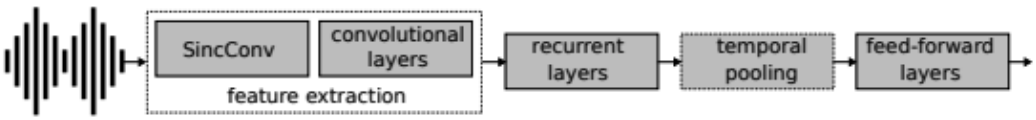

Figure 2.9: Model Architecture of Pyannote's x-vector

Pyannote-audio uses a SincNet neural network layer implemented as a BLSTM to extract features instead of traditional MFCC approach. This is more beneficial because the filters in the SincConv layer are trained rather than setting them manually, and thus are more optimal for the task. The first convolutional layer is critical in this architecture since it deals with high dimensional data and can be affected by vanishing gradients. After feature extraction, the approach is similar to x-vectors with temporal pooling and feedforward layers. The major difference lies in the output size of the x-vector which is 512 for pyannote and 128 for kaldi.[1]

2.4 Neural Network Models

As mentioned before, a major obstacle in optimising speaker diarization systems is the modular approach with which pipelines are built. Error at each stage compounds and reflects in the final DER. At each stage, we are indirectly optimising for our result through parameter tuning but not using the ground truth labels which are available to us.

With the large scale adoption of LSTM architectures in speech and CNN architectures in Computer Vision, a lot of research in speaker diarization has proceeded along similar lines. Since 2017, the new wave of Speaker Diarization systems are increasingly compact and neural network based. These neural networks redesign the problem context and solve multiple tasks in the pipeline at once, thus reducing it.

Yusuke Fujita et al, describe such a system, the EEND neural network in which they introduce a Bi-LSTM neural network to replace the embedding and clustering module.[5]. They further improve upon this work by using self attention blocks instead of BLSTM.[6] This approach offers the advantages discussed above- a short, robust pipeline which is directly optimised with the diarization results of the training data and is shown in Figure 2.10

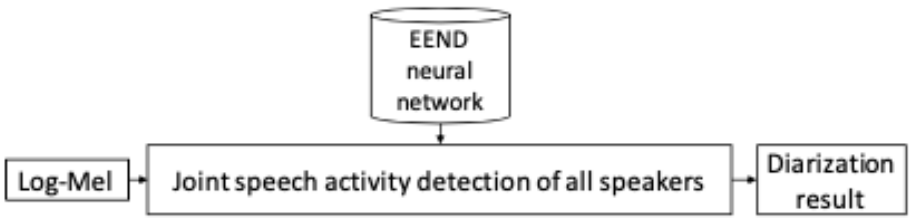

Figure 2.10: The EEND Pipeline

The authors redefine the clustering problem to a multi-label classification problem. A similar approach has been proposed by Quija Li et al, where they transform the unsupervised clustering module to a supervised neural clustering module using encoder-decoder architecture.[11] Aonan Zhang et al. also tackle the problem of unsupervised clustering using a neural network approach, namely a generative RNN model which is trained to generate the most likely sequence of speakers that can generate the input set of features.[22]While these are not end-to-end systems, it is easy to combine them with other neural network modules and train them jointly.

2.5 Summary

In this Chapter, the Speaker Diarization problem is introduced formally, along with the pipeline using which it is commonly solved, and the applications of the Speaker Diarization system are

discussed.

To understand the problem better, we look at the governing metrics of the problem, the DER and the JER, which are often highly correlated resulting in only 1 of them being mentioned, which is mainly the DER. We also understand the datasets that reearchers work with for this problem, and familiarize ourselves with the 2 most common diarization sets, the AMI Meeting Corpus and the Callhome dataset.

In Section 2.2, various components of the Speaker Diarization system were introduced and their evolution was reviewed. Section 2.3 covers the various open source toolkits available online for the task of diarization, their methodologies, architecture and result. Lastly, in Section 2.4, recent neural network approaches were reviewed which disrupted the traditional pipeline and innovated with faster, more robust systems.

The next chapter explores one such research direction, the Region Proposal Network.

