REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DESIGN OF SPEAKER DIARIZATION WITH SPEAKER EMBEDDINGS

Muhammet Mesut TORUK

MASTER OF SCIENCE THESIS
Department of Electronics and Communications Engineering
Telecommunications Program

Advisor
Assoc. Prof. Dr. Ahmet SERBES

Co-Advisor
Assoc. Prof. Dr. Gökhan BİLGİN

July, 2020

REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DESIGN OF SPEAKER DIARIZATION WITH SPEAKER EMBEDDINGS

A thesis submitted by Muhammet Mesut TORUK in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE** is approved by the committee on 16.07.2020 in Department of Electronics and Communications Engineering, Telecommunications Program.

Assoc. Prof. Dr. Ahmet SERBES
Yıldız Technical University
Advisor

Assoc. Prof. Dr. Gökhan BİLGİN
Yıldız Technical University
Co-Advisor

Approved By the Examining Committee

Assoc. Prof. Dr. Ahmet SERBES, Advisor
Yıldız Technical University _____

Assoc. Prof. Dr. Hacı İLHAN, Member
Yıldız Technical University _____

Assist. Prof. Dr. Yusuf YASLAN, Member
İstanbul Technical University _____

I hereby declare that I have obtained the required legal permissions during data collection and exploitation procedures, that I have made the in-text citations and cited the references properly, that I haven't falsified and/or fabricated research data and results of the study and that I have abided by the principles of the scientific research and ethics during my Thesis Study under the title of Design of Speaker Diarization with Speaker Embeddings supervised by my advisor Assoc. Prof. Dr. Ahmet SERBES and my co-advisor Assoc. Prof. Dr. Gökhan BİLGİN. In the case of a discovery of false statement, I am to acknowledge any legal consequence.

Muhammet Mesut TORUK

Signature

Dedicated to my family

ACKNOWLEDGEMENTS

In recent years, various systems have been developed that offer automated solutions in speech technologies. In this study, the speaker diarization problem dealing with multiple speakers is discussed. The speaker diarization systems examine speech recordings containing multiple speakers. It can be used for diarization problems encountered in daily life such as phone records, television broadcasting records, meeting record analysis.

In the completion of this thesis, , I would like to express my thanks to my advisor Ahmet SERBES and my co-advisor Gökhan BİLGİN for their close attention and valuable help. I would also like to thank TUBİTAK for the data set and performance computing facilities as well as the working environment.

I would like to express my endless gratitude to my family for every support they have provided to me.

Muhammet Mesut TORUK

TABLE OF CONTENTS

LIST OF SYMBOLS

$\sum$	Covariance matrix
dB	Decibel
Δx_m	Derivative of input feature vector
$\in$	Element of
H_d	Hypothesis of belonging to the different speaker
H_s	Hypothesis of belonging to the same speaker
x_m	Input feature vector
μ	Mean vector
ms	Milliseconds
$f(.)$	Model function
y_m	Model output
o	Observation
γ	Posterior Probability
$\sigma(.)$	Sigmoid function
ω	Speaker embedding
η	Threshold
w	Weight of component

LIST OF ABBREVIATIONS

ACAM	Adaptive Context Attention Model
AHC	Agglomerative Hierarchical Clustering
ASR	Automatic Speech Recognition
bDNN	boosted Deep Neural Network
DCT	Discrete Cosine Transform
DER	Diarization Error Rate
DNN	Deep Neural Network
EM	Expectation-Maximization
FA	False Alarm
FNN	Fully-connected Neural Network
GE2E	Generalized End-to-End
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
LSTM	Long Short Term Memory
MAP	Maximum a Posteriori
MFCC	Mel Frequency Cepstral Coefficient
MLP	Multi Layer Perceptron
MRCG	Multi-Resolution Cochleagram
MS	Missed Segment
MUSAN	Music, Speech, and Noise
NIST	National Institute of Standards and Technology
OOV	Out of Vocabulary
PLDA	Probabilistic Linear Discriminant Analysis

PLP	Perceptual Linear Prediction
ReLU	Rectified Linear Unit
RIR	Room Impulse Response
RNN	Recurrent Neural Network
RTTM	Rich Transcription Time Marked
SE	Speaker Error
SNR	Signal-to-Noise Ratio
SPK	Speaker
SRE	Speaker Recognition Evaluation
TDNN	Time-delayed Deep Neural Network
UBM	Universal Background Model
VAD	Voice Activity Detection
VB	Variational Bayes

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Design of Speaker Diarization with Speaker Embeddings

Muhammet Mesut TORUK

Department of Electronics and Communications Engineering
Master of Science Thesis

Advisor: Assoc. Prof. Dr. Ahmet SERBES
Co-advisor: Assoc. Prof. Dr. Gökhan BİLGİN

In recent years, the amount of speech data has increased greatly with the development and spread of technology. Other than controlled filing, it is hard to make sure there is one speaker in an audio file. In these situation, the speaker diarization, is needed for labeling the identity of the speaker of the speech segments. This process does not need any information about the identity of the speakers. It can also be accomplished without knowing the number of speakers in the recorded speech.

The speaker diarization system consists of three main modules: i) speech segmentation, ii) speaker representations, and iii) clustering. In this study, different solution techniques for diarization modules through machine learning and deep learning algorithms are examined and implemented. The speaker representations are investigated via vector embedding methods. Finally, the clustering algorithm is applied to collect segments of the same speaker. The time boundaries of the speech segments are determined, and the silent gaps are ignored during the segmentation process. Voice activity detection system gets designed to find speech segments. This system makes binary classification for speech frames as speech/nonspeech, and frames containing the speech signal are detected.

The speaker representation module produces vectors containing speaker-specific information for each speech segment. Machine-learning based i-vectors are an important milestone in the speaker recognition field. Subsequently, better achievements are obtained with the advancement of deep learning-based methods.

These algorithms used in speaker recognition are applied to the speaker diarization problem in the same way. These developments also ensured having successful results for diarization problem. This study examines deep learning-based d-vectors and x-vectors and tests the system with xd-vectors consisting of a fusion of these two vectors.

In the clustering stage, the representation vectors which have been generated for each speech segment and contain information about the segment's speaker are used as an input. In order to measure the similarity of the segments, a similarity score is generated between each segment. Then the similarity score vectors generated for each segment are clustered. In this study, the situations with the known and the unknown number of speakers will be examined. As a parameter for clustering the number of speakers or the threshold value should be defined. The agglomerative hierarchical clustering method was used for clustering. Then, re-segmentation was performed to improve cluster outputs. The variational bayes method was used for this process.

In this thesis, performance comparisons of different diarization techniques are realized. The error metric used for this problem is diarization error rate (DER) and it consists of three values; false alarm, missed segment, and speaker error.

Keywords: Speaker diarization, speaker identification, speech segmentation, biometry, artificial neural networks

ÖZET

Konuşmacı Katıştırmaları ile Konuşmacı Günlükleme Tasarımı

Muhammet Mesut TORUK

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı
Yüksek Lisans Tezi

Danışman: Doç. Dr. Ahmet SERBES
Eş-Danışman: Doç. Dr. Gökhan BİLGİN

Son yıllarda konuşma verilerinin aşırı büyümesi sonucu ses işleme alanında bu verilerin analiz edilmesi için çözüm yolları aranmaya başlamıştır. Kontrollü belgelemeler dışında, bir konuşma dosyasında yalnızca bir konuşmacı olmasını ayarlamak zordur. Bu durumlarda, aynı konuşmacıya ait konuşma bölütlerini etiketliyen konuşmacı günlükleme işlemi yapılır. Bu işlem konuşmacıların kimliği ve kaç konuşmacı olduğu hakkında bilgi sahibi olmadan gerçekleştirilebilir.

Konuşmacı günlükleme problemi üç ana modülden oluşmaktadır: i) konuşma bölütleme, ii) konuşmacı temsili ve iii) kümeleme. Bu çalışmada günlükleme modülleri için makine öğrenmesi ve derin öğrenme algoritmaları ile çözüm önerileri sunacağız. Konuşmacı kimliklendirme için ise vektör katıştırma metodları incelenecek. Son olarak ise kümeleme algoritması uygulanacak ve aynı konuşmacıya ait bölütler bir küme olarak tanımlanacaktır. Konuşma bölütleme işlemi sırasında, konuşma dosyası için günlükleme işlemi yapılırken öncelikle konuşmanın geçtiği zaman dilimleri tespit edilir. Sessizlik alanları görmezden gelinir. Konuşma bölütlerini bulmak için konuşma etkinlik tespit sistemi tasarlanır. Bu sistem ses çerçeveleri için konuşma/konuşma değil olmak üzere ikili sistemde bir sınıflandırma yapar ve konuşma sinyali içeren çerçeveler tespit edilir.

Konuşmacı kimliklendirme modülü her bölüt için kişiye özel bilgiler içeren vektörler çıkarır. Konuşmacı tanıma alanında makine öğrenmesi tabanlı i-vektörler önemli bir kilometre taşıdır. Bu alanda daha sonra derin öğrenme tabanlı metodlarla daha iyi

başarımlar sağlanmıştır. Konuşmacı tanıma alanında kullanılan bu algoritmalar aynı şekilde konuşmacı günlükleme problemine de uygulanmıştır. Bu gelişmeler problem için daha başarılı sonuçlar elde edilmesini sağlamıştır. Bu çalışma kapsamında LSTM tabanlı d-vektörler ve zaman-gecikmeli derin sinir ağı tabanlı x-vektörler incelenmiştir. Daha sonrasında bu iki vektörün birleşiminden oluşan xd-vektörleri ile sistem test edilmiştir.

Kümeleme aşamasında her bölüt için çıkarılan ve bölütün konuşmacısına ait bilgiler içeren temsil vektörleri giriş olarak kullanılır. Bölütlerin birbirine benzerliğini ölçmek için her bölüt arasında benzerlik skoru üretilir. Daha sonra her bölüt için çıkarılan bu benzerlik skoru vektörleri kümelenir. Bu çalışmada konuşmacı sayısının bilindiği ve bilinmediği durumar incelenecektir. Kümeleme için parametre olarak konuşmacı sayısı veya eşik değeri tanımlanmalıdır. Kümeleme için yığınsal hiyerarşik kümeleme metodu kullanılmıştır. Daha sonrasında kümeleme çıktıları üzerinde iyileştirme yapmak için yeniden bölütleme işlemi yapılır. Bu işlem için değişimsel bayes yöntemi kullanılmıştır.

Bu tez kapsamında, günlükleme modülleri için kullanılan farklı yöntemlerin performans karşılaştırması yapılmıştır. Bu problem için kullanılan hata metriğinin adı günlükleme hata oranı'dır ve üç değerden oluşmaktadır; yanlış alarm, bölüt kaybı ve konuşmacı hatasıdır.

Anahtar Kelimeler: Konuşmacı günlükleme, konuşmacı kimliklendirme, konuşma bölütleme, biyometri, yapay sinir ağları

1
INTRODUCTION

1.1 Literature Review

Nowadays, with the overgrowth of speech files, the interest in the applications of speech technologies for searching, indexing and extracting information is increasing. The speaker diarization problem looks for answers to the question "who speaks when?", without prior knowledge of the speakers [1]. The system examines speech files containing multi-speakers and solution to this problem can bring solution to other problems such as speaker change analysis and multimedia information retrieval in the literature. In addition, this system can be used to improve automatic speech recognition (ASR) performance.

Speech signal is a periodical signal that changes relatively slowly with the vibration frequency of the vocal cords, which is called the pitch in voice speech. The contribution of the male to the pitch is usually between 50Hz and 250Hz, the female's contribution to the pitch is between 120Hz and 500Hz.

Speech can be phonetically represented by a set of symbols called phonemes of the language. The number of phonemes for most languages changes between 32-64 [2], [3]. The three consequent phonemes in a sequence is called a triphone. The structure of the phonemes and how they interact with each other are examined.

A speech file is consisted of two main parts: one part contains the speech information, and the other part is the nonspeech section, it does not contain any verbal information. Verbal information is produced by humans and aimed to be detected and analyzed. There are two types of verbal information: voiced and unvoiced speech. When humans speak, they use their air paths larynx; voiced sounds are generated when vocal cords are semi closed, and unvoiced sounds is generated when they are open. Unvoiced speech sounds are produced by air passing through vocal tract formations. Unlike voiced speech, unvoiced speech does not show periodicity and is characterized by a noise-like signal.

Speech files are divided by silence gaps. Since there is no stimulation to the vocal tract in the silence gaps, there is no speech output. Silence is an important part of the speech signal. Acoustic signals in quiet areas are basically background noise and mouth sound not related to speech [4].

A human identity is stored in his or her voice. The voice characteristic differs from person to person. Speaker characteristics can change with

- physical structure of humans.
- environment of humans.

Physical structure is important with organs which produce speech. There are differences in the size and shape of vocal tract. The environmental factors vary based on birthplace and where people are raised [5].

Let extracted speech segment labels are $\Psi = (\psi_1, \psi_2, ..., \psi_T)$. In here, there are T speech segments and it is aimed to assign a speaker ID for each segment. For example, in the $\Psi = (1,1,2,4,3,3,4)$ sequence, there are four different speakers. With the re-segmentation process, improvements are made by post-processing on the diarized file.

Neural embedder is trained to embed speaker information of speech segments into a D-dimensional space. In the embedding space, two segments x_i and x_j of the same speaker are expected to be close to each other. Other situation, different speaker speech segments embeddings are looked for to be far from each other.

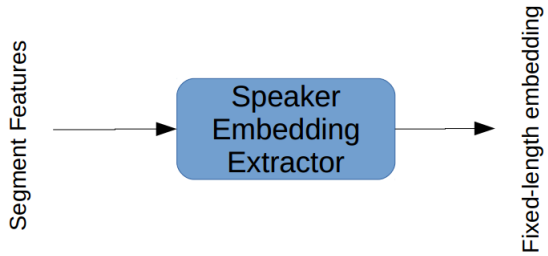

Figure 1.1 Basically, scheme of speaker representations with embedding

Fixed-size embedding vectors containing speaker-specific information are generated during speaker representation. The clustering phase is then responsible for aggregate

these segment vectors based on speaker information. Clustering algorithm takes as input the pair-wise similarities between all pairs of speech and gathers the resemblers.

Probabilistic linear discriminant analysis (PLDA) will be used to generate similarity scores between embedding vectors in the cluster module. With PLDA, a score expressing the similarity of each segment is obtained. Then, agglomerative hierarchical clustering (AHC) method is used to cluster these scores. When the number of speakers is known, the number of speakers is given as the number of clusters, and the threshold value is given when it is not known.

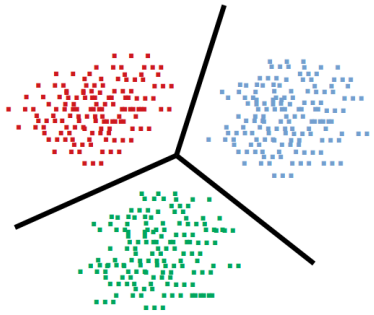

Figure 1.2 Simple definition of clustering in Machine Learning

1.2 Objective of the Thesis

Various speakers may have spoken at different time intervals within the speech files. As the length of the audio files increases, it becomes difficult to analyze these speech intervals. In these audio files, it may be difficult to listen to the whole file to determine the speaking times of the people. Speech containing segments and speaker information inference of these segments can be done automatically.

Speaker diarization has witnessed a major breakthrough in past several years with the development of various techniques suitable for modelling speaker characteristics. The introducing of i-vector gave a significant progress in the field of speaker modelling.

Each segment containing speech should be labeled with at least one speaker label, and segments from the same speaker shall be labeled with the same speaker ID. Speaker ID are not real identities but abstract labels. They are used to observe the differences between segment identities. Diarization technology may be used as front-end module for some other problems such as automatic speech recognizer, automatic speaker recognizer.

1.3 Hypothesis

Speech data can be recorded in different settings such as meetings, TV broadcasts and telephone conversations. Speaker diarization can help analyze these recordings. The speech file can be divided primarily from silence areas into speech segments. These speech segments may belong to different people. These short speech segments need to be assigned a speaker ID. Within the scope of this thesis, a speaker diarization system is developed determines the speaking time intervals of people in speech files. With this system, you can analyze large files faster than humans. In this way, you can save time and resources.

The speech signal is divided into short frames (20-25 ms) for processing. Short term features are extracted for these frames. These features are used for classification and identification. A decision (speech or nonspeech) is made for each frame during speech segmentation and a speech segment is formed. Also, speaker embedder models, which generate the fixed-size speaker vectors containing speaker-specific information for each segment, are trained using these frames.

Voice activity detection (VAD) that uses binary classification (speech/nonspeech) is run for speech segmentation. Deep Neural Network, Time-delayed Deep Neural Network, boosted Deep Neural Network and Adaptive Context Attention Model are used for speech detection. Gaussian mixture model, Time-delayed Deep Neural Network and LSTM are used for speaker representation. The agglomerative hierarchical clustering method is used in the clustering stage. The compendium to this thesis may be helpful for researchers working towards the deployment of speaker diarization systems for application-oriented services.

2
AN OVERVIEW: SPEAKER DIARIZATION PROBLEM

2.1 A Summary about Voice as a Biometric Data

Speaker recognition systems are used for recognizing humans from their voices. The voices of two people are different and can not produce same voice pattern, because they have different sound way shapes, throat length and a difference in other sound-making organs. In addition to these physical differences, every speaker has its speaking style such as accent, rhythm, toning style, pronunciation pattern, word choice. Most advanced speaker recognition systems use some of these features in parallel, tries to cover these different aspects and uses as a complement to get more accurate recognition.

2.2 Definition of Speaker Diarization

The speaker diarization problem investigates the question of "Who speaks when?" [6], [7]. It is used for daily life problems such as data indexing, meeting log analysis and it examines the situations where two or more speakers exists in one speech recording. It is used for daily life problems such as analysis of meeting recordings, telephone conversations, configuration of content of documents. In all these cases, it may be advantageous to automatically determine the number of speakers included in the audio file in addition to the time intervals during which the speakers are active. Applications that can be associated with speaker diarization algorithms include speech and speaker indexing, speaker recognition, and automatic speech recognition (ASR).

Basics of speaker related problems can be considered as speaker recognition or verification [8]. The speaker diarization problem is harder than the speaker recognition problem. Speech segment length being shorter in speaker diarization is a problem hard to resolve.

Various difficulties may arise in research in speaker diarization. Firstly, the dataset/corpus implemented to the speaker diarization system should be considered.

