



North South University

Department of Electrical and Computer Engineering

School of Engineering and Physical Sciences (SEPS)

Senior Project Design

Transfer Learning for Speaker Diarization on Bangla Audio Dataset

A.N.M Fahim Faisal – 1711758642

Umnoon Binta Ali – 1713013042

Md. Rayhan Talukder – 1712356642

Md. Afikur Rahman – 1711035042

Faculty Advisor

Ms. Tanjila Farah

Senior Lecturer

Summer 2021

DECLARATION

We, hereby, declare that the work presented in this report is the outcome of our four months' work performed under the supervision of Ms. Tanjila Farah, Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh. The work was spread over a span of one of the final year courses, CSE 499A and CSE499B, Senior Design Project I and II, in accordance with the course curriculum of the Department for the Bachelor of Science in Electrical and Electronics Engineering program.

Students' name & signature:

fahimfaisal

A.N.M Fahim Faisal

Umnoo

Umnoon Binta Ali



Md. Rayhan Talukder

Afikur Rahman

Md. Afikur Rahman

Approval

The senior project report on ‘Transfer Learning Model in Speaker Diarization on Bangla Audio Dataset’ has been submitted by MD. Rayhan Talukder (1712356642), A.N.M Fahim Faisal (1711758642), Md. Afikur Rahman (1711035042), Umnoon Binta Ali (1713013042) students of the Department of Electrical and Computer Engineering, North South University, Bangladesh. This report partially fulfills the requirement for the degree of Bachelor of Science in Electrical and Electronics Engineering in December 2019 and has been accepted as satisfactory.

Supervisor’s Signature

Ms. Tanjila Farah

Senior Lecturer

Department of Electrical and Computer Engineering North South University, Dhaka,
Bangladesh

Department Chair’s Signature

Dr. Mohammad Rezaul Bari

Associate Professor and Chair

Department of Electrical and Computer Engineering North South University, Dhaka,
Bangladesh

Acknowledgments

First, we express our gratitude to almighty ALLAH for His blessing which makes us possible to complete the project. We are grateful and wish our profound indebtedness to Ms. Tanjila Farah, Senior Lecturer, Department of ECE, North South University, Dhaka. Deep Knowledge & keen interest of our supervisor in the field of “Deep Learning” to carry out this project. Her guidance, constant supervision, enthusiastic encouragement, sagacious advice, and effective surveillance throughout the entire period of the project have made it possible to complete this project.

We would like to thank our entire CSE 499 course mate in North South University, who took part in this discussion while completing the course work. At last, we must express our sincere heartfelt gratitude to all the staff members of the Computer Engineering Department who helped us directly or indirectly during this course of work.

Abstract

Sound classification is a very intriguing concept in the field of artificial intelligence. Speaker Diarization is a very interesting task in the domain of sound classification. It has recently expanded significantly with the introduction of deep learning technology, which has transformed research and practices throughout speech application fields. Speaker diarization is the process of assigning labels to audio data that match the speaker's identity. It is quite beneficial when it comes to identifying audio information. In Bangla, very little work has been done on speaker diarization. This project aimed to build a Bangla dataset for the diarization process in this research. We described our deep learning model for the speaker diarization problem and demonstrated how transfer learning can be utilized to swiftly learn a model with minimal performance loss when compared to a fully trained one. To increase the universal applicability of our model, we focused on transfer learning and tweaked it manually across AMI and Bangla dataset. Additionally, we've been focusing our efforts on improving the Diarization Error Rate (DER) and experimenting with other embedding generation networks. We obtained a DER score of 0.24 using our transfer learning variation trained on Bangla dataset.

Table of contents

DECLARATION	2
Approval	3
Acknowledgments	4
Abstract	5
Table of contents	6
List of tables	10
List of figures	11
Chapter: 1	12
OVERVIEW	12
1.1 Introduction	13
1.2 Project Description	13
1.3 Project Goals	14
1.4 Contribution	15
1.5 Organization	15
Chapter 2	16
Motivation	16
2.1 Background	17
2.2 Motivation toward the project	17
Chapter 3	18
Index terms	18
3.1 Deep Learning	19
3.2 CNN (Convolutional Neural Networks)	20

3.3 LSTM (Long Short-Term Memory Networks)	21
3.4 Bi-LSTM (Bidirectional Long Short-Term Memory Networks)	22
3.5 Speech Processing	23
3.6 MFCC (Mel frequency cepstral coefficients)	23
3.7 Log-mel spectrum	24
3.8 Speech Recognition	24
3.8.1 Hidden Markov Models (HMM)	25
3.8.2 N-grams	25
3.8.3 Neural Networks	25
3.9 Speaker Diarization	26
3.10 Diarization	26
3.11 Voice activity detection	26
3.12 Clustering algorithm	27
3.13 Embeddings	28
3.14 DER calculation	29
3.15 Transfer learning	30
3.16 Domain adaptation	31
3.17 Hyperparameter tuning	32
Chapter 4:	34
Literature Survey	34
4.1 Related works	35
Chapter 5	40
Dataset	40
5.1 AMI Dataset	41
5.2 Bangla Dataset	41
Chapter 6	45

Methodology and Model Architecture	45
6.1 Proposed method	46
6.2 Pre-processing	48
6.2.1 AMI-CNN and AMI-LSTM	48
6.2.2 Bi-LSTM Transfer learning	50
6.3 Voice activity detection	50
6.3.1 Energy-based VAD	50
6.3.2 Neural Network-Based VAD (LSTM)	50
6.3.3 GMM Based VAD (WebRTC)	51
6.4 Embeddings	54
6.4.1 Resemblyzer	54
6.4.1.1 Voice similarity metric	54
6.4.1.2 High-level feature extraction	54
6.5 Clustering Algorithms	54
6.5.1 Spectral Clustering	55
6.5.2 Hierarchical Density-Based Spatial Clustering (HDBSCAN)	55
6.5.3 Mean Shift	55
6.6 DER calculation	57
6.7 Experimental Setup	58
6.7.1 Hardware configuration	58
6.7.2 Software configuration	59
Chapter 7	61
Transfer Learning Variants	61
7.1 Transfer learning	62
7.1.1 Feature extraction	62
7.1.2 Combining Models	63

7.2 Hyperparameters tuning (Knob tuning)	63
Chapter 8	65
Result analysis	65
8.1 Result visualizations	66
8.2 Performance comparison	67
Chapter 9	68
Project Summary	68
9.1 Conclusion	69
9.2 Future work	69
Chapter 10	71
Bibliography	71
Reference	72

List of tables

Table 1: DER comparison for Resemblyzer, Spectral combined approach	55
Table 2: Train and Test DER comparison between all three Embedding architectures	62
Table 3: Comparison between the variants	62

List of figures

Figure 1: Visual representation of Speaker Diarization	13
Figure 2: A deep learning neural network consisting of multiple hidden layers	19
Figure 3: A CNN sequence to classify handwritten digits	20
Figure 4: LSTM cell and its operation	21
Figure 5: Bi-LSTM model architecture	22
Figure 6: Traditional ML vs Transfer learning	30
Figure 7: A working domain adaptation	31
Figure 8: Hyperparameter tuning vs model training	33
Figure 9: Traditional Speaker Diarization System	36
Figure 10: Collection of Audios for Bangla dataset with relevant annotated information	43
Figure 11: Speaker utterances for a single Bangla meeting audio	44
Figure 12: Proposed Approach for Diarization	47
Figure 13: Model Architecture for AMI-CNN	49
Figure 14: Model Architecture for AMI-LSTM	49
Figure 15: Wave signal for an input audio	52
Figure 16: Energy based VAD (log-mel spectrum)	52
Figure 17: LSTM based VAD	53
Figure 18: GMM based VAD (WebRTC)	53
Figure 19: Clustering of embedding vector using different algorithm – (a) True Labels (b) Google’s Spectral Clustering (c) HDBSCAN (d) Nearest Neighbors (e) Radial Basis	56
Figure 20: Spectral Plots with their respective Gaussian Blur Sigma Values ($\sigma = 0.25$ and $\sigma = 0.50$)	63
Figure 21: Spectral Plots with their respective Gaussian Blur Sigma Values ($\sigma = 0.75$, $\sigma = 1$ and Ground truth)	64
Figure 22: Plots of Hypothesis & Reference for Bi-LSTM model (Train DER: 0.114)	66
Figure 23: Plots of Hypothesis & Reference for Bi-LSTM model Test (DER: 0.24)	66

Chapter: 1

OVERVIEW

1.1 Introduction

Speaker diarization is a field in Machine Learning. The goal of speaker diarization is to segment and classify a speech by speakers. It achieves an important task of “who spoke when”. Recently speaker diarization has become a very important topic as deep learning technologies improve [1]. But regarding Bengali sound classification, there hasn’t been a lot of work done in the past years. Along with speaker diarization, sound classification is also a vast field and mostly unexplored in Bangladesh’s context.