Chapter 3

The Region Proposal Network Model

3.1 Background

Region Proposal Network is an innovative Computer Vision inspired neural network for SD task based on Faster RCNN and NMS. It considerably shortens the regular pipeline of Speaker Diarization with less DER.

3.1.1 Faster RCNN

Faster RCNN is a framework built for object detection which is in turn based on Fast RCNN. Both of them take an image as an input to be fed into a CNN which outputs a Convolutional Feature map. Fast RCNN uses a selective search algorithm on the feature map to identify the regions of proposals, whereas Faster RCNN employs a separate network called the RPN to predict the region proposals. [16][8]

3.1.2 Non-Maximum Supression

NMS is also a computer vision concept used in object detection which takes into input a number of proposals, their corresponding scores and a parameter of overlap threshold. The concept is demonstrated in Figure 3.1. The algorithm aims to remove overlapping proposals with the following steps:

1. Select Proposal with highest confidence score and add to final list

2. Compare overlap of this proposal with every other proposal.

3. If overlap > Threshold, remove proposal. Else, repeat until no proposals left.

Figure 3.1: A Visual Depiction of Non-Maximum Suppression

3.2 Architecture

The full pipeline of the RPNSD system is shown in[9] Figure 3.2

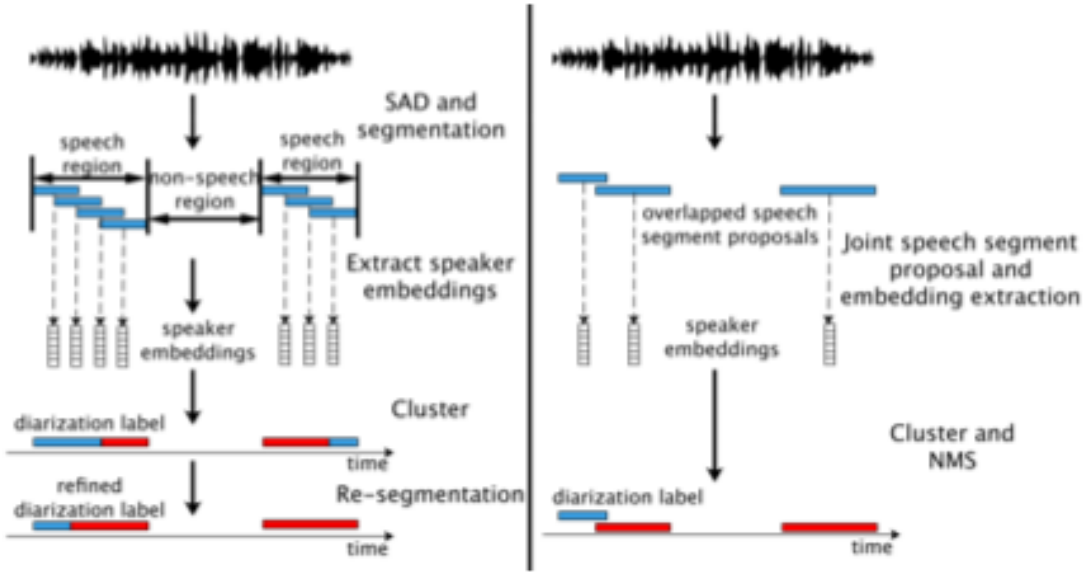

Figure 3.2: Regular Speaker Diarization Pipeline vs RPNSD Pipeline

Region Proposal Network takes feature maps as input, which would be the 1-d feature vectors from the audio instead of 2-d image feature vectors. It outputs multiple region proposals generated by expanding 9 anchors of size (1, 2, 4, 8, 16, 24, 32, 48, 64) at ever timestep, thus

covering speech segments from 16 to 1024 frames. The RPN scores these proposals and refines their boundaries with convolution layers. This process is shown in Figure 3.3 [9]