Another challenge to overcome is that the microphone type alters the recording quality. For example, broadcast news recordings are usually taken in the studio using boom or lapel microphones. In contrast, meetings are usually recorded using desktop or far-field microphones that are either more suitable for head-mounted or lapel microphones. There can be single microphones or microphone arrays. Accordingly, Signal-to-Noise Ratio (SNR) is generally better than meeting recordings for broadcast news data. Also, the differences between recording location configurations and microphone placement affect the recording quality; such as background noise, reverberation and variable speech levels. The distance of the speaker from the microphone is also an important factor. Speech recordings can be of read materials (such as news), or it can be more natural, spontaneous; and contains overlapping speech (like telephone, meeting recordings). Another challenge may be records containing speech that is overlapped with music, laughter, or applause. Having an unknown number of speakers or having too many speakers is also a serious challenge to consider. [9].

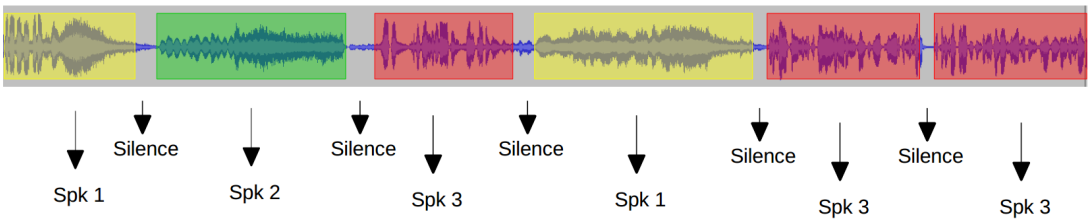

Figure 2.1 Speaker Diarization problem definition scheme

In Figure 2.1, speaker diarization system schematic is given. The purpose of the problem can be seen in the schematics. It is observed that the speech file consists of speech segments of different speakers and silence parts. These recordings should not contain overlap speech segments.

The speaker diarization problem includes three main modules; speech segmentation, speaker representation, clustering.

Figure 2.2 Flowchart of main modules of speaker diarization system

In Figure 2.2, the modules of the diarization system are given. This figure defines the problem in its most basic form. Usage and methods of these modules will be examined in detailed in the section III.

2.3 Speech Segmentation

The speech signal has a non-stationary form. It consists of segments and segments consist of frames (20-25 ms). The purpose of dividing the signal into short audio frames is to put audio signals into statically stationary form and the signal becomes more predictable. The speech segmentation process is used for dividing audio file into small pieces by silence segments. Speech segment boundaries are determined by silence segments and pauses in the audio file.

Voice activity detection (VAD) may be used as front-end module in different speech processing problems, (such as coding, enhancement and recognition, etc.). There are many different approaches reported in the literature. Usually, it is a binary classification problem, which classify speech/nonspeech parts of audio signal.

First step to diarization attempt is to detect voice activity, that is, by having speech segments. Nonspeech segments in audio files can include non-lexical ambient noise such as laughing, silence, coughing and breathing. Each recording can contain different types of background noise. Thus, the energy level of the signal may vary in nonspeech segments. Therefore, the VAD module appears to be critical, and typical techniques based on feature extraction with a threshold-based decision have proven to be relatively ineffective. It is observed that VAD based typical feature extraction techniques (like energy, spectrum differences, and pitch estimation) based on threshold is ineffective.

Short-time spectral features are extracted for every single frame (20-25 ms) in order to speech processing (like MFCC, gammatone, cochleagram), $\mathrm{x} \in \chi$: $\mathrm{x} = (\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_T)$. In VAD system, a decision is made for every frame, $\mathrm{y} = (\mathrm{y}_1, \ldots, \mathrm{y}_T)$, with class count K=2; y_i=1 is speech, y_i=0 is non-speech. Given a classification problem, an observation **o** is predicted to belong to the class whose corresponding output node is assigned 1. The output c_k $(k = 1, 2)$, is calculated with Equation (2.1).

$$c_k = \begin{cases} 1, & \text{if } s_k > s_i, \quad \forall i = 1,\ 2,\ i \neq k \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

where s_k is the probabilistic soft output of the event "$c_k = 1$".

VAD only contains two classes (K=2; speech/non-speech). In Equation 2.2, prediction function of the VAD is given. According to η, it makes a decision about frame information it contains. In here, observation represents with **o**.

$$f(\mathbf{o}) \triangleq s_2 - s_1 \underset{H_d \in H_0}{\overset{H_d \in H_1}{\gtrless}} \eta \tag{2.2}$$

After the frame decisions are made, frames that belong to the same class are collected together to obtain speech segments. After that smoothing process is made to frame decisions that are too small to represent a class. End of the post-processing, speech segments are obtained as an output.

In the scope of this study, the performance of different VAD algorithms and how they affect the speaker diarization system performance are examined. Deep Neural network, Boosted Deep Neural Network, Adaptive Context Attention Model and Time-delayed Deep Neural Network based VAD methods have been implemented. DNN, bDNN and ACAM system works with threshold. TDNN makes a decision based on the maximum value of the output nodes. And these modules explained in detail in the section III.

2.4 Speaker Representation

Speaker recognition algorithms try to recognize speaker identity using voice biometrics. Basically, the most known recognition problems are speaker verification (determining whether the speaker is the claimed person in the speech file) and speaker recognition (identification among a group of speakers in the unknown speech file). These are most popular and most used algorithms [2].

Every speech segments representation with a fixed-size vector including information of speaker after speech segment boundaries are determined by using VAD. This process is called speaker representation. Segment length does not change the size of the embedding vector here.

Speaker embedding features are learned as classifiers for identification in a training set, and it is taken from hidden layer activations of networks. These embedding features keep information about speaker identity [10].

In Figure 2.3, it shows how the network processes fixed-size speech sequences to embed different-length speech sequences produced from the speech segmentation. For this process, slide a fixed-length window over the the speech record, embed each of these subsegment, and then aggregate the embeddings of all overlapping subsegment to get an embedding per speech segment [11]. The embedding of segment i is denoted as ω_i.

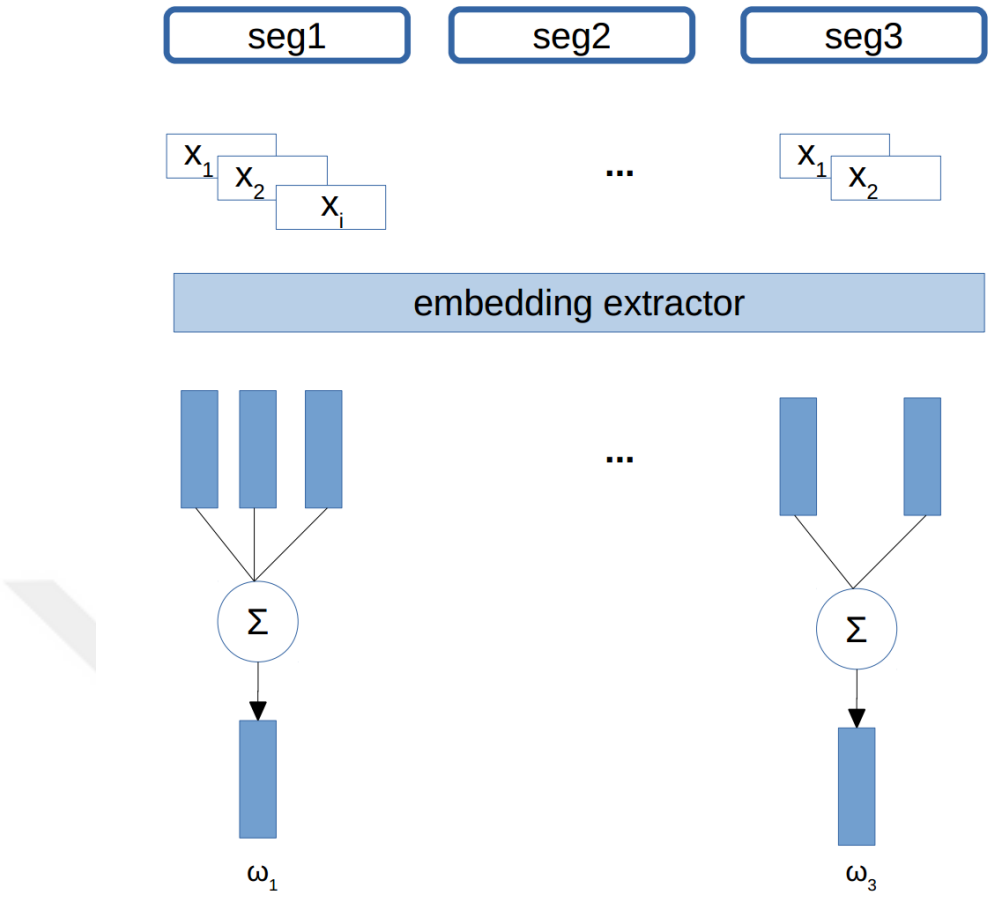

Figure 2.3 A brief summary, about segment-based speaker embeddings extraction. We can see that extracted subsegment embeddings and agglomerated these embeddings. Then, speaker-specific embedding vector is obtained

Deep Learning based speaker representation systems usually use a pooling mechanism to map different-length speech files to fixed-size embeddings. In a standard neural network architecture, it is usually done by a pooling layer that means some frame-level neural network layer features over the speech signal. Speaker embeddings are obtained using segment-level features.

In this study, i-vector, x-vector, d-vector and xd-vectors are implemented for speaker representation and its effect on diarization performance is examined. The embeddings for speaker representation are made with i-vectors based Gaussian Mixture Model, x-vectors extracted using Time-delayed Deep Neural Network, d-vectors extracted using LSTM and xd-vectors which is fusion of x-vector and d-vector.

2.5 Clustering

Every speech segment in the speech signal represented by a fixed-size vector. At the clustering stage, segments spoken by same speaker is aimed to be assigned to a speaker identity. In situations where the number of speakers is unknown in the speech file, the estimation of the number of speakers is made in this stage.

Clustering is very important as it determines the internal grouping between existing unlabeled data. According to the problem, the criteria of the clustering process are determined. [12].

Every speech segment in the speech signal represented by a fixed-size vector. At the clustering stage, segments of the same speaker is aimed to assigned to a speaker identity. In situations where the number of speakers is unknown in the speech file, the estimation of the number of speakers is made in this stage.

In Equation (2.3), the similarity between i^{th} and j^{th} speech segment embeddings can be expressed as,

$$s(i;j) = similarity(\omega_i;\omega_j) \tag{2.3}$$

where ω_i and ω_j are fixed-size embeddings and $s(i;j)$ is similarity score between these embeddings.

By examining the relationship of each segment with other segments, a vector representing its similarity with other segments is created. For example; for the first segment $S = [s(0;0),\ s(0;1),\ \dots,\ s(0;T)]$ where T is segment count. These similarity score vectors are then used as inputs for clustering. Each similarity score vector is assumed a data point. In this study, agglomerative hierarchical clustering (AHC) method is used for clustering. In cases where the number of speakers is unknown, the number of clusters is determined using threshold.

3
FEATURES AND ALGORITHMS USED IN SPEAKER DIARIZATION

The modules forming the speaker diarization problem were introduced in the previous sections. In this section, information about the speech signal will be given. Then, the features used in this study and the network algorithms used in the modules of diarization will be explained in detail.

3.1 A Summary about Speech Feature Types

There are many features that represent the characteristics of speech in the field of speech processing. These features are used according to the purpose of the problem. Features that keep speaker information in the speech signal are important for this problem. For discrimination of speaker; features

- should have big between speaker class differences and small within speaker class differences.
- would be minimally affected by noise and distortion.
- easy to measure while collecting data.
- would be robust for impersonation.
- should not change according to the conditions of the speaker.

Another important point about features, the number of required training samples for reliable recognition grows exponentially with the number of features. If you deal with a machine learning problem, optimization of the number of features may need sometimes. This issue is a usual problem in machine learning and it is known as the curse of dimensionality [13]. Low-dimensional features reduce calculations that you need to do.

Categorization of speech features in different ways is shown in the figure . According to their physical interpretation, it can be divided as follows;

1. short-term spectral features,
2. voice source features,
3. spectro-temporal features,
4. prosodic features,
5. high-level features

Short-term spectral features are extracted from speech frames (about 20–25 ms) in speech signals. These often define the short-term spectral envelope, which is an acoustic correlation of timbre (like the color of sound) as well as the resonance properties of the supralaryngeal vocal tract. Spectral features characterize the source (glottal flow) of the structure from which voice is produced. Prosodic and spectro-temporal features are extracted from larger frames (features such as toning and rhythm). Finally, high-level features, such as how speakers use words. The way everyone says "apple" is different.

The complexity of features increases from short-term to high-level. High-level features have a more high computational cost. Short-term spectral features are used to extract high-level features. Simple features for simple tasks, complex features for difficult tasks.

In the Figure 3.1, physiological features are those related to the physical structure of humans. It differs according to the individual's voice producing organs (like size of vocal folds, length of the vocal tract). Learned-based features are related to the environment in which the person grows and resides (education, place of birth, personality type, etc.).

How to decide which feature to use? The features to be used are determined for the problem to be solved. Firstly, short-term spectral features are extracted in the speaker identification algorithms because their extraction is easy. Then, high-level features that keep the speaker identity are extracted. Also, high-level features are more robust. However, they are less discriminative. As a result, the feature to be used is decided according to the needs of the problem. The trade-off must be adjusted correctly [14].

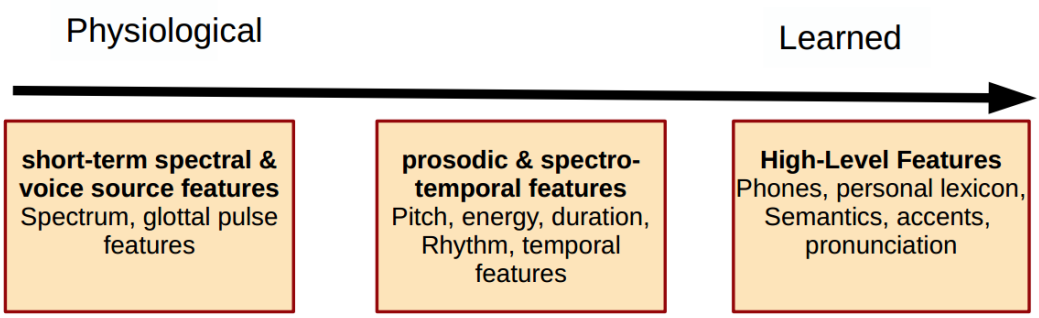

Figure 3.1 A summary of categorization of features according to their physical character. High-level features are learned for complex tasks

3.2 Speech Segmentation

Speech segmentation is used for dividing long duration audio files into speech segments that are shorter from the silence parts. This process aims to obtain the speech segments that belong to single speaker. Within this thesis, four different VAD method used for speech segmentation: TDNN, DNN, bDNN, and ACAM [15].

Binary Classification:

The binary classification method divides data points into two separate classes. There are two feedbacks that the computer can give by binary classification, "1" and "0". For example, there is a dataset with 100 utterances. On this dataset, classification can be applied as "speech" and "nonspeech". This classification will classify and label our data in two separate classes. In this study, deep learning methods will use to do binary classification. And their success in finding speech segments will be examined.

Vector cells showing frames indicate speech information. These frames form a speech signal. They do not represent sufficient information to classify. The frames need to be put into a form that can apply deep learning algorithms. For this, short-term spectral features of the speech are extracted from the frames. The name of this format is "Feature Vector". The purpose of this format is to store values that express the features of a speech numerically, in a vector, in a format suitable for processing.

In a train dataset, a single training utterance looks like this; (X, Y). Here, X is a matrix of mx-dimensional input data, while Y is an output result with the option to be 0 or 1. And m is number of frames. Number of features are equal in each frames. To show "m" training examples our notation will be as follows;

$$M = \{ (x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots (x^{(m)}, y^{(m)}) \} \tag{3.1}$$

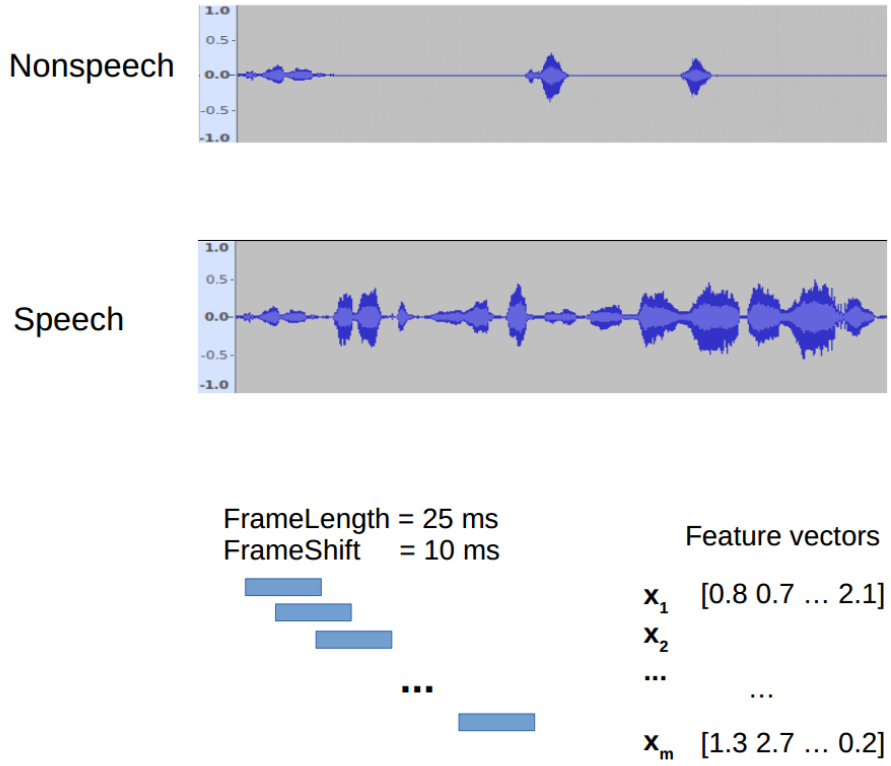

Figure 3.2 A summary of binary classification where classes are speech/nonspeech. We can see framing process to decide for belonging which class. In here, each frame can be assume as a small speech samples. And short-term spectral features are extracted for each frame

3.2.1 Time-delay Deep Neural Network

TDNN structure is similar to the DNN which described in section 3.2.2. The biggest difference between these two structures is TDNN has context-dependent form. Faster models can be produced by canceling some nodes on the TDNN layers. All nodes of the DNNs are fully-connected. TDNNs, however, takes network input features in a narrower context and has different resolution on every layer. It has a wider context at high layers [16].