Since diarization is a process of classifying sounds and labeling them accordingly, it is of great importance. Speaker diarization lessens the work of writing descriptions from listening to audio specimens. A variety of classes and their detailed information opens a huge door of possibility [1].

The chance of creating synthetic or machine-produced images gives the whole process a new dimension. It takes a huge amount of actual data and training time to produce such a system that creates machine images from the information of diarized sounds. Here, we plan to introduce another helpful technology, Speech to Text (STT) which is also known as Speech to text transcription. Transcription is a process where a pile of input sound/audio gets converted into readable texts. Overall the whole process of diarization is very good as well as an interesting use of data which opens a huge possibility and variety of scopes in the sphere of Artificial Intelligence.

1.2 Project Description

Speaker diarization consists of segmenting and clustering a speech recording into speaker-homogenous regions, using an unsupervised algorithm. In other words, given an audio track of a meeting, a speaker-diarization system will automatically discriminate between and label the different speakers (“Who spoke when?”). This involves speech/non-speech detection (“When is there speech?”) and overlap detection and resolution (“Who is overlapping with whom?”), as well as speaker identification. A visual representation of Speaker Diarization is given in Figure 1. The first figure shows the input audio and the segmented and labelled audio with speaker information.

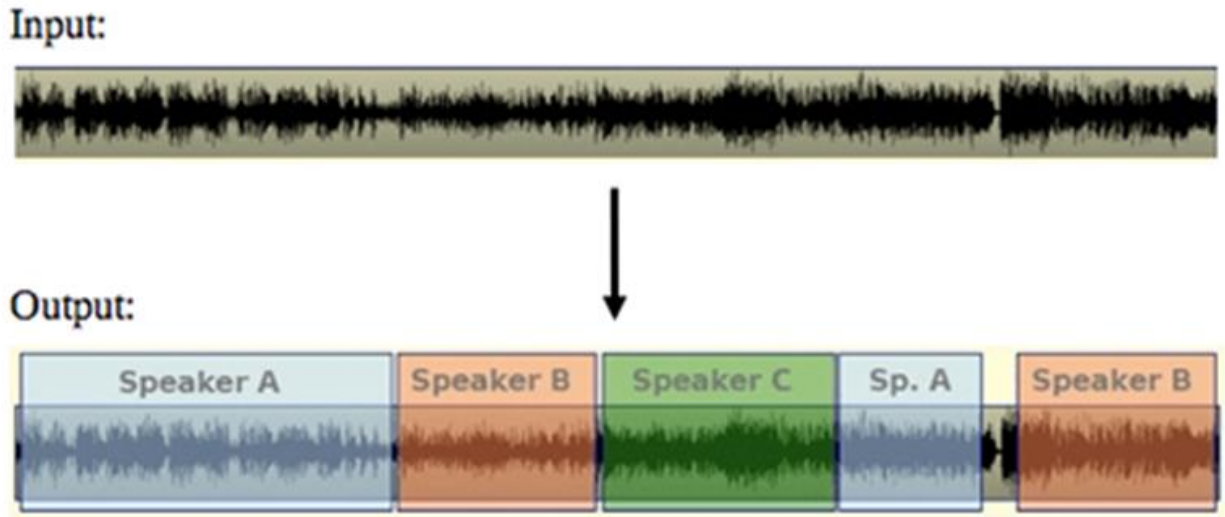


Figure 1: Visual representation of Speaker Diarization.

As shown in Figure 1, in a diarization system an input audio wave is given and the corresponding output comes out as the speaker labelled wave form with their respected time. The most common way is to take the audio waveform, then split it at utterances by silences and finally try to recognize what's being said in each utterance. This process is done by taking all possible combinations of words and then trying to match them with the audio to find the best matching combination.

1.3 Project Goals

Our research focuses on diarizing small clips of sounds. A dataset will contain an adequate number of sound clips. With the dataset, we will work on the first phase which is diarization. We will retrieve information like speaker number, speaker gender, speech time, noise profile, etc. We will finally make a machine learning model that will perform speaker diarization given an input. So, the whole project can be categorized into 2 categories. Which are – Examine different methods of diarization and Create Bangla dataset to run compatible models for best outputs.

To summarize, our goal is to build a prototype interface that can analyze large audio data of meetings (or such things) containing multiple speakers and detect numbers of speakers with lines from each speaker as output.

1.4 Contribution

The contribution of our project is as following-

- Prepare a Bangla Dataset for speaker diarization.
- Perform speaker diarization on the existing AMI dataset using our own model.
- Perform speaker diarization on Bangla dataset.
- Evaluate the results and compare.

1.5 Organization

This report is divided into ten chapters. Chapter one is the overview of the experiment includes the overall project description and project goals. In Chapter 2 we described the background of the project. Chapter three is a thorough explanation of the basic index terms that we used throughout the project. In chapter four there is a detailed literature survey. Chapter five followed by Chapter six describes the technical aspects of the research. Structural design of both the customized dataset and proposed model are described in this two chapter. Chapter six specifically focuses on describing the architectures of different models that we created and ran on the datasets. In chapter seven we explained about the techniques we followed for improving our model for the Bangla dataset using Transfer learning. In chapter eight we stated all the results we got and did a comparative analysis of the models proposed. Chapter nine and ten contains Project summary and references.

Chapter 2:

Motivation

2.1 Background

Figure 1 depicts a typical speaker recognition system. The complete system, as shown in Figure 1, can be divided into two categories of processes: offline and online processes [10] [11]. The offline approach entails developing a training method for producing a background model based on non-target speaker utterances. Here the term "utterance" refers to a segment of spoken language that is preceded by silence and followed by quiet or a change in speaker. The online procedure, on the other hand, entails the speaker recognition system processing in real time, using learned models, to identify the speaker in a new utterance.

After extracting features from non-target speakers, the initial stage is to train a base or background model, such as the Universal Background Model (UBM). After feature extraction, the training set of utterances from the target speakers is fed into this model, resulting in the customized targeted model. Pattern matching is used to test this model on the utterance of an unknown speaker, which provides the speaker a probability score. After that, the score is normalized, and the speaker's identity is determined. The following three steps can be used to summarize the complete procedure, including speaker diarization.

2.2 Motivation toward the project

While Speaker Diarization research drew a lot of attention in the past, it has become rather stagnant in recent years. This could be due to a scarcity of large datasets that look at areas outside of specific categories especially in Bangla. A large amount of audio-video data is circulating all over the internet every day. More and more people are getting connected to the internet and uploading videos and audios of different topics. These videos often need annotation. But no such thing that is automatically annotated with proper labeling of speakers exists at this moment for Bengali contents. Also, for large audio data - to summarize, synthetic or computer-processed, data-driven audio production also adds a new dimension to the content captioning or summarizing. That's where we came up with the idea to create a system that can do the diarization of large audio files and create annotated dataset to be used for multi-purpose fields in machine learning.

Chapter 3:

Index terms

3.1 Deep Learning

As a subfield of machine learning, deep learning is an artificial neural network which includes multiple layers to deliver a highly accurate result in speech recognition, natural language processing, pattern recognition, etc. As it is built upon neural network architectures, sometimes deep learning is also referred to as “Deep neural networks”. Deep learning varies from typical machine learning approaches in that it can learn patterns from data such as photos, video or text without requiring human intervention or hard-coded rules [4]. Their highly adaptable architecture lets them learn directly from raw data and improvise their accuracy when given additional data.