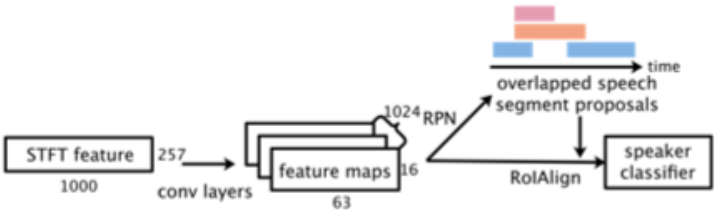

Figure 3.3: Embedding Extraction from Proposals of Speech

STFT features are extracted from 10 second audio chunks with a frame size of 512 and a frame shift of 80, where 1 frame is 1 ms. The feature size for a chunk is [257,1000]. The features are input to a convolutional layer which projects these into higher dimensions and generates feature maps of size [1024,16,63] which indicate 63 timesteps, and 16 frames per timestep. These are then fed to the RPN which extends variable sized anchors at every timestep, 63x9 in total, to propose and score by confidence, speech segment regions, which are overlapping with each other. [9]

Out of these, the lower confidence score proposals are removed, and for the remaining proposals, the corresponding regions are extracted from the feature maps as deep features. The network is optimised for the loss of speech segment boundary loss, Foreground/Background loss and speaker cluster loss.

The second major step involves post processing of the speech segment proposals. Speech segments whose foreground probability is less than a threshold γ are removed. The remaining are clustered using K means or spectral clustering. Finally, NMS is applied for segments in the same cluster to remove highly overlapping proposals. This deals with overlapped speech in a natural way.[9]

3.3 Evaluation

Trained RPNSD models are available online, and were adapted using 5 fold cross validation to the callhome dataset. The results were verified with the published results and are shown in Table 3.1

Collar	DER with overlap	DER without overlap
0 ms	26.40%	22.18%
100 ms	21.46%	16.31%
250 ms	18.22%	12.93%

Table 3.1: RPNSD evaluated on Callhome using 5-fold cross validation

The implemented results were similar to the published results but the DERs were consistenly higher by 1%. This might be due to training inconsistencies between the implementation and published approach.

Note that RPNSD systems show a decrease in the miss error due to their handling of overlapped speech and a similar decrease in the speaker confusion error but at the cost of the false alarm error going up. Thus, the RPNSD system is more prone to falsely labelling a segment as speech, but is quite robust at identification of speakers. This might be because RPNSD generates embeddings from longer audio chunks and the longer context leads to more discriminative embeddings.

3.4 Limitations and Improvements

The post-processing step of RPNSD involves clustering where K means and spectral clustering are utilized. This requires certain tuning of parameters, and the problem of indirect result optimisation is faced again, though it is not as severe as in Kaldi's lengthy pipeline. To improve this limitation, the clustering step could be replaced by a neural network and the loss could be optimised together with the first step of embedding extraction.

Another reason to shift away from these clustering algorithms is that most of them are based on a quantitative distance measure. This might not represent the true similarity between speakers since it is a mathematical construct and some information might not be captured in the measure. Also since our embeddings are generated by a neural network and are not mathematically quantifiable, this deepens the issue.

To improve upon this limitation, K means clustering was replaced with a module of self organising maps and tested on a subset of Callhome dataset for analysis. SOMs are a type of ANN which were originally conceptualized to reduce dimension. Instead of backpropagation, it applies competitive learning to learn the neighbourhood's topological properties. A SOM is shown in Figure 3.4 where it takes input vectors, passes them through the network, and in the process learns the neighbourhood propoerties and how similar they are to each other

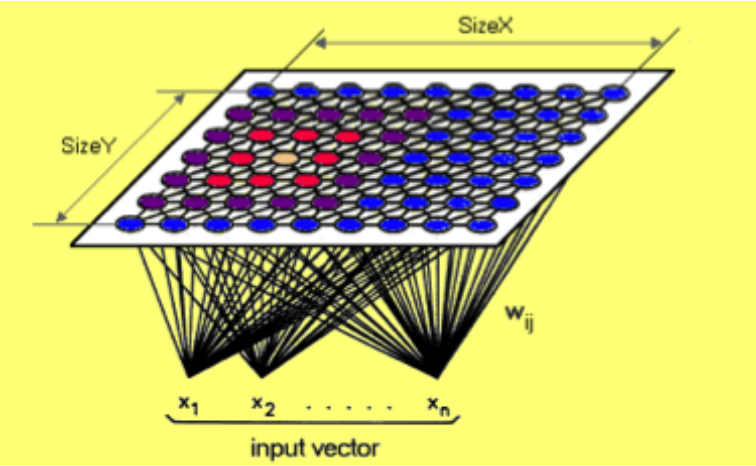

Figure 3.4: A Self Organising Map learning neighbourhood of input vectors.