TDNN models long term temporal dependencies by using short-term features (like MFCC). The activation that are calculated on each layer for all steps shown are in Figure 3.3. In this figure, it can be seen resolution of each layer. The system became faster because calculation parameters are selected during training with sub-sampling. In Figure 3.3, if one node is connected every node at the next layer, it means there is no sub-sampling. System speed without sub-sampling is 10t, when sub-sampling exists speed is 2t. The system becomes five times faster. These tests were done by Peddinti et al. in 2015 [17].

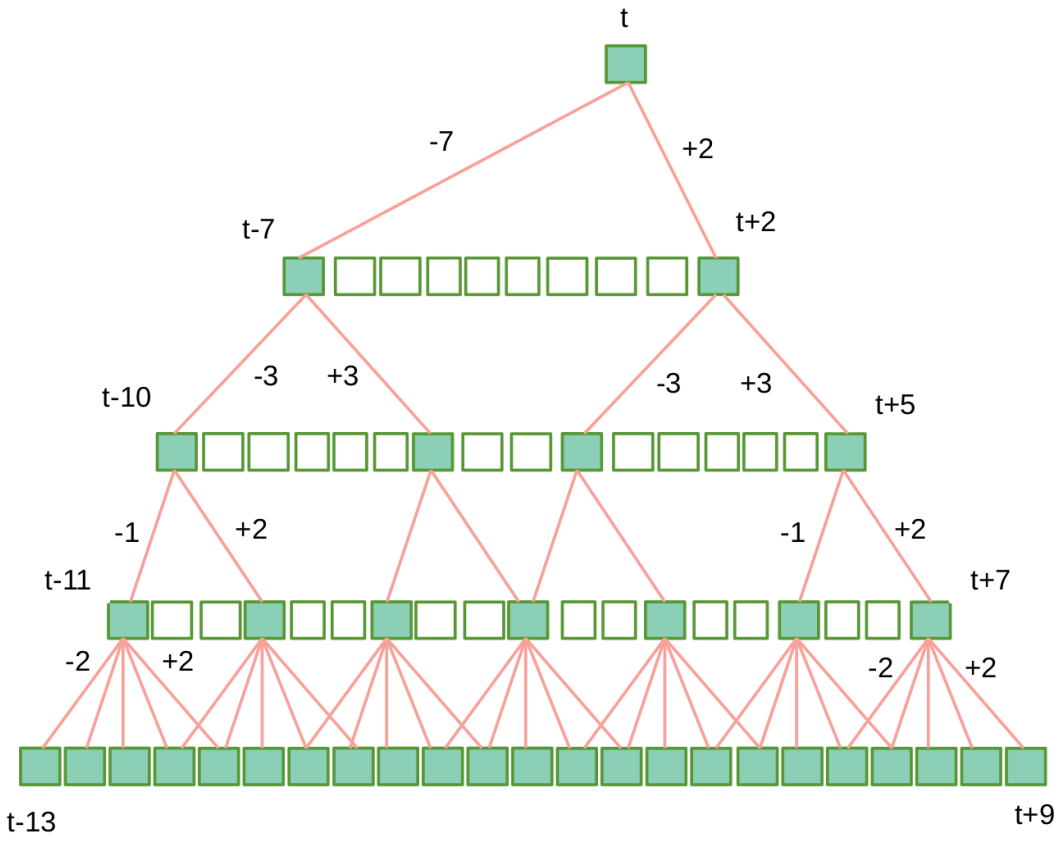

Figure 3.3 An example for TDNN structure. In here, TDNN's context-dependent form can be seen

Table 3.1 An Example TDNN model context structure

Layer	Input Context	Input context with sub-sampling
1	$[-2, +2]$	$[-2; +2]$
2	$[-1, +2]$	$\{-1; +2\}$
3	$[-3, +3]$	$\{-3; +3\}$
4	$[-7, +2]$	$\{-7; +2\}$
5	$\{0\}$	$\{0\}$

TDNN attributes:

- It is important to have an efficient connection between layers and nodes. This is important while having nonlinear decisions.
- Network has ability of having time-wise relation between events.
- It can effectively learn the temporal transitions of the signal from features.
- Training procedure does not require precise temporal alignment of the labels.
- They have fewer parameters.

The input context of each layer is required to calculate an output activation in one step. In Figure 3.3, it is shown the time steps when activation are calculated in each layer. Also, dependencies can be seen between nodes in each layer. It can be seen that dependencies between layers are localized over time. An example context table are given for more clear explanation. The layer context of TDNN is shown in third column of Table 3.1. Where "{}" symbol is used, it refers to sub-sampling. The symbol "[]" is used for the without subsampling version.

3.2.1.1 Mel Frequency Cepstral Coefficient

Mel Frequency Cepstral Coefficents (MFCCs) were presented in the 1980's. And they are most used features in the speech technologies [18]. In the previous sections, the importance of the physical form of the vocal tract is mentioned to extract information from speech signal. Shape of the vocal tract determines what sound comes out. If the features appropriate for this shape are extracted, it can develop more successful recognition systems. The shape of the vocal tract can be seen in the envelope of the short-term power spectrum, and MFCCs try to accurately represent this envelope.

An explanation of the MFCC extraction will be given step by step. MFCCs will combine with their deltas after MFCC extraction. Briefly, MFCC steps are as followings;

1. Firstly, speech signal are divided into short speech frames.
2. Calculate the periodogram estimate of the power spectrum for each frame.
3. Apply the mel filterbank to the power spectrum and sum the energy in each filter.
4. Log of all filterbank energies is taken.
5. Calculate the discrete cosine transform (DCT) of the logarithmic energies.
6. Discard the coefficients except 2-13 (or whichever is used).

For some problems, delta and delta-delta of MFCC can also be used. It is used by adding MFCC and deltas end to end. Delta features are derived from MFCCs. It is easy to extract, its effect on system performance is important. Calculation of the delta features is done by as in Equation (3.2).

$$d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N} n^2} \tag{3.2}$$

where d_t is a delta coefficient, t is speech frame count, cepstral coefficients are given as c_{t+n}, c_{t-n}. Generally, N is equal to 2. Delta-delta a.k.a acceleration coefficients are calculated from the delta coefficients.

The mel scale is about people perceiving different frequencies at different levels. People perceive small frequency changes at low frequencies better than changes at high frequencies. This scale allows people to better represent what they hear.

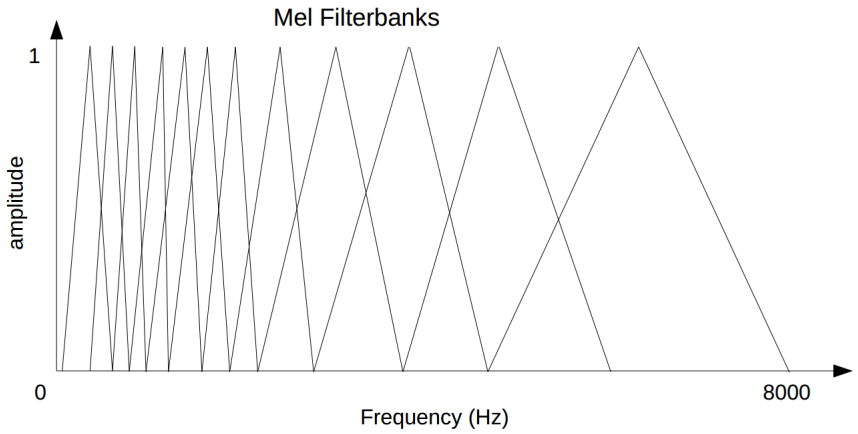

Figure 3.4 An illustration for mel filterbanks. They are wider at the high frequencies

The conversion formula from frequency to mel scale is:

$$M(f) = 1125\ ln(\ 1\ +\ f/700\ ) \tag{3.3}$$

The conversion formula from mels to frequency:

$$M^{-1}(m) = 700\ (\ exp(\ m/1125\ )\ -\ 1) \tag{3.4}$$

3.2.1.2 TDDN based Voice Activity Detection

The VAD module is simply a binary classifier. The nonspeech class can contain non-human sounds such as silence, noise, and music. Also, it may include nonspeech human voice such as laughter and cough, these sounds are labeled as garbage phones. Kaldi toolkit where codes are taken named this module as speech activity detection (SAD). Also SAD name may use sometimes in the literature.

This VAD model is phone recognizer based. For this, acoustic model based on Gaussian Mixture Model (GMM) is trained first. This acoustic model calculates the $\gamma_k^{(i)}$ posterior probability of the triphone k with the x_i features in the i^{th} frame. It is a conditional probability of k given x_i and λ. In Equation (3.5), the posterior probabilities of the phones of each of the classes are given.

$$\gamma_k^{(i)} = P(k|x_i, \lambda) \tag{3.5}$$

where λ is acoustic model. Firstly, a GMM based acoustic model is trained. And posterior probabilities was observed from this GMM. A TDNN model is initialized using these posterior probabilities and TDNN acoustic model is trained with using same dataset. After the TDNN acoustic model is trained, posterior probabilities are now updated using the trained TDNN model. This model is TDNN-based acoustic model. It calculates posterior probabilities corresponding to triphones with observed acoustic features. This model describes probabilities that correspond triphones, and the output node number of the TDNN model and the number of GMM components are equal.

A TDNN classifier with an output size of 3 is trained using a TDNN based acoustic model. These classes are: silence phones, speech phones and garbage phones. Here the garbage phone class includes sounds such as OOV (out of vocabulary), laughing, crying. The acoustic model trained in the previous stage plays a key role in the recognition of garbage phones here. In the VAD system, silence phones and garbage phones are labeled "0" and speech phones are labeled "1".

3.2.2 Deep Neural Network

Deep Neural Networks (DNN) have more than one hidden layer. Network learns more abstract representations from the training data with more layers of nonlinear transform of data. Generally, deep learning models use the back-propagation algorithm for training [19]. DNN models have fully-connected structure.

The DNN model is trained for binary classification with Multi-Resolution Cochleagram (MRCG) feature. Speech/nonspeech labels of frames are used for training. MRCG features will be explained in the section 3.2.2.1. Acoustic features and their labels representation is $\{(x_m, y_m)\}_{m=1}^{M}$. To use context information a wide frame $\{-W, -W+u, -W+2u, ..., W-2u, W-u, W\}$ is produced with neighbor frames. In here, DNN window length is equal to $2(W-1)/u+3$. And DNN input size equal to {(number of frames) x (DNN window length)}.

$$\begin{aligned} x'_m = [x^T_{m-W}, x^T_{m-W+u}, \quad ..., \quad x^T_{m-1-u}, x^T_{m-1}, x^T_m, x^T_{m+1}, x^T_{m+1+u}, \\ ..., \quad x^T_{m+W-u}, x^T_{m+W}]^T \end{aligned} \tag{3.6}$$

In Figure 3.5, general scheme of DNN is given. Also, it can be named as multi layer perceptron (MLP). In Equation (3.7), DNN formulation can be seen;

$$\hat{y}_m = f_{(L)}(\quad \dots \quad f_{(l)} \quad (f_{(2)}(f_{(1)}(x'_m)))) \tag{3.7}$$

where l denotes the hidden layer number, and x_m is the input feature vector.

$$\hat{y}_m = f\left(\begin{bmatrix} x_{m-W} \\ . \\ . \\ . \\ x_{m+W} \end{bmatrix}\right) \tag{3.8}$$

DNN model makes a prediction with using context information. And it makes a decision with threshold η as follows;

$$\bar{y}_m = \begin{cases} 1, & \text{if} \quad \hat{y}_m \geq \eta \\ 0, & \text{otherwise} \end{cases} . \tag{3.9}$$

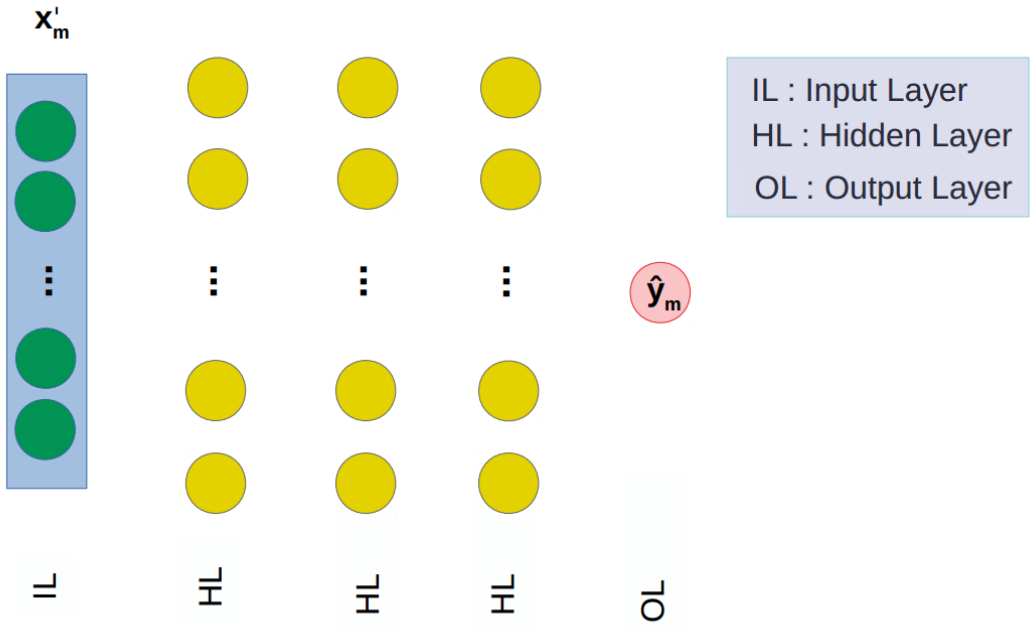

Figure 3.5 A general DNN structure for binary classification. There is no subsampling, it is fully-connected network

3.2.2.1 Multi-Resolution Cochleagram Feature

Cochleagram Features: In Figure 3.6, the extraction of the cochleagram feature is shown. Firstly, input speech signal is filtered by the 64-channel gammatone filterbank. Secondly, calculate the energy of each time-frequency frame for each channel separately. And finally rescale the energy with using $log_{10}(.)$.

Gammatone filter can be produced with multiplication of a gamma distribution and sinusoidal signal. This feature gave good results in noisy environment [20]. It is introduced in 1980 [21]. It is a widely used in the speech processing system [22]. The filter equation is given in the time domain as the following:

$$h(t) = k.t^{n-1}.e^{-2\pi bt}cos(2\pi f t + \phi) \tag{3.10}$$

where k is the amplitude, t is time, n is the filter order, b is the bandwidth of filter, f is frequency and ϕ the phase.

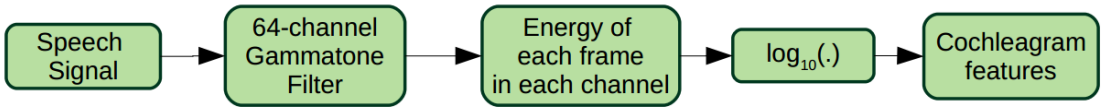

Figure 3.6 Flowchart of cochleagram feature extraction

Multi-resolution cochleagram (MRCG) is to combine both local information and global information by extracting with multi resolutions [23]. Local information is obtained by extracting a cochleagram from small frames (high resolutions). Global information is obtained by extracting a cochleagram from large frames (low resolutions). Using the cochleagram feature at different resolutions reveals the term multi-resolution. Frame lengths can be adjusted as 20 ms and 200 ms. In the studies done, the use of different resolutions has had positive effects on system performance.

As shown in the Figure 3.7, MRCG feature is formed by the combination of 4 cochleagram feature, which are extracted in different resolutions. Cochleagram feature are generated from 64-channel gammatone filterbanks. A high-resolution cochleagram is flattened with a time-frequency unit (with 11x11 and 23x23 filters) to extract 2 more cochleagrams. Finally, a low-resolution cochleagram is extracted and a 256-dimensional MRCG feature is obtained. After calculating the MRCG feature, its delta and delta-delta are calculated. Then all three features are combined into a 768-dimensional feature. Delta feature is extracted with Equation (3.11):

$$\Delta x_n = \frac{(x_{n+1} - x_{n-1}) + 2(x_{n+2} - x_{n-2})}{10} \tag{3.11}$$

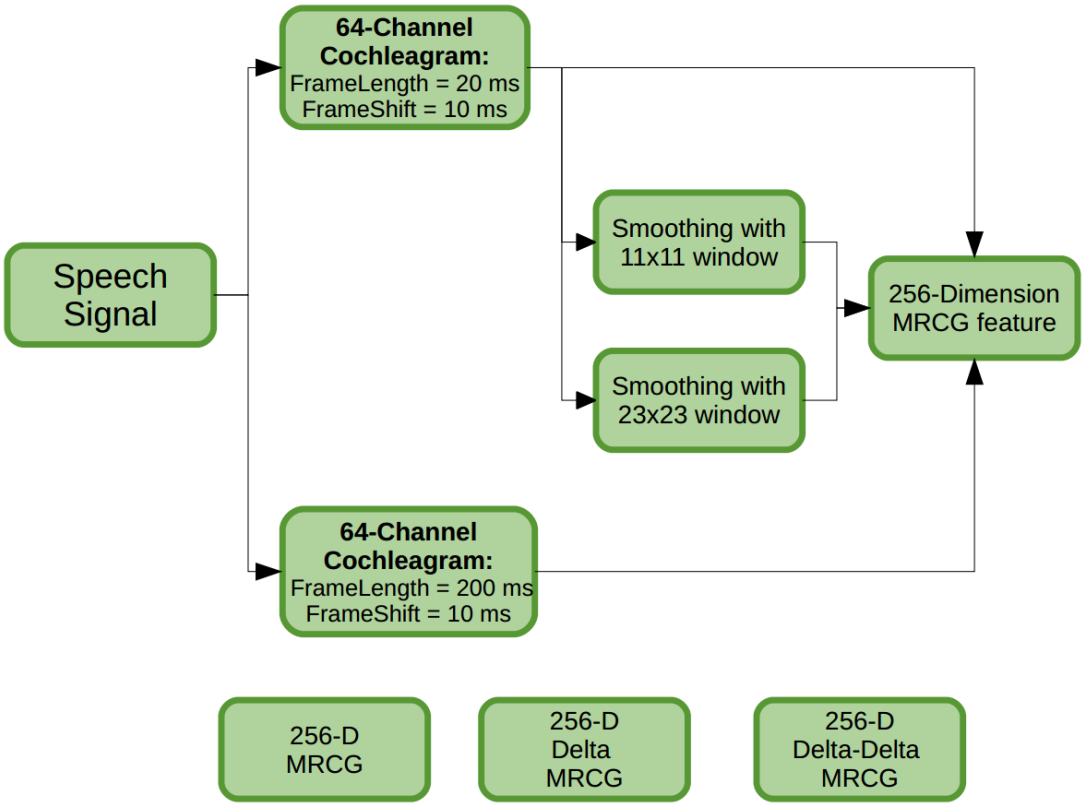

Figure 3.7 Illustration of MRCG feature extraction, and expanding MRCG with delta and delta-delta

3.2.3 Boosted Deep Neural Network

Real-world problems are modeled with machine learning or deep learning algorithms in the literature. The algorithms used in this modeling are chosen according to the form of the training data of the problem or by empirical methods.