Shown in Figure 2, An input layer receives the initial data, while an output layer creates the final prediction in all neural networks. However, between these input and output layers of a deep neural network, there will be several "hidden layers" of neurons flowing data into each other.

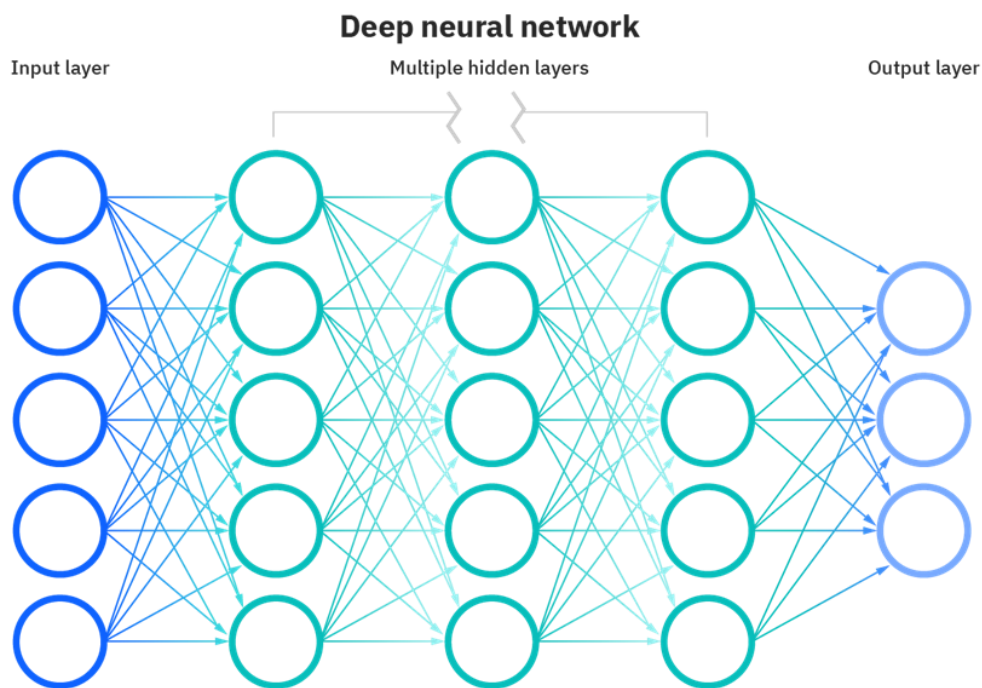


Figure 2: A deep learning neural network consisting of multiple hidden layers [4].

3.2 CNN (Convolutional Neural Networks)

CNN is one of the deep learning algorithms that is applied to analyze visual imagery. It takes images as input then assigns weights and biases based on various aspects or objects in the image and learns to differentiate one from the other. Identifying things in photos before CNNs required laborious, time-consuming feature extraction approaches. The higher performance of convolutional neural networks with image, voice or audio signal inputs sets them apart from conventional neural networks [5]. They are divided into three sorts of layers: Convolutional layer, Pooling layer, Fully-connected (FC) layer.

A convolutional network's initial layer is the convolutional layer. While further convolutional layers or pooling layers can be added after convolutional layers, the fully-connected layer is the last layer. The CNN becomes more complicated with each layer, detecting larger areas of the picture. Earlier layers concentrate on basic features like colors and edges. As shown in Figure 5, the visual data travels through the CNN layers, it begins to distinguish bigger components or features of the object, eventually identifying the target object.

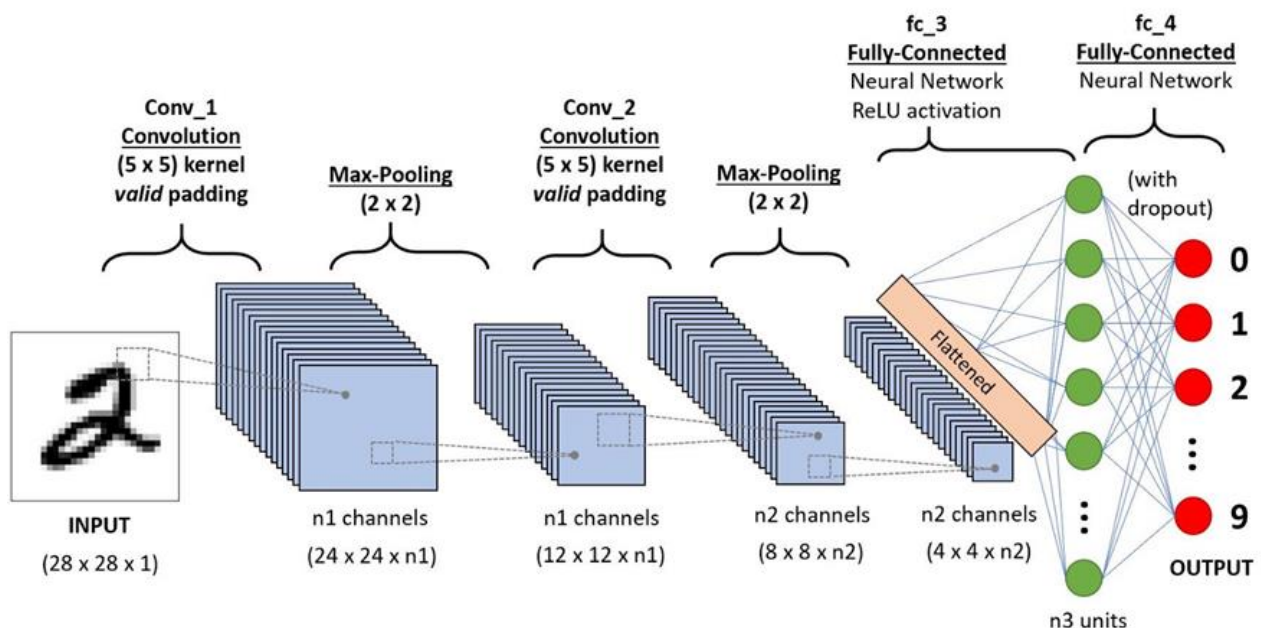


Figure 3: A CNN sequence to classify handwritten digits [5].

3.3 LSTM (Long Short-Term Memory Networks)

Short-term memory is a problem for recurrent neural networks (RNNs). They'll have a hard difficulty transporting information from early stage to later ones if the sequence is lengthy enough. While attempting to predict anything from a paragraph of text, RNNs may leave out critical information at the start. The vanishing gradient problem affects recurrent neural networks during back propagation. Gradients are values that are used to update the weights of a neural network. When a gradient reduces as it propagates through time, this is known as the vanishing gradient issue. When a gradient value falls below a certain threshold, it no longer contributes much to learning. Layers that get a minimal gradient update in RNNs stop learning. Those are frequently the first layers to appear. RNNs can forget what they've seen in longer sequences since these layers don't learn, resulting in a short-term memory. LSTMs were developed as a short-term memory solution. They feature inbuilt processes known as gates that may control the flow of data. These gates can figure out which data in a sequence should be kept and which should be discarded. It can then transfer important information down the lengthy chain of sequences to create predictions as a result of this. These two networks are responsible for nearly all state-of-the-art recurrent neural network findings [6]. Voice recognition, speech synthesis, and text production all use LSTMs. It can even be used to create video captioning.