The motivation for this approach was the common observation that as a by-product SOMs also output groups of similar datapoints. Since our embeddings are generated by a neural network and a non-deterministic pattern is learnt, a SOM neural network might learn the pattern better and group accordingly.

The RPNSD pipeline's clustering module was replaced by SOM and the results were evaluated on 100 callhome files.

Number of Speakers	DER with overlap	DER without overlap
2	13.02%	9.94%
2-3	15.67%	12.72%
Any	26.38%	22.59%

Table 3.2: RPNSD+ SOM evaluated on a subset of Callhome

The results are shown in Table 3.2 for 2 speakers, 2-3 speakers and it was observed that the performance of the SOM deteriorated when more number of speakers were added to the system. SOMs for 2 speaker conversation performed comparable to the RPNSD's standard pipeline but overall the performance was worse.

As the SOM did not perform upto expectations, the next step would be to find neural network architectures that would perform clustering well, so that ultimately they can be jointly optimised with the RPN network too.

3.5 Summary

In this Chapter, the RPNSD system was reviewed and its key concepts and working were understood. The Architecture of RPNSD was seen to make the pipeline of speaker diarization considerably shorter. The system was evaluated on the Callhome dataset.

Finally, the limitations of the approach and possible improvements were discussed. One such improvement, the SOM was implemented and found to work well for audio files with 2 speakers, but its performance deteriorated as the speakers increased. Thus, this approach was rejected and instead the focus of future works should be to replace the clustering module with a neural network so that ultimately the RPNSD and clustering module can be jointly trained and form 1 pipeline.

Chapter 4

LSTM based Similarity Matrix

4.1 Pipeline

This method replaces the PLDA similarity scoring matrix step in Kaldi's baseline with an LSTM trained similarity matrix. AHC clustering is replaced by spectral clustering for optimal results. The motivation to replace PLDA with LSTM is to try and encapsulate conversation structure in the similarity matrix. PLDA ignores all temporal information and sequence of segments, and only captures their pairwise similarity.[12] The pipeline is demonstrated in the figure

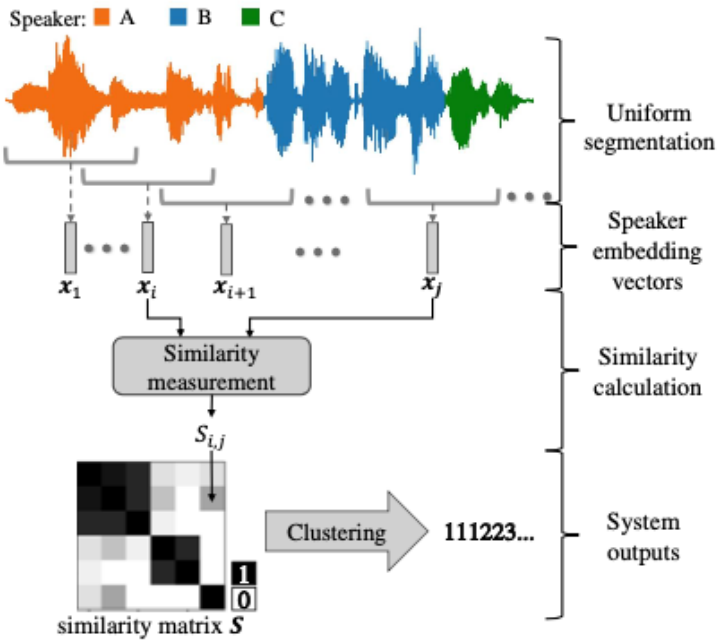

Figure 4.1: Pipeline with LSTM similarity scoring matrix

4.2 Methodology

Similarity matrix S_i is found out by carrying supervised training over the dataset using the LSTM. S_i represents one row , where S_{ij} represents similarity between i^th and j^th segment and is derived as a function of their embeddings.

$$\boldsymbol{S}_i = [S_{i1}, S_{i2}, \dots S_{in}] = f_{LSTM}\left(\begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{x}_1 \end{bmatrix}, \begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{x}_2 \end{bmatrix}, \cdots \begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{x}_n \end{bmatrix}\right)$$

4.3 Training

A major advantage of LSTM is that it works with less training data too. This is especially beneficial for callhome since it is a small dataset (17 hours) and train/test splits usually do not give best results due to less data. Hence, for PLDA, callhome is used to adapt an already trained model, and we cannot train a new one entirely using callhome data.