In ensemble learning, instead of using a single base learner, it is the creation of a new model by using multiple modeling algorithms together. It can be applied to machine learning problems. There are different kinds of ensemble learning techniques: changing features, training data points, output targets and hyperparameters of classifiers.

The boosting process is a method of ensemble learning. One of the ensemble learning techniques is to train different models for the same decision of input. A stronger model is created by combining several weak models. Thus, the accuracy of the decision is confirmed by using more than one classifier for the decision. But this is an expensive method as a process cost.

The base learner is used in ensemble learning. In this task, fully-connected feed-forward DNN is used as base classifier. Boosted Deep Neural Network (bDNN) makes multiple predictions with the help of context information for one frame [20].

For clear explanation, dataset is represented with $\{(x_m, y_m)\}_{m=1}^{M}$. To use context information a wide frame $\{-W, -W+1, -W+2, ..., W-2, W-1, W\}$ is created with neighbor frames. MRCG features are used as input for classification.

The steps to predict frame labels using bDNN are as follows: expansion of input features x'_m, output prediction with $(2W+1)$-dimensional, getting the soft prediction of x_m.

Expansion of input features can be seen in Equation (3.12). Network takes input $(2W+1)$-dimensional frame including its neighbours.

$$x'_m = [x^T_{m-W}, x^T_{m-W+1}, \quad ..., \quad x^T_{m-1}, x^T_m, x^T_{m+1}, \quad ..., x^T_{m+W-1}, x^T_{m+W}]^T \tag{3.12}$$

Network makes $(2W+1)$-dimensional predictions for given x'_m. This prediction is given in Equation (3.13). In Figure 3.8, stages of this predictions can be seen. In here, DNN are used as base classifier. DNN was mentioned above.

$$y'_m = [y^{(-W)}_{m-W}, y^{(-W+1)}_{m-W+1}, \quad ..., \quad y^{(0)}_m, \quad ..., y^{(W-1)}_{m+W-1}, y^{(W)}_{m+W}]^T \tag{3.13}$$

The bDNN generates a prediction with using a DNN classifier. In Equation (3.14), $g(.)$ corresponds to DNN function. There is multiple base predictions for each frame. Multiple predictions can be seen as followings;

$$\begin{bmatrix} y^{(W)}_{m-W} \\ . \\ . \\ . \\ y^{(W)}_{m+W} \end{bmatrix} = g\left(\begin{bmatrix} x_{m-W} \\ . \\ . \\ . \\ x_{m+W} \end{bmatrix} \right) \tag{3.14}$$

Finally, results are aggregated. And soft prediction $\hat{y}_m$ is obtained as followings;

$$\hat{y}_m = \frac{\sum_{w=-W}^{W} y_m^{(w)}}{2W+1}, \quad \bar{y}_m = \begin{cases} 1, & \text{if} \quad \hat{y}_m \geq \eta \\ 0, & \text{otherwise} \end{cases} . \tag{3.15}$$

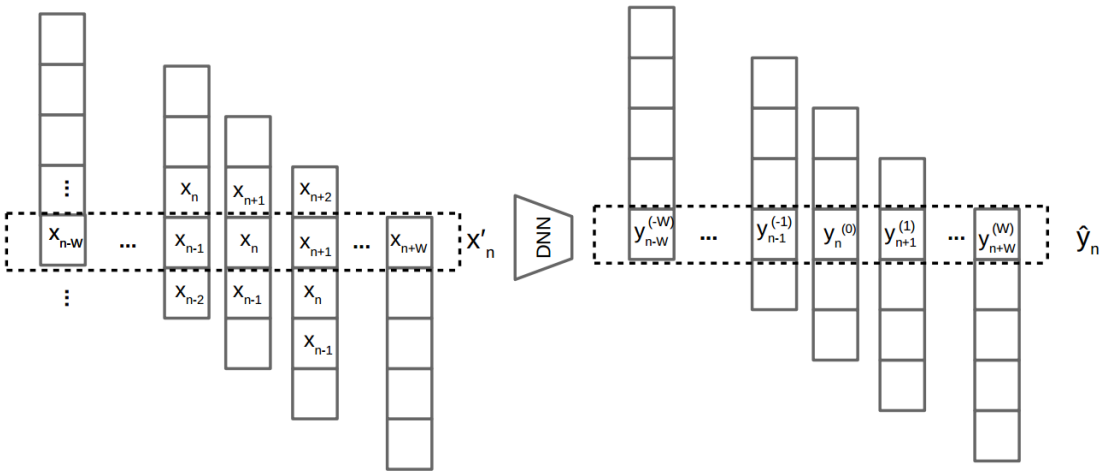

Figure 3.8 Structure of bDNN can be seen clearly

3.2.4 Adaptive Context Attention Model

Adaptive Context Attention Model (ACAM) can be used in speech and image processing effectively [24], [25], [26]. Effective VAD systems can be developed by using ACAM [27]. A recurrent attention-based model can be developed by using context information. ACAM takes multiple input frames that have different frame lengths and decides which one is important for classification.

In the attention mechanism, it is a deep recurrent neural network that processes a multi-resolution features of the input at each step, named a glimpse. The network uses the glimpse information to update the internal state of the input. And outputs are the next glimpse location and possibly the next frame in the sequence [25].

Let's take dataset as $\{(x_m, y_m)\}_{m=1}^{M}$ again. To use context information a wide frame $\{-W, -W+u, -W+2u, ..., W-2u, W-u, W\}$ is created with neighbor frames. MRCGs are used as input features.

$$v_m = [x^T_{m-W}, x^T_{m-W+u}, \quad ..., \quad x^T_{m-1-u}, x^T_{m-1}, x^T_m, x^T_{m+1}, x^T_{m+1+u}, \quad ..., \quad x^T_{m+W}, x^T_{m+W+u}, x^T_{m-W}]^T \tag{3.16}$$

$$y_m = [y^{truth}_{m-W}, y^{truth}_{m-W+u}, \quad ..., \quad y^{truth}_{m-1-u}, y^{truth}_{m-1}, y^{truth}_m, y^{truth}_{m+1}, y^{truth}_{m+1+u}, \quad ..., \quad y^{truth}_{m+W}, y^{truth}_{m+W+u}, y^{truth}_{m-W}]^T \tag{3.17}$$

Model Architecture: ACAM is a recurrent attention model. It takes a different number of frames as input and determines which ones are critical. Flowchart of ACAM is demonstrated in Figure 3.9. It contains decoder, attention, encoder, core and classifier modules [27].

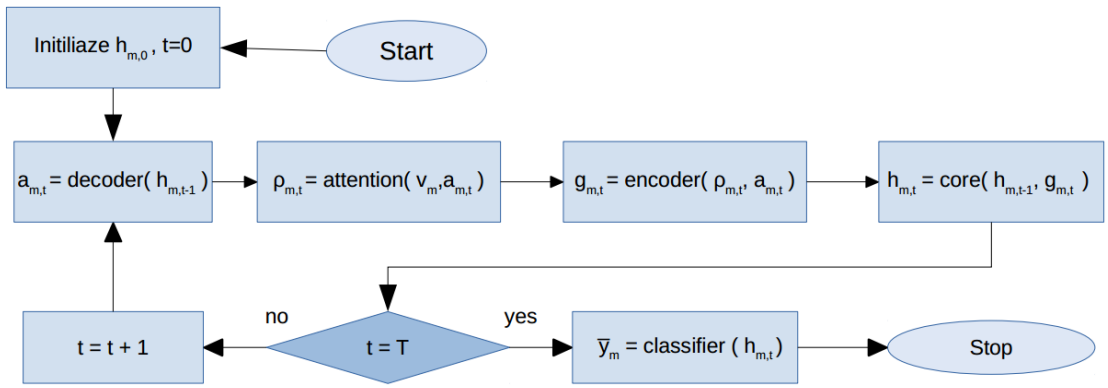

Figure 3.9 Flowchart of ACAM. Usage of modules is shown, respectively. System flows until it reaches T

1) Decoder: Internal state $h_{m,t-1}$ that is coming from core network, goes to the decoder. Then the decoder decides which v_m frames are going to be used by ACAM to generate attention vector $a_{m,t}$. Attend is often implemented by scoring each element in h separately. And normalize these scores.

$$\begin{aligned} \mathbf{a}'_{m,t} &= \text{Relu}\left(\text{Affine}\left(\mathbf{h}_{m,\,t-1}|\theta_{\text{attention}}\right)\right) = \left[a'_{m,t,1}, \ldots, a'_{m,t,k}, \ldots, a'_{m,t,L}\right]^T \\ a_{m,t,k} &= \sigma(a'_{m,t,k}) \Big/ \sum_{k=1}^{L} \sigma(a'_{m,t,k}) \\ \mathbf{a}_{m,t} &= \left[a_{m,t,1}, \ldots, a_{m,t,k}, \ldots, a_{m,t,L}\right]^T \end{aligned} \tag{3.18}$$

On above equations, sample indice is k, $L = 2(W-1)/u + 3$ is context length, affine transformation is $Affine(x|\theta) = w.x + b$, $w, b \in \theta$ are learnable parameters, $Relu(x)$ is the rectified linear unit (ReLU), and sigmoid function is $\sigma(x) = 1/(1 + exp(-x))$. Smoothed softmax function is used in Equation (3.18).

2) Attention: The attended input $\rho_{m,t}$ is obtained with the hadamard product.

$$\boldsymbol{\rho}_{m,t} = [\mathbf{x}_{m-W}^T \cdot a_{m,t,1}, \ldots, \mathbf{x}_{m+W}^T \cdot a_{m,t,L}]^T. \tag{3.19}$$

3) Encoder: The encoder uses $a_{m,t}$ and $\rho_{m,t}$ vectors to obtain aggregated-features $g_{m,t}$.

$$\mathbf{g}_{m,t} = f_g(f_a(\mathbf{a}_{m,t}|\theta_a), f_\rho(\boldsymbol{\rho}_{m,t}|\theta_\rho)|\theta_g) \tag{3.20}$$

in the Equation (3.20), Fully-connected Neural Network (FNN) $f(x|\theta)$ is used with θ parameters. FNNs in the encoder has a hidden layer. In here, $f_a(.)$ is location hidden space (bp hidden layer), $f_\rho(.)$ is glimpse hidden space and $f_g(.)$ has glimpse out hidden layer. They are connected to each other.

4) Core Network: This network dictates what to do at the next step according to the internal state. The internal state encodes the nonspeech environment and the input $g_{m,t}, h_{m,t-1}$ that includes previous action of the modules.

$$\mathbf{h}_{m,t} = \text{LSTM}(\mathbf{g}_{m,t}, \mathbf{h}_{m,t-1}|\theta_{\text{LSTM}}). \tag{3.21}$$

The recurrent network aggregates information extracted from the glimpses. And it combines this information.

LSTM cells are used as recurrent networks because they learn long-range dependencies.

5) Classifier: The decoder works on attended frame $h_{m,t}|_{t=1}^{T-1}$. At the last step T, it is expressed as following:

$$\begin{aligned} \mathbf{y}'_m &= f_c(\mathbf{h}_{m,T}|\theta_c) \\ \mathbf{y}'_m &= [y_{m-W,1}, y_{m-W+u,2}, \ldots, y_{m,k}, \ldots, y_{m+W-u,L-1}, y_{m+W,L}]^T \end{aligned} \tag{3.22}$$

$y_{m,k} \in [0,1]$ is the frame soft prediction within the context. The decision of the frame m is made by the combination of these soft predictions. The decision of frame y_m is made by finding threshold η using the development set. Prediction of targets are made as followings;

$$\hat{y}_m = \frac{1}{L}\sum_{k-1}^{L} y_{m,k}, \quad \bar{y}_m = \begin{cases} 1, & \text{if} \quad \hat{y}_m \geq \eta \\ 0, & \text{otherwise} \end{cases} . \tag{3.23}$$

3.3 Speaker Representation

The speaker identity assignment of the segment is performed after speech segments are generated. Speaker representation is made to get an opinion about the speaker information of the segments. Each various length speech segment is represented by a speaker-specific fixed-size vector. In the scope of this study, for speaker representation i-vector, x-vector, d-vector, and xd-vector are used.

3.3.1 i-vector

The term of i-vector is the short of "identity-vector". It stores information about speaker. Information like channel, microphone, language and acoustic of the room are stored in the vector [28], [29], [30]. The i-vector method is a GMM-based method. There is a speaker adaptation module before i-vector training. To do speaker adaptation Universal Background Model (UBM) is trained without using speaker labels. Then the T matrix, which models the speaker information and non-speaker information, is trained with speaker labels.

3.3.1.1 Gaussian Mixture Model

The gaussian mixture is a model consisting of several gaussians. K is the number of gaussians of the mixture and k is the indice [31]. Each gaussian in the mixture consists of the following parameters:

- Mean μ that indicates the location of centre.
- Covariance Σ defines the width of the gaussian distribution. It is equivalent to the dimensions of the ellipsoid in the mixture model.
- The mixing coefficient w defines the amplitude of the gaussian distribution.

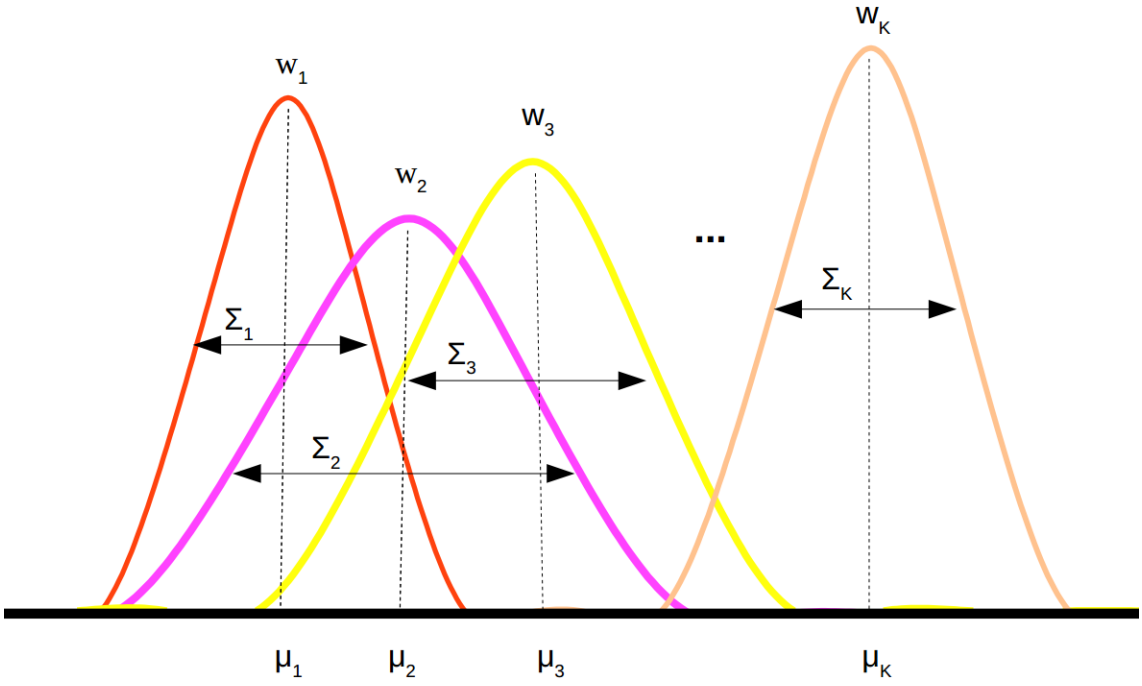

Figure 3.10 An illustration of Gaussian Mixture Model with its parameters

In the Figure 3.10, an example illustration of GMM is given. In here, K can be think as number of component. There are K gaussian distributions and each distribution has own parameters w_k, μ_k and λ_k. The mixing coefficients are probabilities and they provide condition specified in Eq. (3.24).

$$\sum_{k=1}^{K} w_k = 1 \tag{3.24}$$

A gaussian distribution for mixture is given in Eq. (3.25).

$$\mathcal{N}(x|\mu, \Sigma) = \frac{1}{(2\pi)^{K/2}} \frac{1}{|\Sigma|^{1/2}} exp\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\} \tag{3.25}$$

where x is data points, K is the number of component. μ is mean vector with $1xK$ dimension and Σ is covariance matrix with KxK dimension.

The Gaussian Mixture distribution is a summation of gaussian distributions:

$$p(x) = \Sigma_{k=1}^{K} w_k \mathcal{N}(x|\mu_k, \Sigma_k) \tag{3.26}$$

3.3.1.2 Universal Background Model

In the speech processing field use of generative methods based on the gaussian distribution are common. GMM has a important role in speaker-related problems [32]. A research made in this field in 2000 became a milestone especially for speaker recognition problem [33]. A GMM consisted of many gaussian distribution (as mentioned above section) is trained with dataset of many speakers and the name of this GMM is Universal Background Model (UBM).

In Equation (3.27), λ is the model parameters of K-component gaussian distributions for GMM.

$$\lambda = \{w_k,\ \mu_k,\ \Sigma_k\} \qquad k = 1, \quad ..., \quad K \tag{3.27}$$

UBM will be used at later steps to train a good speaker model with maximum a posteriori (MAP) from limited speech data. In Machine Learning, MAP occurs when try to optimize the model using training data as given in Equation (3.28).

$$\gamma_k^{(i)} = Pr(k|x_i, \lambda) \tag{3.28}$$

$\gamma_k^{(i)}$ represents probability of existence the k triphone if it has x_i acoustic features at the i^{th} frame. In Equation (3.28), $\gamma_k^{(i)}$ represents the latent-variables. Small speech unit is called phoneme, and triphone formed by aligning three phoneme and can be called as senone. Triphone is used for examining the relation of speech units with each other.

$$\begin{aligned} w_k &= \textstyle\sum_{i=1}^{N} \gamma_k^{(i)} \\ \mu_k &= \frac{1}{w_k} \textstyle\sum_{i=1}^{N} \gamma_k^{(i)}\ x_i \\ \Sigma_k &= \frac{1}{w_k} \textstyle\sum_{i=1}^{N} \gamma_k^{(i)}\ (x_i - \mu_k)(x_i - \mu_k)^T \end{aligned} \tag{3.29}$$

where w_k is mixture weight, μ_k is mean value and Σ_k is covariance matrix. These are our GMM model parameters which defined in Equation (3.27).