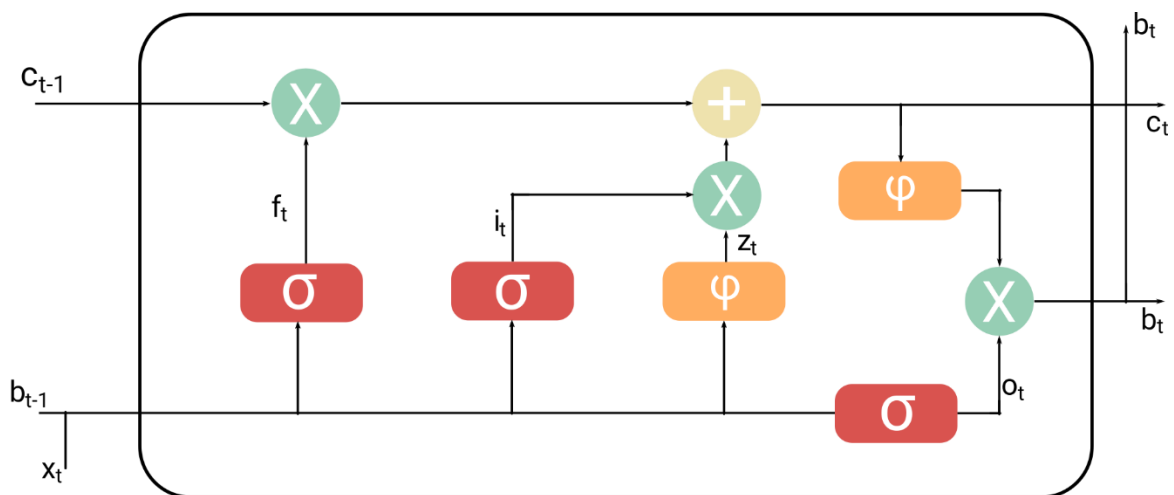


Figure 4: LSTM cell and its operation.

The cell state and its many gates are at the base of LSTMs. The cell state serves as a transportation route for relative information as it travels down the sequence chain. You might think of it as the network's "memory." In principle, the cell state can carry meaningful information throughout the sequence's processing. As a result, information from earlier time steps might make its way to later time steps, lessening the short-term memory effects. Information is added or withdrawn from the cell state via gates as the cell state travels. The gates are several neural networks that determine whether information about the cell state is permitted. During training, the gates might learn what information is important to preserve or forget.

3.4 Bi-LSTM (Bidirectional Long Short-Term Memory Networks)

Bi-LSTM as shown in Figure 5, is a model architecture that connects two independent RNNs. At each time step, this structure allows the networks to have both forward and backward data flow about the sequence. Bidirectional data inputs will go in two directions, one from the past to the future and the other from the future to the past. What distinguishes this technique from unidirectional is that in a backward-running LSTM, we must maintain information from the future, but by combining the two hidden states, we may preserve information from both the past and the future at any point in time.

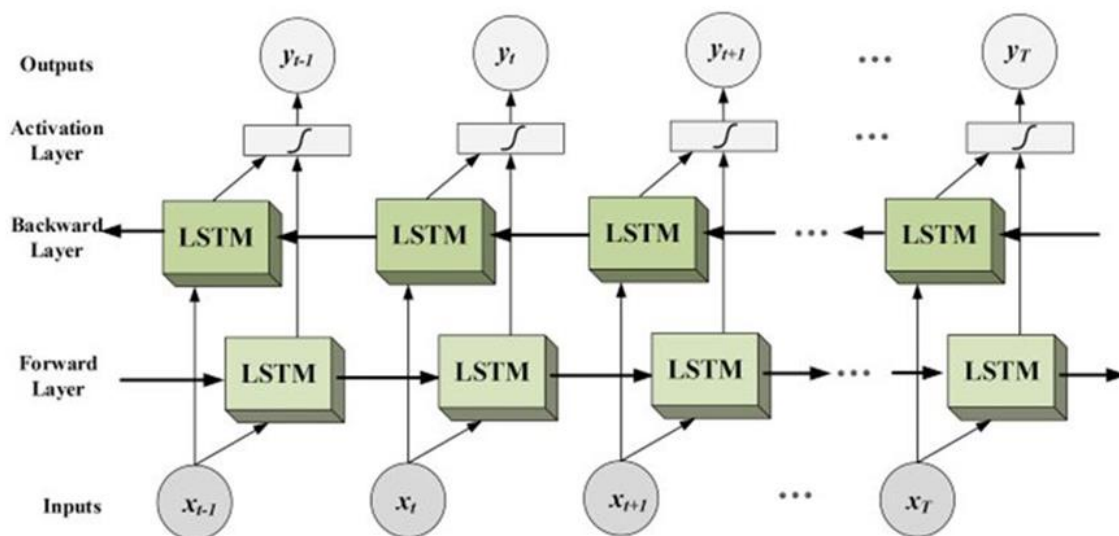


Figure 5: Bi-LSTM model architecture [7].

The flow of information from backward and forward layers may be seen in the figure. When tasks requiring sequence to sequence are needed, BI-LSTM is typically used [7]. Text classification, speech recognition, and forecasting models can all benefit from this type of network

3.5 Speech Processing

Speech processing is the process of interpreting, comprehending, and acting on speech signals. It refers to digital systems that process human speech, such as voice recognition software or text-to-speech applications. Intelligent gadgets, such as smartphones, can interact with consumers via vocal language thanks to speech processing [8]. Speech processing includes a wide range of scientific and technology fields. For storing or transmitting voice, it incorporates speech analysis and variable rate coding. Voice synthesis, particularly from text, speech recognition, including speaker and language identification, and spoken language comprehension are also included. Apple's Siri, Microsoft's Cortana, and Amazon's Alexa are some of the highly accurate speech recognition services available right now.

There are several techniques available for processing natural language or speeches. Some notable techniques are, Dynamic time warping, Hidden Markov models, Artificial neural networks, Phase-aware processing.

3.6 MFCC (Mel frequency cepstral coefficients)

MFCCs are utilized in speech recognition to extract the speech signals stimuli from speech. The goal of MFCC is to convert time-domain audio into frequency-domain audio so that we can comprehend all of the information in speech signals. However, just translating time-domain signals to frequency domain signals may not be the best solution. More than simply translating time-domain signals to frequency domain signals is possible. Our ear has a cochlea, which contains a lot of low-frequency filters and very few high-frequency filters. Mel filters can be used to imitate this. So, the idea behind the goal of MFCC is to transform time-domain signals into frequency-domain signals by utilizing Mel filters to simulate the cochlea function.

The following is the fundamental technique for creating MFCCs:

- Convert between Hertz and Mel Scale.
- Take the logarithm of Mel's audio representation.
- Discrete Cosine Transformation is applied on logarithmic magnitude.
- As a result, a spectrum is created across Mel frequencies rather than time, resulting in MFCCs.

3.7 Log-mel spectrum

Mel Spectrograms are sound spectrograms that display sounds on the Mel scale rather than the frequency domain. A mel spectrogram depicts frequencies over a specific threshold logarithmically (the corner frequency). The vertical space between 1,000 and 2,000Hz, for example, is half of the vertical space between 2,000Hz and 4,000Hz in the linearly scaled spectrogram. The distance between the ranges on the mel spectrogram is almost the same. This scaling is related to human hearing, in which we can discern between similar low-frequency noises more easily than similar high-frequency ones.

3.8 Speech Recognition

Speech recognition, also known as automatic speech recognition (ASR), computer speech recognition, or speech-to-text, is a feature that allows a computer program to convert human speech into written text. While it's sometimes mistaken with voice recognition, speech recognition is concerned with converting speech from a verbal to a written format, whereas voice recognition is solely concerned with identifying a single user's voice. In summary, speech recognition is a device's capacity to respond to spoken commands. There are several voice recognition programs and devices available, but the most advanced options rely on artificial intelligence and machine learning. To interpret and analyze human speech, they combine grammar, syntax, structure, and composition of audio and voice signals. They should, in theory, learn as they go, changing their reactions with each engagement. While evaluating a speech recognition program its word error rate and speed are some important factors. Pronunciation, accent, pitch, loudness, and background noise are all characteristics that might influence word error rate. Speech

recognition systems have long aspired to achieve human parity, or an error rate comparable to that of two humans conversing.

To convert speech to text and increase transcription accuracy, a variety of algorithms and computational strategies are applied. The following are some of the most regularly utilized approaches:

3.8.1 Hidden Markov Models (HMM)

Hidden Markov Models are based on the Markov chain model, which states that the probability of a given state is determined by its present state, rather than its previous states. While a Markov chain model is good for observable events like text inputs, HMM let us include hidden events like part-of-speech markers in a probabilistic model. They're used as sequence models in speech recognition, giving labels to each item in the sequence—words, syllables, phrases, and so on. These labels form a mapping with the input, allowing it to choose the best label sequence.