Training the BLSTM follows a mini-batch process to deal with variable sized audio chunks. Audio files might be lengthy, and hence are chopped off at 400 segments to train a new LSTM matrix. Thus, for larger files, the Similarity matrix is broken down into multiple trainable similarity matrices and combined at the end.[**LSTM**]

The process of learning the similarity matrix is shown in Figure 4.2.

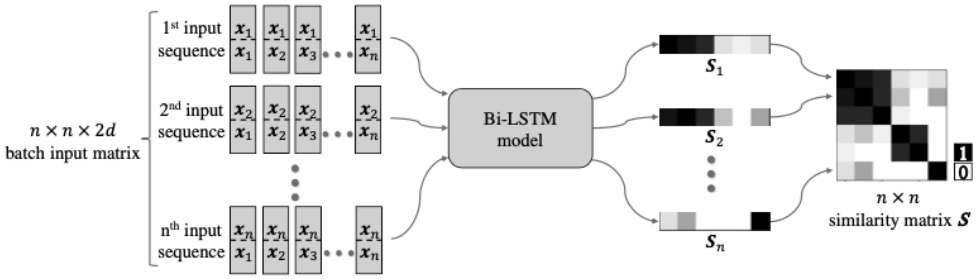

Figure 4.2: Training the BLSTM model

4.4 Evaluation

The above pipeline was trained and tested on the Callhome dataset using 5 folds cross validation. The results were verified with the published results and are demonstrated in Table 4.1 below:

Pipeline Settings	DER with overlap	DER without overlap
No resegmentation	9.04 %	18.85%
VB resegmentation%	6.52%	16.52%

Table 4.1: LSTM pipeline Evaluted on Callhome

4.5 Summary

In this Chapter, the LSTM based similarity matrix was reviewed by analysing its pipeline, methodology and training methods. Finally, the model was evaluated on the Callhome dataset and gave better results than the RPNSD.

Chapter 5

The Overlap Detection Module

5.1 Motivation

Kaldi's x-vector diarization system does not have an overlapped speech detection module and thus, consistently reports a higher DER when overlapped speech is taken into consideration. The contribution of overalpped speech is especially high in spontaneous conversations like meetings where multiple speakers are present and there is no fixed order of speaking.

5.2 Algorithm

Thus to evaluate overlapped speech, it is proposed that an overlap detection module be present in the pipeline after the SAD module, which labels speech segment as 1 if it contains overlapped speech, and 0 if it only contains 1 speaker. The SAD and overlap detection can also be combined into 1 module easily.

Once we have overlapped regions stored in a separate file, we let kaldi perform diarization as it usually does. We incorporate labelling overlapped speech as part of the post processing step. The algorithm to label assumes that overlapped speech occurs when 2 speakers are speaking around the same time as each other. Thus for a particular overlapped segment, it scans the final result produced by Kaldi and finds the 2 nearest speakers for that time period and labels the overlapped segment accordingly.[2]

5.3 Evaluation

The algorithm was tested on the callhome dataset and the results are listed in the table below:

Overlap	DER	False Alarm Error	Missed Error	Speaker Error
No Overlap detected	18.34 %	0%	10.1%	8.3%
Overlap detected included %	16.69%	0%	0.3%	16.4%

Table 5.1: Overlap Detection Algorithm Evaluated on Callhome

Note- the False Alarm is 0 since oracle SAD labels were used as we solely wanted to see the performance of the overlap detection module.

As seen the total DER reduces with overlap detection from 18.34 to 16.69. As for now we are working with oracle overlap labels, the miss error reduces considerably. The speaker error however shoots up which indicates there is confusion in speaker labelling. Our assumption of overlapped speech occurring near each other might be incorrect in this context.