Training Method: The Expectation-Maximization (EM) algorithm [31] is used to train UBM. It is an iterative machine learning algorithm. And the training procedure is given step by step below.

1. Initialize the model parameters (μ_k, Σ_k, w_k) with random values.
2. Compute the $\gamma_k^{(i)}$ latent-variables for all k components.
3. Then estimate model parameters using the updated $\gamma_k^{(i)}$.
4. Calculate log-likelihood function.
5. Set the convergence criterion.
6. If the log-likelihood function converges to criterion, stop, else go back to Step 2.

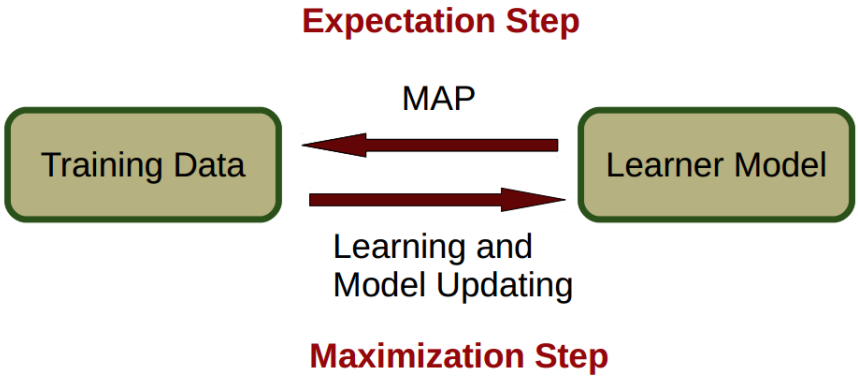

Figure 3.11 Basically, a scheme for training a model with EM

3.3.1.3 i-vector extraction

Speaker identity representing i-vectors are said to be belonging to total variability space (T matrix). This approach is based on defining only one space [28]. In the previous works, speaker and channel spaces are defined as two separate spaces. These spaces are known as the eigenvoice matrix and the eigenchannel matrix [34].

The term of m is UBM supervector and it is independent from speaker and channel, w standard vector with normal distributions $\mathcal{N}(0, I)$, and T represents a low-level rectangular matrix representing speaker-channel variabilities. The term of w represents i-vectors here. A speech segment is represented by speaker-channel dependent supervector (M matrix). And it is assumed to be distributed normally with the mean vector m and the covariance matrix TT^t [35], [36].

$$M = m + T\phi \tag{3.30}$$

Baum-Welch statistics are used for the i-vector extraction. These sufficient statistics are created by using UBM λ. There is a speech records with L frames and corresponding acoustic features are $\{x_1, x_2, ..., x_L\}$.

$$\begin{aligned} N_c = &\quad \sum_{i=1}^{L} \mathrm{P}(k|x_i, \lambda) \\ F_c = &\quad \sum_{i=1}^{L} \mathrm{P}(k|x_i, \lambda) x_i \end{aligned} \tag{3.31}$$

k=1,2, ..., K is the component indices and $\mathrm{P}(k|x_i, \lambda)$ is the posterior of the feature vector x_i is calculated for by the k^{th} component. $\mathrm{P}(k|x_i, \lambda)$ posteriors came from UBM. N_c and F_c are 0^{th} and 1^{th} order Baum-Welch statistics. These are given in Eq. (3.31). Also, for i-vector extraction central 1^{th} order Baum-Welch statistic is needed which is given in Eq. (3.32).

$$\tilde{F}_c = \sum_{i=1}^{L} \mathrm{P}(k|x_i, \lambda)(x_i - \mu_k) \tag{3.32}$$

where μ_k term is the mean of k^{th} gaussian of the UBM (which symbolized with λ). The i-vector calculation of the segment is given at Equation (3.33) [28].

$$w = \left(I + T^t \Sigma^{-1} N(u) T\right)^{-1} . T^t \Sigma^{-1} \tilde{F}(u) \tag{3.33}$$

3.3.1.4 Speaker diarization with using i-vectors

After the speech segment boundaries are determined, segment speaker information is extracted. GMM-based UBM and i-vector models are trained for i-vector based speaker diarization, respectively. Using this i-vector model, fixed-size ivectors are extracted for variable-length segments. Then, PLDA scoring method is used to measure the similarities of the segments in the speech file. PLDA models are crossed and supervised calibration. Finally, by clustering, segments that are similar to each other are labeled as belonging to the same speaker [37]. The scheme of this system is given in Figure 3.12.

3.3.2 x-vector

The x-vectors [38] extracted by using TDNN is implemented to the speaker recognition problem [39], [40]. Better results are obtained compared to i-vectors extracted by using GMM. TDNN based x-vector is trained by only using speaker labels. TDNN model is optimized to make a discrimination between speakers of segments.

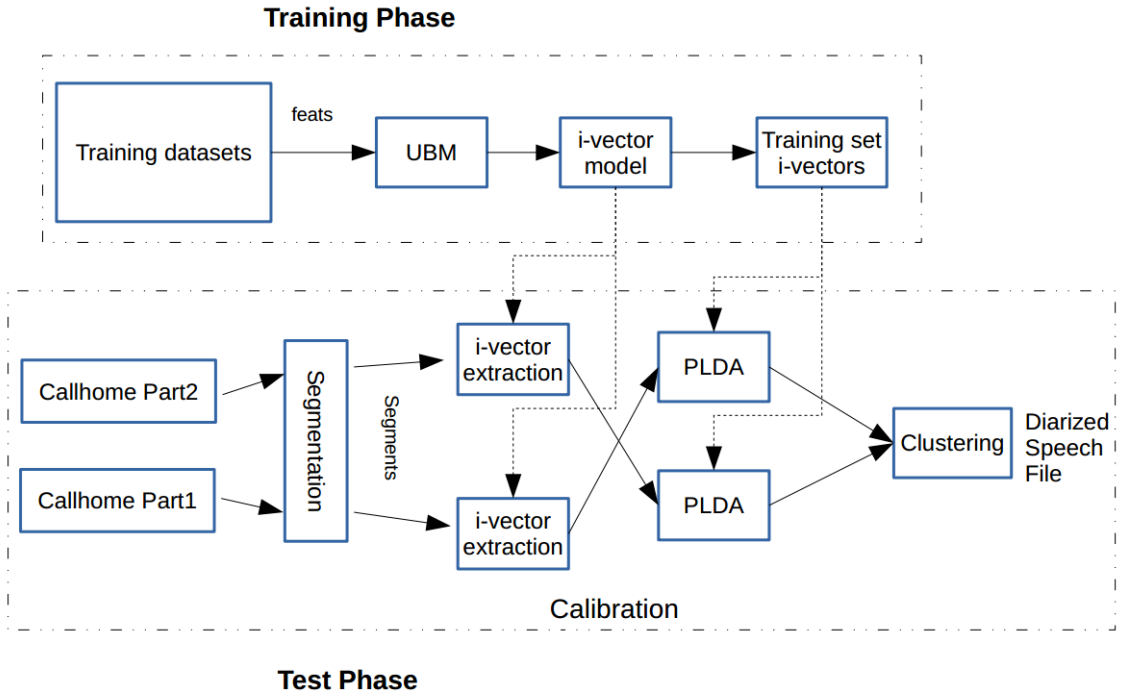

Figure 3.12 General scheme for Speaker Diarization with i-vectors. The extraction of i-vector can be clearly seen here

Speaker embedder based on TDNN is a data-hungry model. This model is augmented with data babble, music, environment noise and reverberation method and a powerful system is obtained.

The network structure of the x-vector is given in Figure 3.13. 40-dimension MFCC, delta and delta-delta features are taken as the network input. On the Table 3.2, T specifies the number of the frames in the one segment, and t $(0 \leq t \leq T)$ is the frame indices. A pooling layer is placed on the network for the segments with variable length. Sub-segmentation with fixed-size is done for each segment and given to model input and an embedding vector is produced. Sub-segment window length is 1.5 seconds and the overlap amount is 0.75 seconds for TDNN based system. All vector's mean, and the standard deviation is taken on the pooling layer. A fixed-size embedding vector is calculated for each segment.

The TDNN configuration is given in Table 3.2. The first layers are frame level layers. They process with t centered with a small temporal context in the current frame. For example,frame3 layer's input features is the combined output of frame2's layer $\{t-3, t, t+3\}$. Local context of frame3 is 7 but it sees a total of 15 frame contexts from bottom. Following frame level layers (frame4-frame5) have same context.

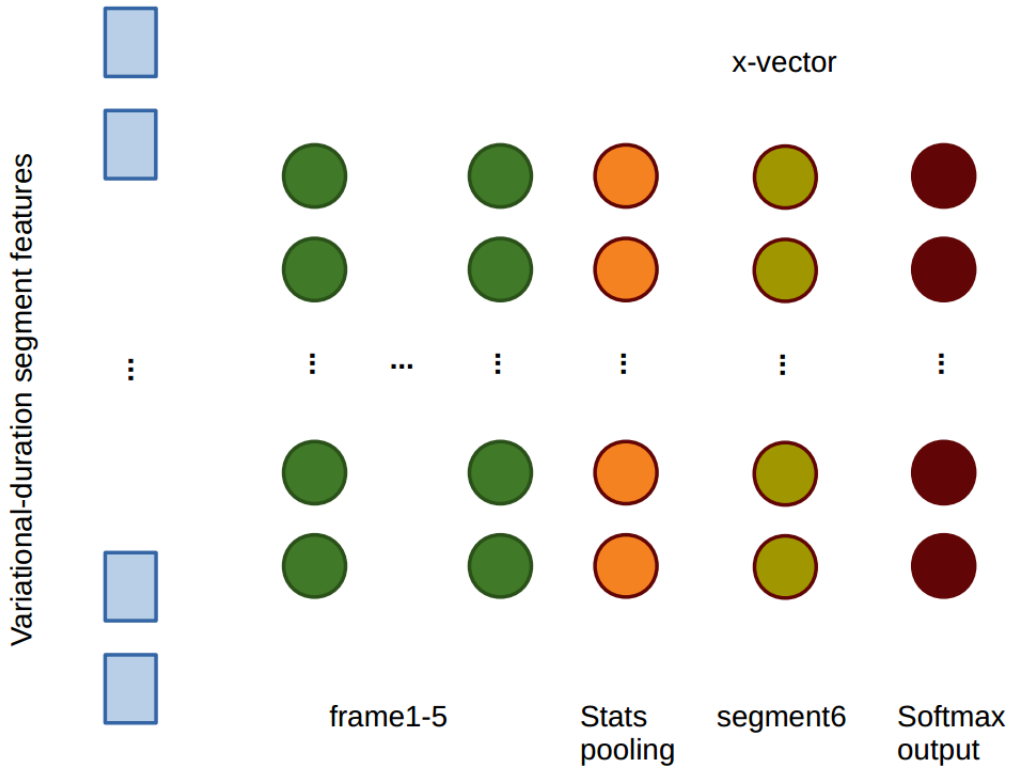

Figure 3.13 Scheme of TDNN-based x-vector model structure. It represents to each segments with fixed-size vector it contains speaker-specific information

The pooling layer combines T frame level outputs from 5^{th} layer, and computes their average and standard deviation. 1500-dimensional vectors are produced for inputs with different lengths from frame level statistics. In the table 3.2 this is indicated by the layer context {} and the total T context. Average and standard deviation values are combined. And it learns the differences between speakers in the segment-level layers, and this is done along the softmax output layer. After training, softmax output layer is not used. The main idea in the embedding of the speaker is to use the hidden layers as a representative of the speaker feature and to distinguish between the speakers.

The N is the speaker number of training dataset and TDNN model is trained to classify training speakers. After training phase, embedding vectors are extracted from the layer segment6 affine component. In here, non-linearity is provided with Rectified Linear Unit (ReLU).

Table 3.2 TDNN embedding layer architecture

Layer	Layer Context	Total context	Input x Output
frame1	$[t-2; t+2]$	5	120x512
frame2	$\{t-2; t; t+2\}$	9	1536x512
frame3	$\{t-3; t; t+3\}$	15	1536x512
frame4	$\{t\}$	15	512x512
frame5	$\{t\}$	15	512x1500
stats pooling	$[0; T)$	T	1500Tx3000
segments6	0	T	3000x128
softmax	0	T	128xN

3.3.2.1 Speaker diarization with using x-vectors

Figure 3.14 shows how x-vectors are used in speaker diarization. TDNN model is trained with speaker labels. The training dataset is reproduced with the augmentation method. Callhome test set is divided into two and utterances are divided into segments. MFCCs are extracted from these segments, and x-vectors have been extracted using these short-term features. During PLDA scoring, supervised calibration has been done. Speaker clusters are created by clustering the generated PLDA scores and diarized speech file is obtained.

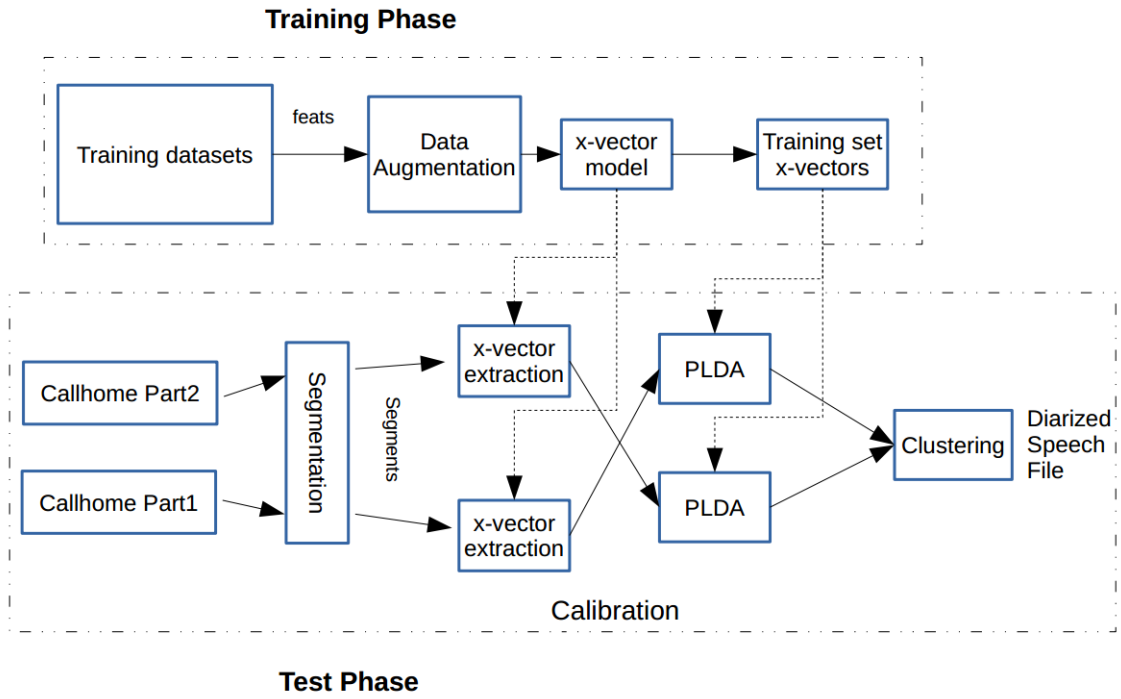

Figure 3.14 Speaker diarization with x-vectors

3.3.2.2 Data Augmentation

Augmentation is made to scale up the variety and size of the training dataset. For this model, augmentation will be performed with reverberation and additive noise. Reverberation includes room impulse response (RIR) noises [41]. For additive noise, the MUSAN dataset is used [42]. We increase our training data three times with data augmentation method. Types of augmentation data used:

- babble: It is selected from MUSAN speech files randomly and added to the original signal (13-20 dB SNR).
- music: It is selected from MUSAN music files and added to the original file (5-15 dB SNR).
- noise: MUSAN noises are added to the original signal at one-second intervals throughout the recording (0-15 dB SNR).
- reverb: Speech file reverberates artificially through convolution with RIRs.

3.3.3 d-vector

The d-vectors based on LSTM [43] are used for speaker recognition systems [44]. The embedder model trained with fixed-size segments extracted from big speech datasets. The Figure 3.15 shows the d-vector extraction steps from a speech file.

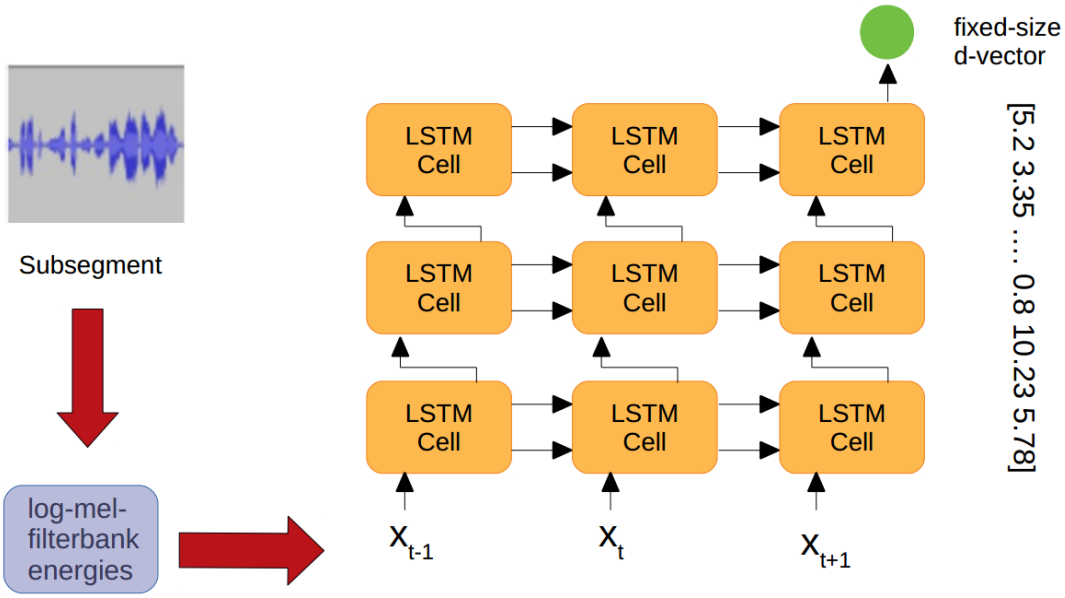

Figure 3.15 Flowchart of d-vector extraction

Speech segment boundaries are determined by using VAD, and then these segments (maximum 240 ms) are divided into sub-segments. The overlap window width of the sub-segments is 120 ms. Corresponding d-vectors for each segment are applied L2 normalization first, then averaged to create the embedding of the segment. This process shows different duration speech segments as a fixed-size embedding. The diarization system scheme is given in Figure 3.18.