3.8.2 N-grams

This is the most basic form of language model, in which sentences or phrases are assigned probability. An N-gram is a collection of N words. "Cancel the delivery" for example, is a trigram or 3-gram, whereas "Kindly cancel the delivery" is a 4-gram. To increase recognition and accuracy, grammar and the probability of particular word sequences are applied.

3.8.3 Neural Networks

Neural networks, which are mostly used for deep learning algorithms, analyze training data by simulating the interconnection of the human brain through layers of nodes. Inputs, weights, a bias (or threshold), and an output make up each node. If the output value reaches a certain threshold, the node fires or activates, sending data to the network's next tier. Through supervised learning, neural networks learn this mapping function, then alter it depending on the loss function using gradient descent. While neural networks are more accurate and can handle more input, they have a lower performance efficiency than classic language models since they take longer to train.

3.9 Speaker Diarization

Speaker diarization algorithms recognize and segment speech based on the identification of the speaker. This enables programs to differentiate between people in a discussion and is commonly used in contact centers to distinguish between customers and salespeople.

3.10 Diarization

Speaker diarization is one kind of speech processing where it is necessary to identify the speaker as well as the boundary or frame of the speech delivered by that speaker. The task of segmenting and co-indexing audio recordings by speaker is known as speaker diarization. The purpose of the job, as it is typically characterized, is to co-index segments that are attributed to the same speaker, rather than to identify recognized speakers [1]. Diarization helps identify the number of separate speakers by locating speaker boundaries and grouping segments that belong to the same speaker. Diarization, when used in conjunction with voice recognition, allows for speaker-attributed speech-to-text transcription. Speaker diarization transcription includes slicing an audio recording file into single-speaker segments and embedding the speech segments into a space that reflects each individual speaker's unique qualities. The segments are then grouped and ready for labeling. Speaker diarization systems follow a step-by-step process that includes: Speech Detection, Speech Segmentation, Embedding Extraction, Clustering, Labeling Clusters, Transcription.

3.11 Voice activity detection

Voice activity detection (VAD) is used to separate an audio stream into time intervals that include speech activity and time intervals where speech is missing in various speech signal processing applications. For each frame of the noisy signal, voice activity detection normally involves a binary choice on the occurrence of speech. Extensions of VAD include methods for locating speech chunks in the temporal and frequency domain, such as speech presence probability (SPP) and ideal binary mask (IBM) estimation. The majority of VAD algorithms may be broken down into two steps of processing:

- To develop a representation that distinguishes between speech and noise, characteristics are first retrieved from the noisy speech signal.
- In the next stage, the features are subjected to a detection technique, which leads to the final conclusion.

3.12 Clustering algorithm

Clustering is the process of grouping data in such a manner that objects belonging to one group share many common features with one another while remaining distinct from objects belonging to other groups created throughout the process. Clustering algorithms take data and arrange it into groups based on some type of similarity metric; these groups may then be utilized in a variety of business processes [9] such as information retrieval, pattern recognition, image processing, data compression, bioinformatics, and so on. A distance-based similarity measure plays a critical role in determining grouping in the machine learning process. There are numerous types of clustering techniques that tackle one or more of these difficulties, as well as several statistical and machine learning clustering algorithms that apply the approach, to meet the concerns – scalability, dimensionality, attributes, noise, interpretation and boundary shape. Some of the clustering algorithms are:

- Connectivity-based Clustering (Hierarchical clustering)
- Centroids-based Clustering (Partitioning methods)
- Distribution-based Clustering
- Density-based Clustering (Model-based methods)
- Fuzzy Clustering
- Constraint-based (Supervised Clustering)

3.13 Embeddings

The term "embedding" refers to a technique for representing discrete variables as continuous vectors. Embeddings are low-dimensional, learnt continuous vector representations of discrete variables in neural networks. Because they can lower the dimensionality of categorical data and properly represent categories in the converted space, neural network embeddings are helpful [1]. The objective of neural network embeddings is threefold:

- In the embedding space, find the closest neighbors. These can be used to give suggestions depending on the user's preferences or cluster categories.
- For a supervised task, providing input to a machine learning model.
- Concepts and relationships between categories may be visualized using this application.

The drawbacks of a typical approach for encoding categorical data, one-hot encoding, are solved via neural network embeddings. The approach of one-hot encoding has two major drawbacks:

- The dimensionality of the converted vector becomes unmanageable for high-cardinality variables – those with a large number of distinct categories.
- The mapping is erroneous to begin with. In embedding space, related categories aren't clustered together.

The fundamental disadvantage of one-hot encoding is that it does not require any supervision. Learning embeddings with a neural network on a supervised task can considerably enhance them. The embeddings are used to create the network's parameters, or weights, which are adjusted to minimize task loss. The embedded vectors that arise are representations of categories, with identical categories being closer together in relation to the task. Embeddings are a powerful method of dealing with discrete variables, and they're a great place to start with deep learning.

3.14 DER calculation

The diarization output is assessed using the standard NIST-RT (National Institute of Standards and Technology - Rich Transcription) measure of Diarization Error Rate or DER [10]. The total of speech/non-speech and speaker error is known as DER. To calculate the DER, the best one-to-one mapping of reference speakers to system output speakers is needed to be found first. The DER is then calculated by dividing the total speech time in an audio file by the total amount of the per speaker false alarm time (the period of time the system overestimates the number of speakers), miss time (the period of time the system underestimates the number of speakers), and speaker error time (the amount of time the speculated speaker(s) is (are) not matched to the reference speaker(s). To avoid penalizing minor differences in speech segment start and finish timings, a 0.25 second no-score collar is put at the beginning and end of each segment boundary.

3.15 Transfer learning

In machine learning, transfer learning refers to the reuse of a previously learned model on a new problem. In transfer learning, a computer leverages information from a previous assignment to improve prediction about a new assignment. During transfer learning, the information of a previously trained machine learning model is transferred to a separate but closely related task. We strive to apply what we've learned in one activity to better comprehend concepts in another through transfer learning [11]. Weights are automatically changed from a network that conducted new "task B" to a network that is completing "task A." Transfer learning is commonly used in computer vision and natural language processing applications like sentiment analysis due to the enormous amount of CPU resources required.

When we don't have enough annotated data to train our model with or there is a pre-trained model that has been trained on similar data and tasks, we apply transfer learning. TensorFlow may simply restore and retrain certain layers for the job if it was used to train the original model. On the other hand, transfer learning only works if the features learned in the first task are generalizable to other activities [12]. The key benefits of transfer

learning are that it saves time while training, that the Neural Network performs better in most circumstances, and that we don't require a large amount of data. To train a Neural Network from the start, a lot of data is usually required, however we don't always have access to enough data. That's where Transfer Learning comes in, because it allows us to build a good machine learning model with very minimal training data because the model has already been pre-trained. This is especially useful in Natural Language Processing (NLP), because huge labeled datasets demand a lot of expert knowledge as shown in Figure 6. As a result, a significant amount of training time may be saved, as training a deep Neural Network from the start on a demanding job might take days or even weeks.