5.4 Summary

In this chapter, we introduced a minor improvement over Kaldi's baseline by including an overlap detection module. The algorithm to control for overlapped speech using an overlap detection module decreased the DER,which is a positive sign. More research is needed into this matter since the inclusion of such a module reduces the Miss Error, but increases the Speaker Error.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This thesis has presented an overview of the Speaker Diarization Systems, analysed and verified recent research directions and proposed and implemented minor improvements. The main contributions of the thesis was the 2 improvements that were done over RPNSD and the regular Kaldi baseline.

In Chapter 3, the Region Proposal Network Model was discussed at length, with its key concepts and architectures. Areas of improvement were identified, with the need for a neural clustering module standing out. An experiment with SOMs was performed and found that they worked well for 2 speakers but performance deteriorated with increase in number of speakers.

In Chapter 4, the LSTM based Similarity Matrix was analysed, and implemented to see its results. The method performed better than the Kaldi pipeline it modified, and the massive reduction in DER is attributed to the LSTM learning context about the sequence while learning the similarity matrix in comparison to PLDA which just views 2 segments individually.

In Chapter 5, a major gap in Speaker Diarization Systems was called out- the lack of overlap detection. All systems consistently perform worse when scored for including overlapped speech. This is a problem since overlapped speech is a very natural occurrence in conversation and needs to be identified and dealt with.

To this effect, an overlap detection algorithm is proposed that works on the assumptions that speakers speaking in the overlapped segment also speak at other times near that segment. The results are promising as there is a 2% DER reduction when evaluating for overlapped speech.

In conclusion, from all the systems reviewed, the one described in Chapter 4- Kaldi's baseline with LSTM Similarity scoring matrix outperforms the others and gives the lowest DER on the Callhome Dataset.

6.2 Future Work

6.2.1 RPNSD

1. Merely rejecting proposals with overlap more than threshold might lead to 'bad' rejections, that is proposals with high confidence values representing some other part of speech also. To avoid this, Soft NMS can be applied where before rejecting, their confidence value can be scaled proportional to the amount of overlap and this value can be compared with a tuned threshold.

2. The Clustering algorithms used in the RPNSD systems are based on a quantitative distance measure. This might not represent the true similarity between speakers since it is a mathematical construct and some information might not be captured in the measure. Also since our embeddings are generated by a neural network and are not deterministic, this deepens the issue. To resolve the same, SOMS were introduced, but were found to be effective for only less speakers. Future works could aim to introduce a neural network module to transform the clustering problem to a supervised multi-label classification problem.

6.2.2 LSTM Similarity Matrix

1. Replace BLSTM with other neural network architectures like Transformer or add Attention. The motivation to do so is that these architectures capture larger context of information in a sequence, compared to BLSTM, which depends only on the previous cell.

2. Since x-vectors and similarity scoring matrix are now both neural networks, we could jointly optimize them as we train them together and find out if the results are better. The problem that might crop up here is that x-vectors generally require more data for training than we used for LSTM. For training x-vectors, we generally use speaker recognition data such as NIST SRE or Switchboard, which does not have diarization labels and hence cannot be used for LSTM. Thus training data might need to be artificially generated or augmented.

6.2.3 Overlap Detection

1. The overlap detection algorithm proposed here demonstrates promising results for including and dealing with overlapped speech. However, it is not the most optimal at its task. Thus, a new overlap detection algorithm can be worked on, which outputs the second most probable speaker from a segment which has been labelled overlapped. This could be done in the first pass itself or in the second pass, along with resegmentation.

Bibliography

[1] Herve Bredin et al. “Pyannote.Audio: Neural Building Blocks for Speaker Diarization”. In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2020). DOI: 10.1109/icassp40776.2020.9052974. URL: http://dx.doi.org/10.1109/ICASSP40776.2020.9052974.

[2] D. Charlet, C. Barras, and J. Liénard. “Impact of overlapping speech detection on speaker diarization for broadcast news and debates”. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. 2013, pp. 7707–7711. DOI: 10.1109/ICASSP.2013.6639163.