There is three LSTM layer and a linear layer in speaker identification network to calculate the embeddings. Generalized end-to-end loss (GE2E) is used to train LSTM network [44].

3.3.3.1 LSTM

Long-Short Term Memory (LSTM) network has a recurrent structure. It is a special version of Recurrent Neural Networks (RNN). It has capable of learning long-term dependencies between time steps of sequence data. These networks were introduced in 1997 [43]. It has been applied to various and numerous problems and solutions have been proposed for these problems (it could be a daily-life problem, an academic problem, or any other kind of problem).

LSTM is designed to deal with the long-term dependency problem. The network may or may not hold the memory, the decision is made depending on the data. LSTM can determine when to remember and forget information and how to make connections between old memory with the new input. It is a characteristic behavior for LSTMs. Holding the long term dependencies in the network is done by its gating system. There is three gate; forget gate, input gate and output gate.

RNNs have a recurrent structure. It has a chain structure. This repeating module has only one $tanh$ layer. LSTMs also have a chain structure, but there are differences in the repeating module. The LSTM unit cell has four neural network layers that interact in a special way. This interaction is given in the Figure 3.16.

In Figure 3.15, interaction of unit cells can be seen with each other. Cell state and hidden state are transferred between cells. These three gates that process the data. These are forget gate, input gate, output gate. The forget gate is responsible to get rid of unnecessary information. The input gate adds information to the cell state. And the output gate select useful information and gives as output.

In the Figure 3.16, each line carries one feature vector, from the output of one node to the input of the other nodes. There is point-wise operations (such as "+" and "x"), like vector addition and multiplication. Neural network layers learn problem solution.

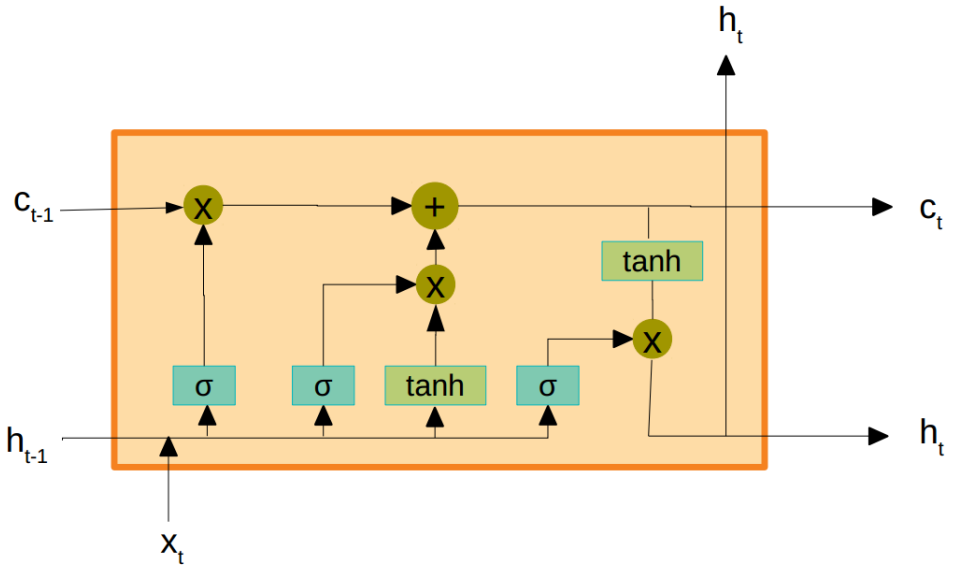

Figure 3.16 Scheme of LSTM unit cell and interaction between its layers. This is basic definition of LSTM cell

3.3.3.2 Generalized End-to-End loss method

Generalized end-to-end (GE2E) training should do with large number of utterances. There are N speakers and M speech files in a batch. [44], as is shown in Figure 3.17.

NxM utterances are fetched to make one batch. Each feature vector represented with x_{ji} $(1 \leq j \leq N$ and $1 \leq i \leq M)$ where j is speaker indices and i is utterance indices. Output of the network is $f(x_{ji}; \lambda)$. The term λ represents all the parameters of the whole embedder network. The d-vector is given in the Equation (3.34) and the L2 normalization is applied to the output of the network. This is the final process in order to obtain d-vector embeddings.

$$e_{ji} = \frac{f(x_{ji}; \lambda)}{||f(x_{ji}; \lambda)||_2} \tag{3.34}$$

For each input feature, d-vector eambedding e_{ji} is extracted from LSTM network. It belongs to i^{th} input of j^{th} speaker. The centroid c_j of one speaker d-vectors $(e_{j1}; ...; e_{jM})$ extracted from M utterances. These centroids will use to build a similarity matrix. They represents as followings;

$$c_k = \frac{1}{M}\sum_{m=1}^{M} e_{km} \tag{3.35}$$

The similarity matrix $S_{ji,k}$ is calculated with cosine similarities between each d-vector e_{ji} to all centroids c_k (where $1 \leq j,k \leq N$ and $1 \leq i \leq M$), as given in Equation (3.36). GE2E builds a similarity matrix as depicted in Figure 3.17;

$$S_{ji,k} = w.cos(e_{ji}, c_k) + b \tag{3.36}$$

where weight (w) and bias (b) are learnable parameters. There is a condition $w > 0$, because similarity between e_{ji} and c_k should be larger when cosine similarity is larger [44]. Similar data points have larger scores.

In Equation (3.37), contrast loss is defined in positive pairs and the most aggressive negative pairs.

$$L(e_{ji}) \;=\; 1 - \sigma(S_{ji,j}) + \max_{\substack{1<k<N \\ k\neq j}} \sigma(S_{ji,k}) \tag{3.37}$$

The Softmax loss function brings the embedding vector closer to its centroid, and away from all other centroids. Its Equation is given as followings;

$$L(e_{ji}) \;=\; S_{ji,j} + log\sum_{k=1}^{N} exp(S_{ji,k}) \tag{3.38}$$

Losses are calculated based on positive and negative hypotheses. The positive component is that the hypothesis claimed in the segment is compatible with the d-vector. The negative component is the incompatibility of these two vectors. When calculates similarity matrix, to find negative similarity we use Eq. 3.39.

$$c_j^{(-i)} = \frac{1}{M-1}\sum_{\substack{m=1 \\ m\neq i}}^{M} e_{jm} \tag{3.39}$$

$$S_{ji,k} = \begin{cases} w.cos(e_{ji}, c_j^{(-i)}) + b & \text{if} \quad \text{k = j} \\ w.cos(e_{ji}, c_k) + b & \text{otherwise} \end{cases} . \tag{3.40}$$

GE2E loss observed with combination of Equations (3.34), (3.37) and (3.40). It is the sum of all losses over the similarity matrix. Loss function is given as followings;

$$L_G(x;w) = L_G(S) = \sum_{j,i} L(e_{ji}). \tag{3.41}$$

In the Figure 3.17, each speaker is filled in different colors. Alternative hypotheses are gray. Example building in the this illustration, three utterances are used for each speaker.

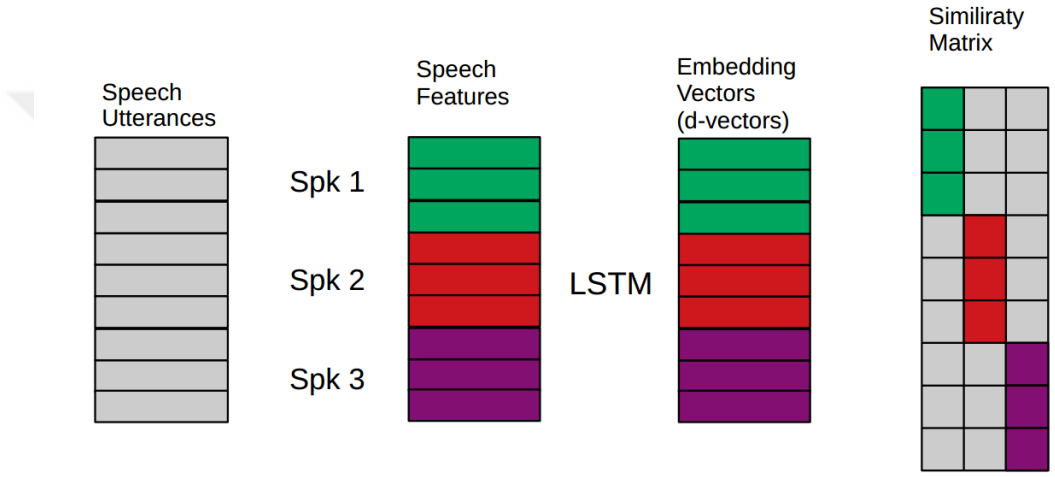

Figure 3.17 Same colours are belong to same speaker. Grey area means different speaker. Generalized end-to-end loss is calculated with using similarity matrix

3.3.3.3 Speaker diarization based on d-vector

Mel filterbank energies are used to d-vector extraction. LSTM layers take mels as input features. The linear layer following the LSTMs produce the d-vector. Again, the score calibration is used during PLDA modeling. And PLDA scores are observed. Then diarized speech files are obtained after clustering phase [45], [46]. One speech file diarization is illustrated in Figure 3.18

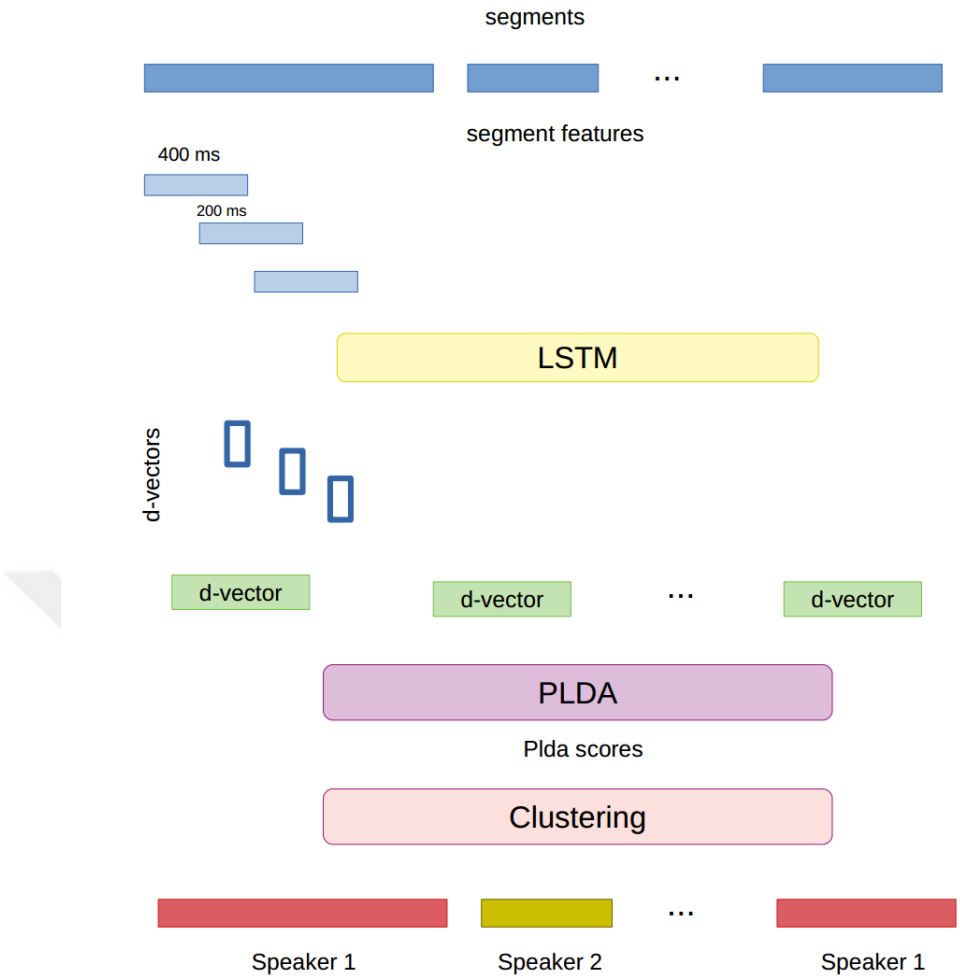

Figure 3.18 Flowchart of LSTM model used for speaker diarization. Extraction of the d-vectors used for speaker representation and their use in the system are shown

3.4 Clustering

Clustering is basically a machine learning technique that groups similar data points under the same cluster [12]. The group of similar data points is named as cluster. Clusters have not any label.

Each segment in the speech file is represented by a fixed-size embedding vector. With the clustering process, it is aimed to assign the parts of the same speaker as a speaker identity. Each segment in a speech file is represented by a vector and similarity scores are obtained between the segments. The segments of the same speaker are assigned as a speaker ID during the clustering process.

3.4.1 Probabilistic Linear Discriminant Analysis

Fixed-size vectors used in speaker representation include information such as channel, session, microphone as well as speaker information. Information that is independent of the speaker is called in-class variablity. The probabilistic linear discriminant analysis (PLDA) method separates in-class variablity and between-class variablity. PLDA method is used to generate similarity scores between segment representation vectors [47], [48].

In the training data, each individual is represented by the L and the individual's speech is represented by the K. Hence, l individual's k speech is x_{kl}. Data generation is modeled as Equation (3.42).

$$\phi_{kl} = \mu + Fh_k + Gw_{kl} + \epsilon_{kl} \tag{3.42}$$

PLDA model has two components. These are the signal and noise component ($\mu + Fh_k$, $Gw_{kl} + \epsilon_{kl}$ respectively). The signal component represents between-speaker changes, and the noise component represents in-speaker changes. The term μ refers to the overall mean value of the training data set. The columns of the matrix F contain the information of the between speakers subspace, and the vector h_k contains the position information in the subspace. The matrix G represents the in-class variabilities subspace, the vector w_{kl} represents the position information of this subspace. Unexplained data changes are defined as diagonal covariance Σ, with the help of ϵ_{kl}, where is $\epsilon_{kl} \sim \mathcal{N}(0, \Sigma)$. An EM algorithm can be used to learn the parameters of this model (μ, F, G and Σ) [49].

In Equation (3.42), h_k term is important because it represents the identity of individual k. It is named as *latent identity variable*. To measure the similarity, the likelihood of the underlying h_i is calculated.

Similarity score is extracted with the two covariance scoring method using the PLDA model. This method calculates the logarithmic-likelihood ratio between the two speaker representation vectors [50].

PLDA takes the fixed-size vector ϕ_{ij} as input and assumes it has a gaussian distribution as Eq. 3.43;

$$P(\phi_{ij}) = \mathcal{N}(\phi_{ij}|\mu, FF^T + GG^T + \Sigma) \tag{3.43}$$

where μ is mean and $FF^T + GG^T + \Sigma$ term corresponds to covariance matrix, ϕ_{ij} is vector that belongs to j^{th} segment of i^{th} speaker.

In the Equation (3.44), the embedding vectors between ϕ_1 and ϕ_2 are desired to be calculated. H_s is the hypothesis that the person speaking in two segments is the same, H_d is the hypothesis that the person speaking in two segments is different. In here, λ is the logarithmic-likelihood ratio between the two segments.

$$\lambda = \log \frac{P(\phi_1, \phi_2|H_s)}{P(\phi_1, \phi_2|H_d)} = \log P(\phi_1, \phi_2|H_s) - \log P(\phi_1, \phi_2|H_d) \tag{3.44}$$

PLDA is latent variable modeling approach, and H_s, H_d are illustrated in Figure 3.19. In the PLDA model, If $\{\phi_1, \phi_2\}$ vectors belong to the same speaker, they share the same speaker-specific latent variable $h_{1,2}$. H_s refers to same speaker hypothesis. On the other hand, $\{\phi_1, \phi_2\}$ vectors belong to different speakers and hence have separate latent variables, h_1 and h_2, in the model. H_d is different speaker hypothesis. The similarity score is calculated as the log-likelihood ratio between two models.

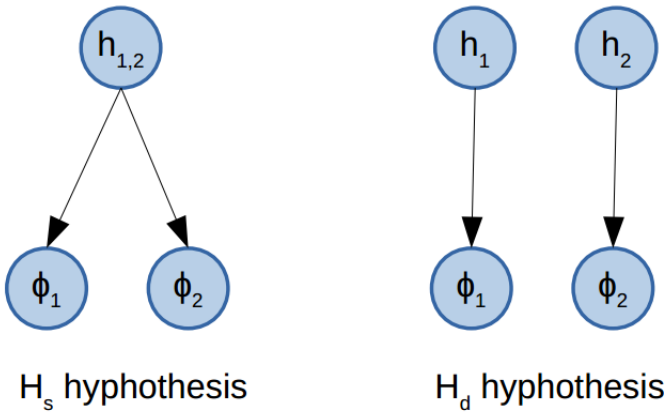

Figure 3.19 Hypothesis to generate PLDA scoring between two speech segment vectors

The generative equation for the H_s is given by Eq. (3.45). It is expansion of 3.42.

$$\underbrace{\begin{bmatrix}\phi_1\\ \phi_2\end{bmatrix}}_{\tilde{\phi}} = \underbrace{\begin{bmatrix}\mu\\ \mu\end{bmatrix}}_{\mu} + \underbrace{\begin{bmatrix}F & G & 0\\ F & 0 & G\end{bmatrix}}_{\tilde{A}} \begin{bmatrix}h_{1,2}\\ w_1\\ w_2\end{bmatrix} + \begin{bmatrix}\epsilon_1\\ \epsilon_2\end{bmatrix} \tag{3.45}$$

Probability of ϕ_1 and ϕ_2 belong same speaker are derived from 3.42 and 3.43.

$$\log\ P(\phi_1, \phi_2|H_s) = log\mathcal{N}(\tilde{\phi}|\mu, \tilde{A}\tilde{A}^T + \Sigma) \tag{3.46}$$

The generative equation for the H_d is given by Eq. (3.47).

$$\begin{bmatrix}\phi_1\\ \phi_2\end{bmatrix}=\begin{bmatrix}\mu\\ \mu\end{bmatrix}+\begin{bmatrix}F & G & 0 & 0\\ 0 & 0 & F & G\end{bmatrix}\begin{bmatrix}h_1\\ w_1\\ h_2\\ w_2\end{bmatrix}+\begin{bmatrix}\epsilon_1\\ \epsilon_2\end{bmatrix} \tag{3.47}$$

Separate operations can be made for the first and second rows in the Equation 3.47. The log-likelihood score of H_d is given as sum of log-likelihoods. This equation is given in (3.48). Because of logarithmic scale, operations are summation process [51].

$$\log\ P(\phi_1,\phi_2|H_d)=\sum_{l=1}^{2} log\mathcal{N}(\phi_l|\mu,FF^T+GG^T+\Sigma) \tag{3.48}$$

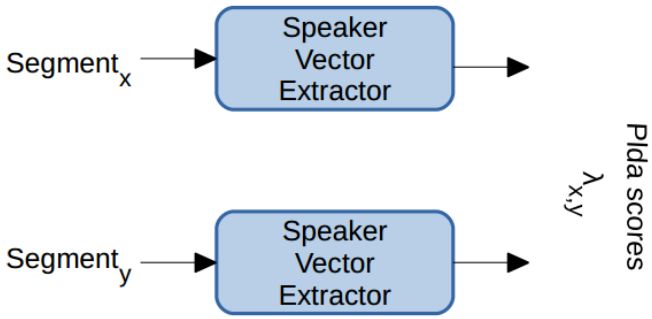

Figure 3.20 Observed PLDA scoring between two segment vectors. PLDA model is used to get log-likelihood ratio

3.4.2 Calibration

In the calibration process [52], the test set is divided into two equal parts and the representation vectors are extracted. This process is a supervised approach where only labeled in-domain scores are required. It needs access to the evaluation data to learn the parameters.