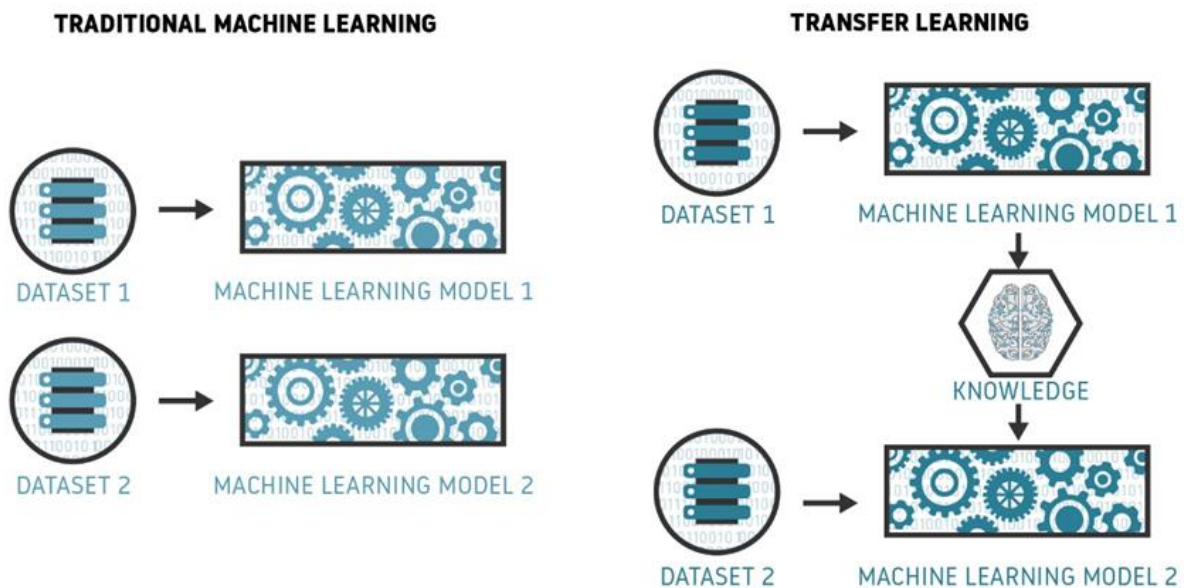


Figure 6: Traditional ML vs Transfer learning [12].

3.16 Domain adaptation

The training and test sets in traditional machine learning are assumed to come from the same distributions. As a result, a model learned from labeled training data should do well on the test data. However, in real-world applications where the training and test data come from different distributions, this assumption may not always hold due to a variety of factors, such as collecting the training and test sets from different sources or having an out-of-date training set due to data changes over time [13]. There would be a disparity

between domain distributions in this situation, and applying the trained model to the new dataset naively might result in performance loss. Domain adaptation is a branch of machine learning that seeks to solve challenges like these by aligning domain disparities so that the trained model may be extended to the domain of interest. So, the purpose of domain adaptation is to train a neural network on one dataset for which a label or annotation is accessible and then ensure acceptable performance on a different dataset for which a label or annotation is not available.

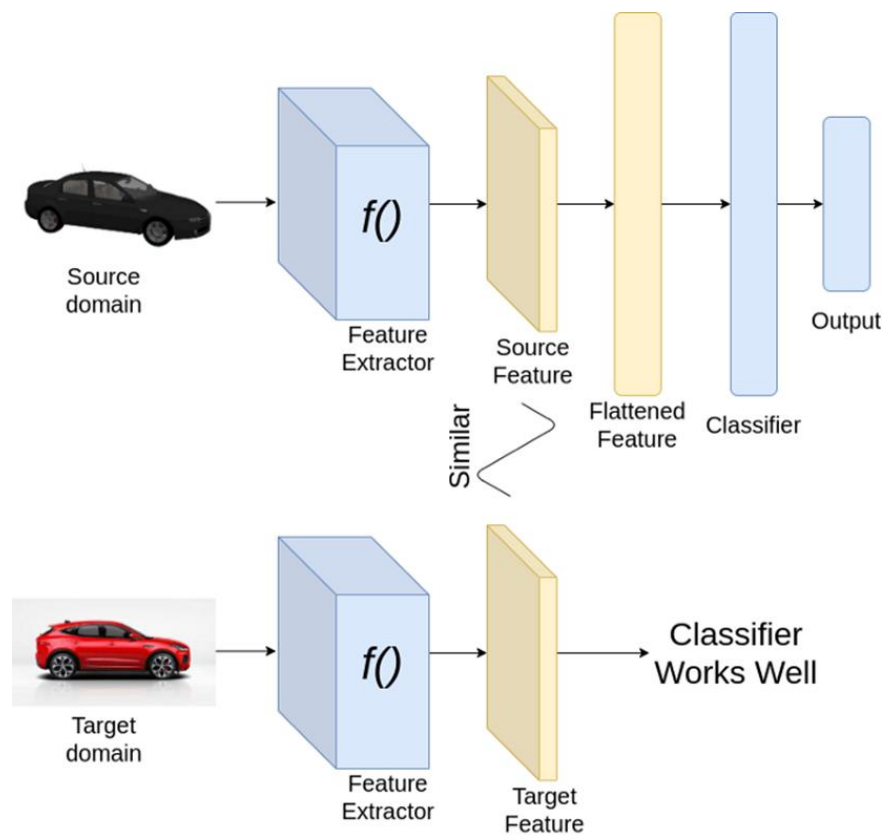


Figure 7: A working domain adaptation [13].

Domain adaptation may be categorized into the following categories based on the type of data available from the target domain:

Supervised: Data from the target domain has been labeled, and the target dataset is substantially smaller than the source dataset.

Semi-Supervised: Data from the target domain that is both labeled and unlabeled.

Unsupervised: A large number of unlabeled target domain sample points.

3.17 Hyperparameter tuning

The process of selecting a set of ideal hyperparameters for a learning algorithm is known as hyperparameter tuning. A hyperparameter is a model argument whose value is determined prior to the start of the learning process. The data that controls the training process is stored in hyperparameters. They're frequently employed in procedures to aid in the estimation of model parameters [14]. The practitioner usually specifies them.

Heuristics are frequently used to set them. They're usually fine-tuned for a specific predictive modeling challenge. Model architecture options are sampled from a range of potential hyperparameter values using hyperparameter tuning algorithms. The process is known as "searching" the hyperparameter space for the best values. Figure 8 shows a basic block diagram for hyperparameter tuning process.

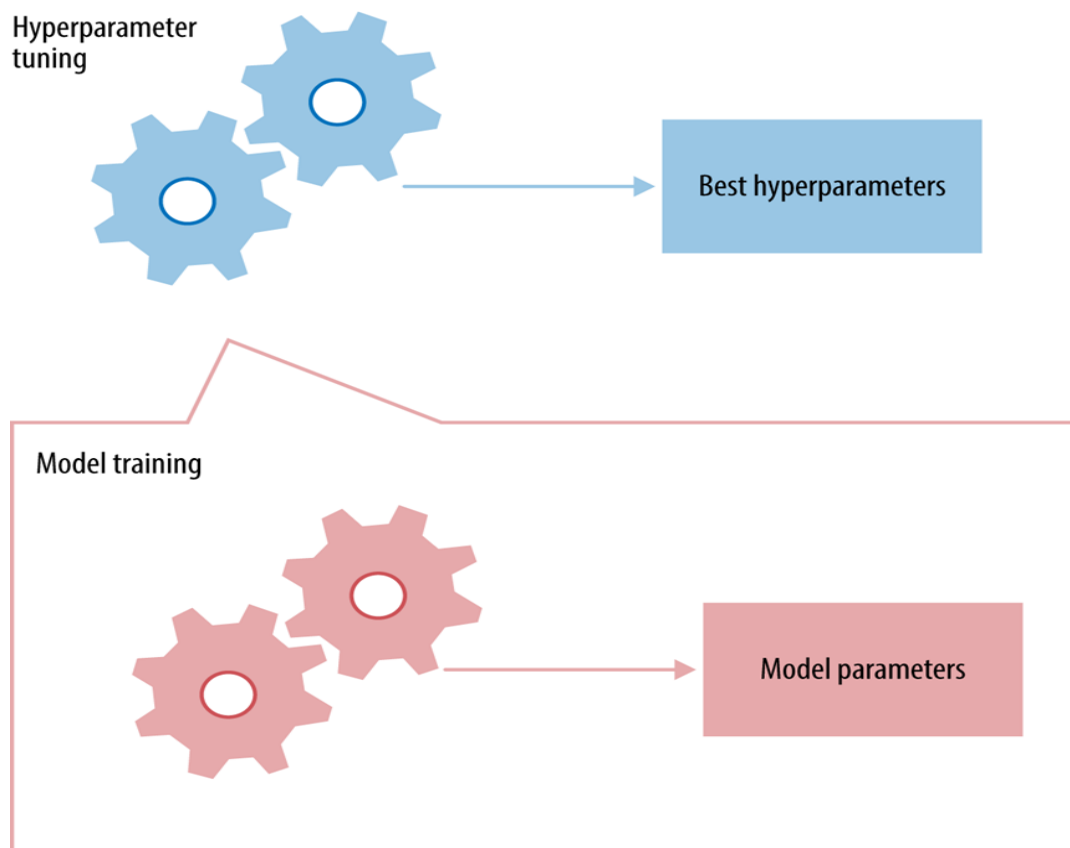


Figure 8: Hyperparameter tuning vs model training [14].