[3] Mireia Diez, Lukas Burget, and Pavel Matejka. “Speaker Diarization based on Bayesian HMM with Eigenvoice Priors”. In: *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*. 2018, pp. 147–154. DOI: 10.21437/Odyssey.2018-21. URL: http://dx.doi.org/10.21437/Odyssey.2018-21.

[4] Mireia Diez et al. “BUT System for DIHARD Speech Diarization Challenge 2018”. In: Sept. 2018, pp. 2798–2802. DOI: 10.21437/Interspeech.2018-1749.

[5] Yusuke Fujita et al. “End-to-End Neural Speaker Diarization with Permutation-Free Objectives”. In: *Interspeech 2019* (2019). DOI: 10.21437/interspeech.2019-2899. URL: http://dx.doi.org/10.21437/interspeech.2019-2899.

[6] Yusuke Fujita et al. “End-to-End Neural Speaker Diarization with Self-Attention”. In: *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (2019). DOI: 10.1109/asru46091.2019.9003959. URL: http://dx.doi.org/10.1109/ASRU46091.2019.9003959.

[7] Jürgen Geiger, Frank Wallhoff, and Gerhard Rigoll. “GMM-UBM based open-set online speaker diarization”. In: Jan. 2010, pp. 2330–2333.

[8] Ross B. Girshick. “Fast R-CNN”. In: *CoRR* abs/1504.08083 (2015). arXiv: 1504.08083. URL: http://arxiv.org/abs/1504.08083.

[9] Zili Huang et al. "Speaker Diarization with Region Proposal Network". In: *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*. IEEE, 2020, pp. 6514–6518. DOI: 10.1109/ICASSP40776.2020.9053760. URL: https://doi.org/10.1109/ICASSP40776.2020.9053760.

[10] Patrick Kenny et al. "A Study of Interspeaker Variability in Speaker Verification". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 16 (Aug. 2008), pp. 980 –988. DOI: 10.1109/TASL.2008.925147.

[11] Qiujia Li et al. *Discriminative Neural Clustering for Speaker Diarisation*. 2019. arXiv: 1910.09703 [eess.AS].

[12] Qingjian Lin et al. *LSTM based Similarity Measurement with Spectral Clustering for Speaker Diarization*. 2019. arXiv: 1907.10393 [eess.AS].

[13] U. V. Luxburg. "A tutorial on spectral clustering". In: *Statistics and Computing* 17 (2007), pp. 395–416.

[14] S. Meignier and T. Merlin. "LIUM SPKDIARIZATION: AN OPEN SOURCE TOOLKIT FOR DIARIZATION". In: 2010.

[15] Daniel Povey et al. "The Kaldi speech recognition toolkit". In: *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding* (Jan. 2011).

[16] Shaoqing Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *CoRR* abs/1506.01497 (2015). arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

[17] G. Sell and D. Garcia-Romero. "Diarization resegmentation in the factor analysis subspace". In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2015, pp. 4794–4798. DOI: 10.1109/ICASSP.2015.7178881.

[18] G. Sell and D. Garcia-Romero. "Speaker diarization with plda i-vector scoring and unsupervised calibration". In: *2014 IEEE Spoken Language Technology Workshop (SLT)* (2014), pp. 413–417.

[19] Mohammed Senoussaoui et al. "An i-vector extractor suitable for speaker recognition with both microphone and telephone speech". In: *Proc. Odyssey Speaker and Language Recogntion Workshop, 2010* (Jan. 2010).

[20] David Snyder et al. "X-Vectors: Robust DNN Embeddings for Speaker Recognition". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*. IEEE, 2018, pp. 5329–5333. DOI: 10.1109/ICASSP.2018.8461375. URL: https://doi.org/10.1109/ICASSP.2018.8461375.

[21] Q. Wang et al. "Speaker Diarization with LSTM". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2018, pp. 5239–5243. DOI: 10.1109/ICASSP.2018.8462628.

[22] Aonan Zhang et al. “Fully Supervised Speaker Diarization”. In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2019). DOI: 10.1109/icassp.2019.8683892. URL: http://dx.doi.org/10.1109/ICASSP.2019.8683892.