After speaker representation vectors are extracted, two different PLDA models are trained for two different parts of the test set. Training set vectors are also used to data adaptation. The PLDA model of the cross-set is used to generate similarity scores [37].

Another important point for calibration, there must be more than 100 speakers in the test set to adapt to data.

In Figure 3.12 and Figure 3.14, how the calibration process is applied to the diarization problem is shown. Calibration process is implemented all speaker representation methods.

3.4.3 Agglomerative Hierarchical Clustering

Similarity scores are generated between the representation vectors of the segments by PLDA method. In this study, Agglomerative Hierarchical Clustering (AHC) method is used in clustering similarity scores to determine the parts of the same speakers. In AHC, each of the data samples is considered a cluster first, and then clusters that are close together are combined. If the number of clusters is specified in the AHC, clustering continues until the number of clusters reaches or if a threshold value has been given, until it reaches the threshold value between the clusters [53].

1. initialize each segment of speech as a cluster;
2. calculate the similarity between each cluster;
3. assign clusters closest to each other as a cluster;
4. update the distances of the remaining clusters to the specified clusters;
5. iterate steps 2)–4) until stopping criterion is met.

The general scheme of AHC is given in Figure 3.21. In this diagram, the distance expression and data points are expressed symbolically. The stopping criteria has been set as "6". Accordingly, data points are divided into 3 clusters. These clusters are; {"a"to"g"}, {h} and {"i"to"p"}. If the cluster number is given instead of the stopping criteria, for example "2". In this case, clusters are; {"a"to"h"} and {"i"to"p"}.

3.4.3.1 Estimation of the Number of Speakers

In the method used in this study, a threshold was used to estimate the number of clusters. When the distance between the clusters falls below the threshold, the merging process is stopped.

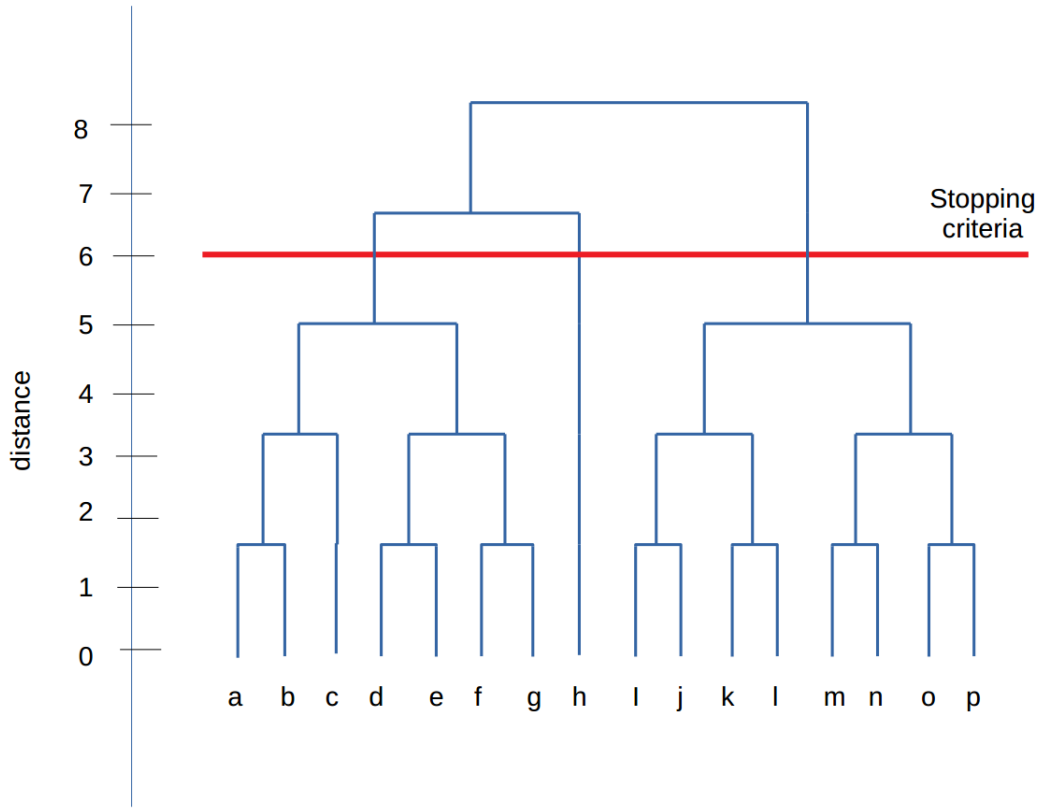

Figure 3.21 Agglomerative hierarchical clustering merges closest samples. Algorithm clusters until it is reached specified threshold or cluster count

A supervised calibration approach is used where labeled in-domain scores are required to determine the threshold. While deciding the threshold, an optimum value is determined on the development set by testing for different threshold values. This approach uses a generative score model and labeled in-domain scores require a 2-component GMM.

3.4.4 Variational Bayes Resegmentation

Diarization algorithms use speech segment boundaries from the VAD output. However, information about the speaker is not used while determining these limits. Outputs determined by VAD may include overlapped speech parts, and this results in segments with multi-speakers containing partial overlap. Variational bayes (VB) is used to improve diarization outputs. It is refined by matching a model of the speaker's features with temporal constraints on speaker transitions.

I-vector model and diagonal UBM are trained for VB. These models are mentioned in the previous sections. Frame log-likelihoods are calculated for the Hidden Markov Model (HMM). Then, HMM calculates speaker posterior probabilities [54], [55], [56].

This loop is shown in Figure 3.22.

VB contains three hidden random variables [57]. These roles are as below.

1. Assign speaker ID to segments.
2. i-vector extraction.
3. The assignment of frames to speaker i-vector.

Frame level statistics cannot be optimized directly. For this reason, VB is used. Temporal continuity is not considered in the original system. HMM takes frame loglikelihoods as input and tries to guess which speaker the frames belong to.

VB is a post-processing process. After the speech file is diarized, refinement is done for the segment IDs. The i-vector model is trained to calculate the speaker posterior. Segments are arranged more likely with speaker posteriors.

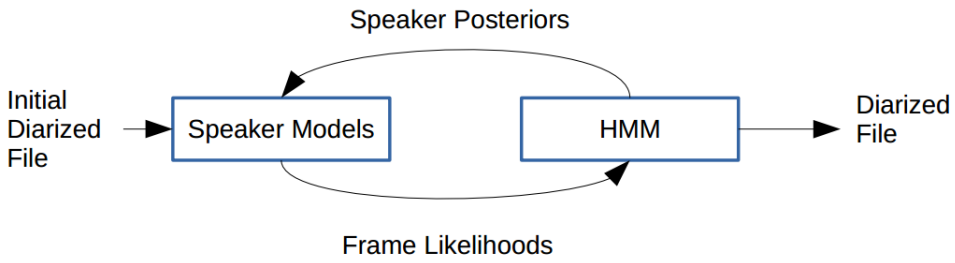

Figure 3.22 Variational bayes resegmentation structure

4
MODELLING AND EXPERIMENT DETAILS

4.1 Test Scenarios

The number of speakers is important in the speaker diarization problem. Two different scenarios can be considered depending on the state of this prior knowledge (If it exists or is not exist). If data such as telephone conversations, interviews are examined, the number of speakers can be given as pre-info. However, it may not be possible to give information about the number of speakers in the analysis of a large amount of TV recordings. In this study, tests will be done for two different scenarios and its effect on success will be examined. Another important point is that speech recordings should not contain overlap segments. Segments containing overlap are not included in the evaluation.

4.2 Performance Metrics

It is the metric diarization error rate (DER)[1] used as an evaluation criterion in speaker diarization systems. DER consists of the sum of three errors: Missed Segment (MS), False Alarm (FA), and Speaker Error (SE). MS is that the system cannot find the reference criteria and label it as silence. FA is the system tagging non-reference segments as if there are speech. SE is when a speaker identity is assigned to the wrong speaker. The most important of these three errors is SE. And in some studies, only SE can be used during evaluation[58].

Detection error rate can be used to examine the performance of the VAD module. It is the sum of FA and MS errors that relate to the metric speech segmentation module. A detailed explanation is given in the following sections. SE is affected by all speaker diarization modules. In order to analyze the performance of Speaker embedder, experiments in which speech boundaries information is used from the reference are also conducted.

[1]The perl script of performance criteria can be found in the following address for evaluation: www.itl.nist.gov/iad/mig/tests/rt/2006-spring/code/md-eval-v21.pl

4.2.1 Detection

VAD can be evaluated with detection error rate metric. In Equation 4.1, formulation of this metric is given.

$$detection\ error\ rate = \frac{FA\ +\ MS}{total\ reference\ speech\ time} \tag{4.1}$$

It is a rate that true speech/nonspeech detection error rate throughout the entire recording.

4.2.2 Diarization

DER is evaluated speaker diarization system as a whole. Its formulation is given in the Figure (4.2)

$$diarization\ error\ rate = \frac{FA\ +\ MS\ +\ SE}{total\ reference\ speech\ time} \tag{4.2}$$

It is a rate that false speech/nonspeech detection rate plus false speaker identity assigning throughout the entire recording.

Evaluation of system can be seen obviously in Figure 4.1. System compares hypothesis with its reference.

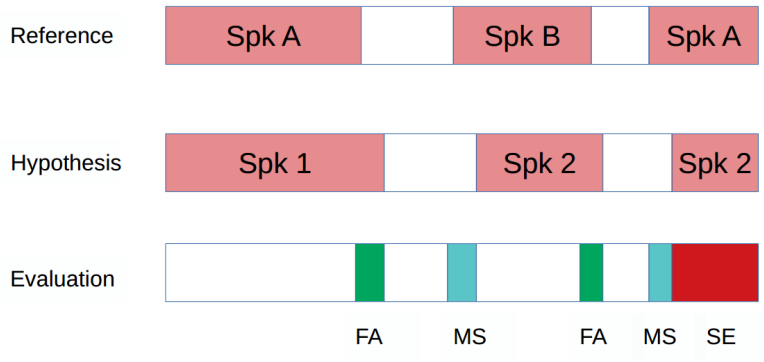

Figure 4.1 Diarization Error Rate is the sum of three error rates

4.3 Output File Format: rttm

Rich transcription time marked (RTTM)[2] files are generated as output from speaker diarizaton system. It contains information (such as speech boundaries, speaker ID) about records. One line represents one segment and each line has ten columns. These columns meanings are explained:

1. Type: It is always SPEAKER.
2. File ID: It is utterance name.
3. Channel ID: It takes same value for same utterance. Problem usually examines mono records.
4. Turn Onset: It is beginning point of the segment in seconds.
5. Turn Duration: Duration of segment in seconds.
6. Orthography Field: is always <NA>
7. Speaker Type: It is always <NA>
8. Speaker Name: Name of speaker of segment. It should be unique within scope of each utterance.
9. Confidence Score: System probability that information is correct; It is always <NA>
10. Signal Lookahead Time: It is <NA>

For example:

SPEAKER gabzh 1 0.17 0.42 <NA> <NA> SPEAKER0 <NA> <NA>
SPEAKER gabzh 1 1.42 1.29 <NA> <NA> SPEAKER1 <NA> <NA>
SPEAKER gabzh 1 2.73 0.43 <NA> <NA> SPEAKER1 <NA> <NA>
SPEAKER gabzh 1 3.78 1.31 <NA> <NA> SPEAKER0 <NA> <NA>

In here: "gabzh" is utterance name, "1" is channel information, "0.17 0.42" is start-duration time respectively and "SPEAKER0" is speaker ID. This format is most usual output type for diarization.

[2]Explanation of file format in detail: https://catalog.ldc.upenn.edu/docs/LDC2004T12/RTTM-format-v13.pdf

4.4 Implementation Details

Four different methods are used for VAD; TDNN, DNN bDNN and ACAM. Kaldi's aspire recipe is used for TDNN-VAD. MRCG feaures for DNN, bDNN and ACAM based VAD are extracted with MATLAB codes. The tensorflow framework is used for modeling operations.

Kaldi callhome_diarization/v1 recipe as used for training of i-vector model. For speaker embedding vectors, x-vectors are trained with the help of the 'Kaldi' library [59]. PyTorch library is used for LSTM based d-vectors.

4.5 System Architecture

The architecture of the models used for VAD will be given in this section. DNN hidden layer dimensions are {512,512}. Input features are 768-dimensional MRCG. Context information values are (W=19, u=9). These values are same for bDNN and ACAM. Hidden layer dimensions are {512,512}. ACAM architecture: glimpse hidden = 128, bp hidden = 128, glimpse out = 128, nGlimpse (T) = 7, lstm cell size = 128, actionHidden1 = 256, actionHidden2 = 256. MFCCs are extracted every 1Oms with 25ms of frame length for TDNN-VAD. TDNN context structure is {-2,-1,0,1,2}, {-1,0,1,2}, {-3,0,3,6}, {-6,0,6,12, 3^{rd} layerStats} and {-12,0,12,24, 4^{th} layerStats}, respectively.

For the x-vector based TDNN used, the MFCC features frame length is 25 ms and the shift length is 10 ms. The TDNN context structure is in layers, respectively: {t—2, t—1, t, t+1, t+2}, {t—2, t, t+2}, {t—3, t, t+3}. Table 3.2 gives x-vector TDNN architecture in detailed. The network structure receives 7 past frames and 7 future frames as input. The TDNN network produces a fixed-size speaker embedding vector for each segment.

The LSTM model consists of 3 layers and 768 hidden nodes. LSTM takes 40 dimensional mel filter energies as input. The output layer is the size of the d-vector, and this vector is called the speaker embedding vector.

During the test, calibration is done to process large data and PLDA model is trained from the other part of the test data for the test and data adaptation is made. PLDA model of the cross data set is used for the similarity score. Calibration has been used in i-vector, x-vector, d-vector and xd-vector based systems. This should be more than 100 speakers [37].

4.6 Datasets

NIST SRE dataset (SRE 04, 05, 06, 08, 10) is used to train i-vector, x-vector and d-vector models used for speaker representation. This dataset consists of 16555 speakers and 49362 utterances. In the i-vector method, a subset of 32000 files of the SRE set is used when training UBM. Representation vectors extracted from this dataset are used when training the PLDA model. For the PLDA model, the i-vectors of the test set are used crosswise and the scores are calibrated.

sre08 https://catalog.ldc.upenn.edu/LDC2011S08
sre06 https://catalog.ldc.upenn.edu/LDC2011S10
sre05 https://catalog.ldc.upenn.edu/LDC2011S04
sre04 https://catalog.ldc.upenn.edu/LDC2006S44

MUSAN [42] and RIR [41] noise is used in the data augmentation process during x-vector training. In the previous section, we mentioned about data augmentation.

Training data TIMIT [60] dataset is used for DNN, bDNN, ACAM based methods used for VAD system. Pre-trained models are used for these systems. 1800 hours of English fisher dataset [61] is used while training the acoustic model for TDNN-VAD. Aspire data is used while training the phone recognizer based TDNN-VAD model.

Callhome disk-8 dataset is used as test data. It consists of 500 audio files. It contains speeches in several languages: Arabic, English, German, Japanese, Mandarin and Spanish. The number of speakers in an audio file ranges from 2 to 7 people. There are overlapping segments in the records, these parts are ignored during evaluation. These recordings are single channel (mono) and 8 kHz.

4.7 Test Results

The bar graph examining the VAD performance is given in Figure 4.2. Performance comparison is made for four different VAD methods used in the study. Although TDNN based VAD is independent from any threshold, it performed better than other VADs.

Speaker diarizaton system is tested for two different scenarios. While analyzing conversational telephone speeches, meeting speeches, the number of speakers can be given to the system and better results can be obtained. Each utterance may contain the different number of speakers in the Callhome dataset.

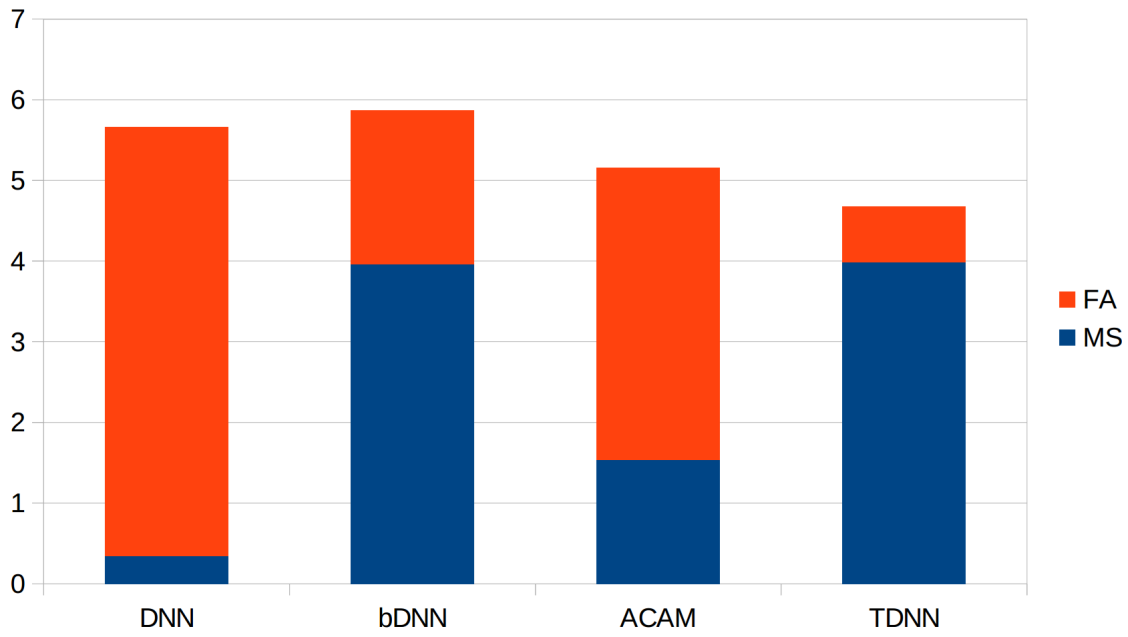

Figure 4.2 Graph of detection error rate. It is sum of FA and MS. VAD systems is tested with Callhome data

Firstly, we are going to compare i-vector and x-vector which are implemented with the Kaldi toolkit. And the effect of the segmentation module also will be examined. Detection error rate is given as a bar graph in Figure 4.2. In the following tables (4.1, 4.2, 4.3 and 4.4), effect of VADs to speaker representation is observed with i-vector and x-vector methods.