Chapter 4:

Literature Survey

4.1 Related works

The process of splitting an input audio stream into homogenous segments based on the speaker identification is known as speaker diarisation or diarization. It can improve the readability of an automatic speech transcription by breaking down the audio stream into speaker turns and, when combined with speaker identification systems, revealing the speaker's genuine identity.

It's used to figure out "who said what when?" Speaker diarization is the result of combining speaker segmentation and clustering. Speaker segmentation seeks to locate speaker transitions in an audio stream. Clustering seeks to group speech segments together based on the qualities of the speaker.

With the growing number of broadcasts, meeting recordings, and voice mail collected each year, speaker diarization has gotten a lot of attention from the speech community, as evidenced by the National Institute of Standards and Technology's (NIST) specific evaluations [15] for telephone speech, broadcast news, and meetings.

One of the most common ways for speaker diarization is to use a Gaussian mixture model (GMM) to model each of the speakers and then assign the associated frames for each speaker using a Hidden Markov Model (HMM).

Clustering scenarios can be divided into two categories [16]. The first, named Bottom-Up, is by far the most popular. The method begins by separating the entire audio stream into a series of clusters, then gradually tries to combine the redundant clusters until each cluster belongs to a real speaker. The second clustering approach, known as top-down clustering, begins with a single cluster for all audio data and works its way down to a number of clusters equal to the number of speakers.

As seen in Figure 9, traditional speaker diarization systems are made up of many, independent sub-modules. Various front-end processing techniques, such as speech augmentation, dereverberation, speech separation, or target speaker extraction, are used to mitigate any artifacts in auditory environments.

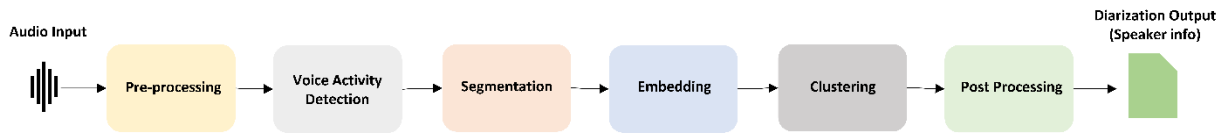


Figure. 9: Traditional Speaker Diarization System

After then, voice or speech activity detection (VAD or SAD) is used to distinguish speech from non-speech events. In the selected speech segment, raw speech signals are processed into acoustic characteristics or embedding vectors. The altered speech segments are classified and labeled by speaker classes during the clustering stage, and the clustering findings are enhanced further during the post-processing stage. In general, each of these sub-modules is optimized separately.

Using the above mentioned traditional diarization system, the researches benefitted automatic speech recognition (ASR) on air traffic control dialogues and broadcast news recordings, by separating each speaker’s speech segments and enabling speaker adaptive training of acoustic models [17, 18, 19, 20, 21, 22, 23]. During this time, the generalized likelihood ratio (GLR) [17] and the Bayesian information criterion (BIC) [24] were established and quickly became the gold standard for quantifying the distance between speech segments for speaker change detection and grouping.

All of these efforts combined laid out paths to consolidate activities across research groups worldwide, leading to several research consortia and challenges in the early 2000s, including the European Commission-supported Augmented Multiparty Interaction (AMI) Consortium [25] and the National Institute of Standards and Technology-hosted RT Evaluation [15]. (NIST).

Ning et al [26] proposed some domain-specific solutions to the open issues of the algorithm like choice of metric; selection of scaling parameter; estimation of the number of clusters. They proposed a postprocessing step – “Cross EM refinement” – is conducted to further improve the performance of spectral clustering. Thus, they solved the problems

such as (1) Cost and complexity (2) cost depending only on the number but not the length of the segments. So, they reduced the number of segmentations by 66%; (3) solves the problem of short speaker turns. The Dataset that they used was audio records of Japanese Parliament Panel Discussions. It contained 9 audio files with each length 20-45 minutes. They used a different Spectral Clustering: Ng-Jordan-Weiss (NJW) algorithm with GMM base model and Evaluation matrix: “diarization error” defined by the NIST Rich Transcription Evaluation. The average DER and purity are 11.25% and 89.14% respectively for our system with the spectral approach. Compared with the average performances of 10.98% and 89.67% for the baseline system, it can be seen that the two systems give similar results.

Tranter et al [27] provide an overview of the same speaker diarization approaches and discuss performance and potential applications. It discusses some more clustering methods that have been used by different researchers such as hierarchical, agglomerative clustering with a BIC-based stopping criterion; Joint Segmentation and Clustering: Viterbi decoder; Cluster Recombination: UBM; Re-segmentation: Viterbi decode with base models GMM, HMM, and EHMM. Dataset that they used is the RT-04F speaker diarization data. These were derived from TV shows: three from ABC, three from CNN, two from CNBC, two from PBS, one from CSPAN, and one from WBN. The dataset had 1 file that is 30 minutes long. The Evaluation matrix DER shows agglomerative clustering BIC-based scheme got around 17%–18% DER; Viterbi re-segmentation between each step providing a slight benefit to 16.4%; Further improvements to around 13% were made using CLR cluster recombination; The CLR cluster recombination stage which included feature warping produced a further reduction to around 8.5%–9.5%; using the whole of the RT-04F evaluation data in the UBM build of the CLR cluster recombination stage gave a final performance of 6.9%; EHMM gave 6.1% DER.

Castaldo et al [28] present a stream-based approach for unsupervised multi-speaker conversational speech segmentation and to find a low dimensional vector of speaker factors that summarize the salient speaker characteristics. They evaluated diarization problems such as (1) cost that depends only on the number but not the length of the segments. (2) require low complexity. (3) a good distance measure between models for calculating the number of speakers in the audio on multilingual Callhouse database, and

from the Italian, Swedish, and Brazilian Portuguese SpeechDat corpora. BIC criterion; Cross Likelihood Ratio clustering method was used with GMM and UBM as base models. Results were evaluated with Detection Error Tradeoff (DET); Equal Error Rate (EER) matrix. The result shows a reduction in the error rates of 18%.

Sun et al [29] describe an improved speaker diarization system based on the previous system for the MDM task of the 2007 NIST Rich Transcription (RT07) Meeting Recognition Evaluation MDM. a high-order GMM clustering method is proposed. BIC criterion is applied for cluster merging. They implemented a high-order GMM modeling approach for segment purification (EM Algorithm) and a cluster merging via BIC Criteria on NIST RT05 and RT06 Dataset with GMM as the base model. The system had three major modules: data preparation, initial speaker clustering, and cluster purification/merging. The data preparation module consists of the raw data Wiener filtering and beamforming for enhanced speech signals, time difference of Arrival (TDOA) estimation, and speech activity detection (SDA). The evaluation matrix DER shows the SAD module produced significant improvements. It reduces an absolute reduction of 11.2% in DER.

Valente et al [30] make use of the speaker roles n-gram model to capture the conversation patterns probability and investigate its use as prior information into a state-of-the-art diarization system. They used the AMI Corpus which has 17 audio files. The proposed technique reduces the diarization speaker error by 19% when the roles are known and by 17% when they are estimated. Furthermore, the paper investigates how the n-gram models generalize to different settings like those from the Rich Transcription campaigns. Experiments on 17 meetings reveal that the speaker error can be reduced by 12% also in this case thus the n-gram can generalize across corpora. Different approaches have been taken so far for producing better result to solve diarization problems. Both supervised and unsupervised methods been tried upon noisy and clean data to classify speaker identity, utterances and overlapping.

Wang et al. [31] in 2018 proposed a LSTM based system with different non-parametric clustering algorithms like – naive, k-means, links and spectral. The LSTM network

consist of 3-layers. The final layer is a linear one. Each LSTM layer has 768 nodes with a projection of 256 nodes. The baseline LSTM here is an i-vector model.