The first scenario is a known number of speaker situation. Table 4.1 is a performance report for i-vector. There are 4 different DER results for i-vector diarization which used different VAD modules.

Table 4.1 DER evaluation of speaker diarization systems based on i-vector where the number of speakers is given as pre-info

i-vector based diarization with num spk				
VAD for segmentation	Error types			
	MS	FA	SE	DER
DNN	0.34	5.32	8.73	14.39
bDNN	3.96	1.91	8.05	13.92
ACAM	1.54	3.62	8.16	13.32
TDNN	3.99	0.69	6.73	11.41

In Table 4.2, i-vector method replaced with x-vector. Then the effect of VAD modules on the speaker representation is investigated using the x-vector.

Table 4.2 DER evaluation of speaker diarization systems based on x-vector where the number of speakers is given as pre-info

x-vector based diarization with num spk				
VAD for segmentation	Error types			
	MS	FA	SE	DER
DNN	0.34	5.32	6.89	12.55
bDNN	3.96	1.91	5.75	11.62
ACAM	1.54	3.62	6.66	11.82
TDNN	3.99	0.69	5.64	10.32

When we compare Table 4.1-4.3 and Table 4.2-4.4 line by line, it is observed that the number of speakers given a relatively %10-20 gain.

Another scenario is unknown number of speaker. Two following table analyzed this scenario for i-vector and x-vector.

Table 4.3 DER evaluation of speaker diarization systems based on i-vector where the number of speakers is not given as pre-info

i-vector based diarization without num spk				
VAD for segmentation	Error types			
	MS	FA	SE	DER
DNN	0.34	5.32	10.24	15.9
bDNN	3.96	1.91	9.62	15.49
ACAM	1.54	3.62	10.84	16.0
TDNN	3.99	0.69	8.55	13.23

Again, you can see the contribution of the number of speaker info to 4.3 and 4.4. Also, you can examine the effect of VAD modules on SE value.

Table 4.4 DER evaluation of speaker diarization systems based on x-vector where the number of speakers is not given as pre-info

x-vector based diarization without num spk				
VAD for segmentation	Error types			
	MS	FA	SE	DER
DNN	0.34	5.32	9.26	14.92
bDNN	3.96	1.91	8.38	14.25
ACAM	1.54	3.62	8.62	13.78
TDNN	3.99	0.69	7.13	11.81

In Table 4.1, 4.2, 4.3 and 4.4, we can see that x-vector's DER lower than i-vector's DER under different VAD condition. When we look at the SE, we can compare the two different methods used during the speaker identification process, i-vector and x-vector speaker representation algorithms.

Table 4.5 Speaker diarization studies with using all modules which tested with Callhome dataset

Comparison Table				
Algorithm	Error types			
	MS	FA	SE	DER
Wang et. al. [45]	4.6	2.2	12	18.8
TDNN/x-vector	3.99	0.69	**7.13**	**11.81**

In Table 4.5, we can also see that TDNN-VAD performance is better than it used. They have GMM-VAD which used Perceptual Linear Prediction (PLP) features as input.

In the speaker diarization problem, SE is seen as the most important of DER error types. In Table 4.6 and 4.7, experiment results are given in which speech boundaries are taken from the reference VAD marks. Kaldi called it Oracle segmentation. There are overlapping speech segments on the Callhome test set. Overlapping speech segments are ignored during the evaluation because overlap speech detection is a different research topic. Since reference time information is used, speech segmentation errors are zero. In other studies conducted in the literature, it may be seen that a scoring can be done without including FA and MS error rates because SE has associated with speaker identification algorithms.

In these experiments, deep learning based methods have been tried for identification. These methods are commonly called speaker embedding. Methods used: d-vector, x-vector and xd-vector. In the Table 4.6, the test results for the situation where the number of speakers is known are given.

Table 4.6 Performance of speaker diarization system if the number of speaker is provided as pre-info. SE values of speaker embedder are examined

Speaker Embedder results with num spk				
Speaker Embedder	Error types			
	MS	FA	SE	DER
d-vector	-	-	12.67	12.67
x-vector	-	-	6.28	6.28
xd-vector	-	-	6.09	**6.09**

Test results are given for the scenario where the number of speakers is unknown in table 4.7. The effect of the number of speakers information can be seen between Table 4.6 and Table 4.7. In this experiment, providing this info reduces the error rate relatively by about %20.

Two different embedding vectors extracted are combined to create xd-vectors. Diarization performance using these vectors is analyzed in Table 4.6 and Table 4.7.

Table 4.7 Performance of speaker diarization system if the number of speaker is not provided as pre-info. SE values of speaker embedder are examined

Speaker Embedder results without num spk				
Speaker Embedder	Error types			
	MS	FA	SE	DER
d-vector	-	-	13.8	13.8
x-vector (+VB)	-	-	8.14 (6.48)	8.14 (6.48)
xd-vector (+VB)	-	-	7.93 (5.80)	**7.93 (5.80)**

TDNN based x-vector with Kaldi codes and LSTM based d-vector are combined and DER is obtained as %7.93. The Kaldi team reduced the SE to %6.48 by applying the Variational Bayes (VB) method to the x-vector method. In the study in reference [45], d-vectors are identified with %12 DER using the spectral clustering method. The training dataset used in the study also differs from the reference study. We observed %5.8 DER with the system developed using xd-vector and VB. The VB method relatively reduces the error rate by %25.

Table 4.8 Speaker diarization studies with using speaker representation module which tested with Callhome dataset

Comparison Table				
Algorithm	Error types			
	MS	FA	SE	DER
Shum et. al. [62]	-	-	14.5	14.5
Castaldo et. al. [63]	-	-	13.7	13.7
Senoussaoui et. al. [64]	-	-	12.1	12.1
Sell et. al. (+VB) [55]	-	-	13.7 (11.5)	13.7 (11.5)
Garcia-Romero et. al. (+VB) [65]	-	-	12.8 (9.9)	12.8 (9.9)
Zhang et. al. [46]	-	-	7.6	7.6
xd-vector (+VB)	-	-	**7.93 (5.80)**	**7.93 (5.80)**

5
RESULTS AND DISCUSSION

In this study, the effects of speech segmentation and speaker representation algorithms on speaker diarization systems are investigated. The speaker diarization system has been tested for two different scenarios and it has been observed that the speaker prior knowledge reduces the error rate.

Studies on the VAD module have been conducted and it has been observed that the TDNN-based model gives better results for speech segmentation than the other three models. Then, i-vector and x-vector extraction are made for speaker representation and they compared. It has been observed that the relative gain of x-vectors produced using TDNN is about %10 compared to i-vectors.

Then, solutions are sought for the speaker diarization problem by using deep learning-based embedding vectors. Firstly, TDNN-based x-vectors and LSTM-based d-vectors are used. Then, the xd-vectors formed by combining these two vectors are examined. An improvement in diarization performance is observed with xd-vectors created by combining two vectors.

Similarity scores are obtained using the PLDA method for clustering. In addition, the PLDA model is calibrated and low error rates are obtained with the PLDA model used. AHC, which is frequently used in this problem, is used for clustering and resegmented with Variational Bayes method. This system has been compared with other studies on the Callhome test set.

REFERENCES

[1] D. A. Reynolds and P. A. Torres-Carrasquillo, "Approaches and applications of audio diarization," *Proceedings. (ICASSP '05)*. *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, vol. 5, v/953–v/956 Vol. 5, 2005.

[2] H. Beigi, *Fundamentals of Speaker Recognition*. Springer Publishing Company, Incorporated, 2011, ISBN: 0387775919.

[3] L. R. Rabiner and R. W. Schafer, "Introduction to digital speech processing," *Foundations and Trends in Signal Processing*, vol. 1, no. 1–2, pp. 1–194, 2007, ISSN: 1932-8346. DOI: 10.1561/2000000001. [Online]. Available: http://dx.doi.org/10.1561/2000000001.

[4] A. Sakran, S. Abdou, S. E. Hamid Morsi Metwally, and M. Rashwan, "A review: Automatic speech segmentation," *International Journal of Computer Science and Mobile Computing*, vol. 6, Apr. 2017.

[5] L. Mary, *Extraction of Prosody for Automatic Speaker, Language, Emotion and Speech Recognition*. Springer International Publishing, 2019.

[6] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker diarization: A review of recent research," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 356–370, 2012.

[7] Z. Zajíc, M. Hrúz, and L. Müller, "Speaker diarization using convolutional neural network for statistics accumulation refinement," in *INTERSPEECH*, 2017.

[8] J. Markowitz, "Voice biometrics," *Communications of the ACM*, vol. 43, Sep. 2000. DOI: 10.1145/348941.348995.

[9] S. Hernawan, "Speaker diarization: Its developments, applications, and challenges," 2012.

[10] M. Rouvier, P. Bousquet, and B. Favre, "Speaker diarization through speaker embeddings," in *2015 23rd European Signal Processing Conference (EUSIPCO)*, 2015, pp. 2082–2086.

[11] R. Yin, H. Bredin, and C. Barras, "Neural speech turn segmentation and affinity propagation for speaker diarization," Sep. 2018, pp. 1393–1397. DOI: 10.21437/Interspeech.2018-1750.

[12] L. Rokach and O. Maimon, "Clustering methods," in. Jan. 2005, pp. 321–352. DOI: 10.1007/0-387-25465-X_15.

[13] A. K. Jain, R. P. W. Duin, and Jianchang Mao, "Statistical pattern recognition: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[14] T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: From features to supervectors," *Speech Commun.*, vol. 52, pp. 12–40, 2010.

[15] M. Toruk, A. Serbes, and G. Bilgin, "Speech segmentation and speaker diarization using time-delay neural network," in *2019 Innovations in Intelligent Systems and Applications Conference (ASYU)*, 2019, pp. 1–5.

[16] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 3, pp. 328–339, Mar. 1989, ISSN: 0096-3518. DOI: 10.1109/29.21701.

[17] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *INTERSPEECH*, 2015.

[18] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357–366, 1980.

[19] G. E. H. D. E. Rumelhart and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986.

[20] X. Zhang and D. Wang, "Boosting contextual information for deep neural network based voice activity detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 2, pp. 252–264, Feb. 2016, ISSN: 2329-9304. DOI: 10.1109/TASLP.2015.2505415.

[21] A. Aertsen and P. Johannesma, "Spectro-temporal receptive fields of auditory neurons in the grassfrog - i. characterization of tonal and natural stimuli," *Biological Cybernetics*, vol. 38, pp. 223–234, Nov. 1980. DOI: 10.1007/BF00337015.

[22] R. Schluter, L. Bezrukov, H. Wagner, and H. Ney, "Gammatone features and feature combination for large vocabulary speech recognition," vol. 4, May 2007, pp. IV–649. DOI: 10.1109/ICASSP.2007.366996.

[23] J. Chen, Y. Wang, and D. Wang, "A feature study for classification-based speech separation at very low signal-to-noise ratio," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 7039–7043.

[24] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *NIPS*, 2015.

[25] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," Dec. 2014.

[26] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, "Recurrent models of visual attention," *Advances in Neural Information Processing Systems*, vol. 3, Jun. 2014.

[27] J. Kim and M. Hahn, "Voice activity detection using an adaptive context attention model," *IEEE Signal Processing Letters*, vol. 25, no. 8, pp. 1181–1185, Aug. 2018, ISSN: 1558-2361. DOI: 10.1109/LSP.2018.2811740.

[28] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011, ISSN: 1558-7924. DOI: 10.1109/TASL.2010.2064307.

[29] M. M. Toruk and R. Gokay, "Short utterance speaker recognition using time-delay neural network," in *2019 16th International Multi-Conference on Systems, Signals Devices (SSD)*, 2019, pp. 383–386.

[30] D. Garcia-Romero and C. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems.," Jan. 2011, pp. 249–252.

[31] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[32] A. Poddar, M. Sahidullah, and G. Saha, "Quality measures for speaker verification with short utterances," *Digital Signal Processing*, vol. 88, pp. 66–79, 2019, ISSN: 1051-2004. DOI: https://doi.org/10.1016/j.dsp.2019.01.023. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1051200418304287.

[33] D. Reynolds, T. Quatieri, and R. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing*, vol. 10, pp. 19–41, Jan. 2000. DOI: 10.1006/dspr.1999.0361.

[34] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

[35] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A study of interspeaker variability in speaker verification," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 16, pp. 980–988, Aug. 2008. DOI: 10.1109/TASL.2008.925147.

[36] P. Kenny, G. Boulianne, and P. Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 345–354, 2005.

[37] G. Sell and D. Garcia-Romero, "Speaker diarization with plda i-vector scoring and unsupervised calibration," *2014 IEEE Workshop on Spoken Language Technology, SLT 2014 - Proceedings*, pp. 413–417, Apr. 2015. DOI: 10.1109/SLT.2014.7078610.

[38] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5329–5333.

[39] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, Dec. 2016, pp. 165–170. DOI: 10.1109/SLT.2016.7846260.

[40] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 5796–5800.

[41] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2017, pp. 5220–5224. DOI: 10.1109/ICASSP.2017.7953152.

[42] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," Oct. 2015.

[43] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, Dec. 1997. DOI: 10.1162/neco.1997.9.8.1735.

[44] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, *Generalized end-to-end loss for speaker verification*, 2017.

[45] Q. Wang, C. Downey, L. Wan, P. Mansfield, and I. Moreno, "Speaker diarization with lstm," Oct. 2017.

[46] A. Zhang, Q. Wang, Z. Zhu, J. Paisley, and C. Wang, "Fully supervised speaker diarization," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6301–6305.

[47] S. Ioffe, "Probabilistic linear discriminant analysis," in *ECCV*, 2006.

[48] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matejka, and N. Brummer, "Discriminatively trained probabilistic linear discriminant analysis for speaker verification," May 2011, pp. 4832–4835. DOI: 10.1109/ICASSP.2011.5947437.

[49] S. Prince and J. Elder, "Probabilistic linear discriminant analysis for inferences about identity," Jan. 2007, pp. 1–8. DOI: 10.1109/ICCV.2007.4409052.

[50] S. Cumani, N. Brummer, L. Burget, P. Laface, O. Plchot, and V. Vasilakakis, "Pairwise discriminative speaker verification in the i-vector space," *IEEE Transactions on Audio Speech and Language Processing*, vol. 21, pp. 1217–1227, Jun. 2013. DOI: 10.1109/TASL.2013.2245655.

[51] Y. Jiang, K. A. Lee, and L. Wang, "Plda in the i-supervector space for text-independent speaker verification," Jul. 2014. DOI: 10.1186/s13636-014-0029-2.

[52] D. Van Leeuwen and N. Brummer, "The distribution of calibrated likelihood-ratios in speaker recognition," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, Apr. 2013.

[53] G. Sell, A. McCree, and D. Garcia-Romero, "Priors for speaker counting and diarization with ahc," Sep. 2016, pp. 2194–2198. DOI: 10.21437/Interspeech.2016-1380.

[54] P. Kenny, P. Kenny@crim, and Ca, "Bayesian analysis of speaker diarization with eigenvoice priors," Jan. 2008.

[55] 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.

[56] M. Diez, L. Burget, and P. Matejka, "Speaker diarization based on bayesian hmm with eigenvoice priors," Jun. 2018, pp. 147–154. DOI: 10.21437/Odyssey.2018-21.

[57] P. Kenny, D. Reynolds, and F. Castaldo, "Diarization of telephone conversations using factor analysis," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, pp. 1059–1070, Jan. 2011. DOI: 10.1109/JSTSP.2010.2081790.

[58] O. Galibert, "Methodologies for the evaluation of speaker diarization and automatic speech recognition in the presence of overlapping speech," Aug. 2013.

[59] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz, J. Silovský, G. Stemmer, and K. Vesel, "The kaldi speech recognition toolkit," *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, Jan. 2011.

[60] J. Garofolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallett, N. Dahlgren, and V. Zue, "Timit acoustic-phonetic continuous speech corpus," *Linguistic Data Consortium*, Nov. 1992.

[61] C. Cieri, D. Miller, and K. Walker, "The fisher corpus: A resource for the next generations of speech-to-text," Jan. 2004.

[62] S. H. Shum, N. Dehak, R. Dehak, and J. R. Glass, "Unsupervised methods for speaker diarization: An integrated and iterative approach," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2015–2028, 2013.

[63] F. Castaldo, D. Colibro, E. Dalmasso, P. Laface, and C. Vair, "Stream-based speaker segmentation using speaker factors and eigenvoices," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008, pp. 4133–4136.

[64] M. Senoussaoui, P. Kenny, T. Stafylakis, and P. Dumouchel, "A study of the cosine distance-based mean shift for telephone speech diarization," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 1, pp. 217–227, 2014.

[65] D. Garcia-Romero, D. Snyder, G. Sell, D. Povey, and A. McCree, "Speaker diarization using deep neural network embeddings," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 4930–4934.

PUBLICATIONS FROM THE THESIS

Contact Information: mesut.toruk@tubitak.gov.tr

Conference Papers

1. Mesut Toruk, Ahmet Serbes and Gökhan Bilgin, "Speech segmentation and speaker diarization using time-delay neural network," in 2019 Innovations in Intelligent Systems and Applications Conference (ASYU), 2019, pp. 1–5

2. Mesut Toruk, Gökhan Bilgin and Ahmet Serbes, "Speaker Diarization using Embedding Vectors", Signal Processing and Communication Applications (SIU) Conference, 2020 (Accepted)