Another GMM UBM model was also been trained using CALLHOME American English (LDC97S42 +LDC97T14); 2003 NIST Rich Transcription (LDC2007S10); the English conversational telephone speech (CTS) part; 2000 NIST Speaker Recognition Evaluation (LDC2001S97), Disk-8 dataset which is basically a d-vector model created based on the baseline d-vector model. It only has two full covariance Gaussians, one for speech, and one for non-speech. As a part of data pre/post processing they excluded overlapped speech from evaluation. For offline clustering algorithms intentionally constrained the system to produce at least 2 speakers. Spectral clustering provides better result which is almost a 90% accuracy in number.

Another d-vector or speaker discriminative model has been created using UIS-RNN. This model by Zhang et al. [32] in 2019 produced better results than the systems that are using spectral and offline clustering's with a DER of 7.9% which is an accuracy near 92%. The model was trained on the NIST SRE 2000 CALLHOME dataset. The system uses a VAD with only two full-covariance gaussians to remove non-speech parts, and partition the utterance into nonoverlapping segments with max length of 400ms. Commonly used clustering modules are replaced by a trainable unbounded interleaved-state RNN here. Bayesian non-parametric – ddCRP (distance-dependent Chinese Restaurant Process) clustering has been used for identification process.

The work by Sharma et al. [33] (2020) describes a system with baseline RNN of 3 layers each with 150 LSTM cells and the output only has one neuron with a sigmoid activation function to predict 0 or 1, since the embedding technique used here is binary vector. The model is trained using 37 custom made audio of 15 minutes using different channel sampling -44100. The system produces a result around 90%. A x-vector based system proposed by Nauman et al. [34] (2020) which is a TPIB based speaker diarization system. This model shown competitive results for meeting datasets with noise. Proposed system consists of two different model which are, VarTPIB-NN and VarTPIB-LDA. The VarTPIB-NN showed an improvement of 3.6% and 2.6% on RT-04Eval and AMI-2 datasets,

respectively. The VarTPIB-LDA showed an improvement of 3.9% on RT-05Eval dataset which are significantly better than the results produced by previous models.

Speaker specific representation in a total variability space derived from simplified Joint Factor Analysis (JFA) [35], known as i-vector [36], was quickly adopted by speaker diarization systems as feature representation for short speech segments segmented in an unsupervised fashion and found great success in speaker diarization. When combined with principal component analysis (PCA) [37, 38], variational Bayesian Gaussian mixture model (VB-GMM) [39], mean shift [40], and probabilistic linear discriminant analysis (PLDA) [41], i-Vector successfully replaced predecessors such as merely mel-frequency cepstral coefficient (MFCC) or speaker factors (or eigenvoices) [28] to boost clustering performance in speaker diarization.

The options to choose a baseline architecture are the old i-vector, relatively old d-vector, and the state-of-the-art x-vector architecture. The i-vector architecture is quite old, though we see methods [42] formulated for the i-vector architecture used effectively on newer architecture. But the baseline i-vector architecture is quite ineffective today, also the computational method for the i-vector is very complex. The d-vector architecture is promising, showing good results and having room for improvements. But d-vector being a Google built system documentation, and information is scarce. Pretrained models, or implementation are hard to come by. However, on the state-of-the-art x-vector architecture, the pre-trained models are available on Kaldi [43], an ASR toolkit.

Chapter 5:

Dataset

We used two different datasets here. One is the AMI dataset [25] and the other one is a manually created Bangla dataset. We ran our experiment of CNN-architecture and LSTM model on AMI dataset with the embedding developed by Wan et. al. [44]. We have taken our experiment further with the triplet loss function as an embedding module on Bangla dataset on BiLSTM model with three of its variants.

5.1 AMI dataset

The AMI meeting corpus [25] is a multi-modal data set. It consists of a number of meeting recordings in both clear and noisy environments. We used a few specific subsets of the recordings and also used the acoustic features for further training purposes. Each subset has 1 to 4 speakers labeled as A, B, C, and D. Each word spoken by the different speaker is also stored as an XML file. The file contains start time, end time, and speaker id. We used three specific subsets of the AMI dataset here – ES, IS, and TS. The XML file is used to segment the input audio. A variety of signals are used in the recordings, all of which are synced to a similar chronology. Close-talking and far-field microphones, individual and room-view video cameras, and slide projector and electronic whiteboard output are among them. The discussions here were taped in English in three distinct rooms with varying acoustic features, with the majority of the participants being non-native English speakers.

5.2 Bangla dataset

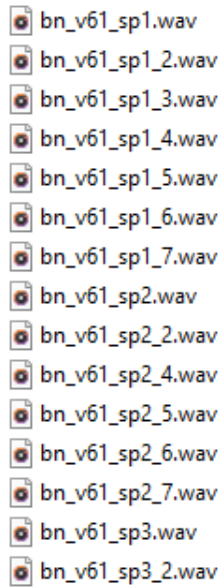
After evaluating the standard procedure, we have moved on to create our own dataset. We have used audio files of TV talk shows to prepare our Bangla dataset. In our dataset, we have around one hundred videos from different talk shows in Bangla. Figure 9 shows the collection of our data. All of these files have been collected from YouTube. Each of these videos is approximately forty minutes long. We used audacity audio processing software to segment those audios. Figure 10 shows manually segmented audios for a single input audio file that has been done using audacity. we have manually labeled the duration of the speakers from the audio files and stored them in our dataset. The same procedure has been used for all the hundred audio files and in this way, we constructed our dataset. These data are now ready for pre-processing and further training. Each input audio file

contains an average of 4 to 5 speakers. All of these input audios were mono channel wav files with 16-bit sample width. We annotated 75 hours of audio data with start time, duration, and speaker id. A sample of our dataset is given below-

	A	B	C	D	E	F	G	H
1	title	url	new_title	video_source	audio_format	number_of_speakers	total_duration	labeled_audio
2	ইভাণি, আলো	e.com/watch	bn_v1	দ মুহিউদ্দীন উ	.wav	3	0:52:00	https://drive.google.com/drive/folders/1vQGP50ZamqLLdEq8SWk061z63e8yzca?usp=sharing
3	ক্লাব, মদ, জুয়া	e.com/watch	bn_v2	দ মুহিউদ্দীন উ	.wav	2	0:58:17	https://drive.google.com/drive/folders/126N-6lnXb6vEeSxlrJrL_i3h9KdLzD_X?usp=sharing
4	শিক্ষাব্যবস্থা ও	e.com/watch	bn_v3	দ মুহিউদ্দীন উ	.wav	2	1:00:18	https://drive.google.com/drive/folders/11_oKyaN1PBaCYANDz7q6YM0TMzOccXH?usp=sharing
5	আগামী নির্বাচন	e.com/watch	bn_v4	দ মুহিউদ্দীন উ	.wav	3	0:52:23	https://drive.google.com/drive/folders/1-HzwYcy95pYT8sfyuW9_BHv3ANgdsDO?usp=sharing
6	ইসরায়েল ও প্য	e.com/watch	bn_v5	দ মুহিউদ্দীন উ	.wav	3	0:50:22	https://drive.google.com/drive/folders/1f8kphGMgaTjEX3EBe7Bw_K2spFqo8FEW?usp=sharing
7	রোজিনা ও রাফ	e.com/watch	bn_v6	দ মুহিউদ্দীন উ	.wav	3	0:53:57	https://drive.google.com/drive/folders/1U-kV9Chyv1Av2Xiu1OFJK6LNIQbuuy3?usp=sharing
8	সব মানুষের ঈ	e.com/watch	bn_v7	দ মুহিউদ্দীন উ	.wav	3	0:46:25	https://drive.google.com/drive/folders/1lITU-TpmbUH9ZRWtXSLtab_o-LerPWT?usp=sharing
9	করোনা টিকা	e.com/watch	bn_v8	দ মুহিউদ্দীন উ	.wav	3	0:46:14	https://drive.google.com/drive/folders/119S9LauS088QI8B8DYUALsPx9ZMVZ7JC?usp=sharing
10	গুলশানে তরুণ	e.com/watch	bn_v9	দ মুহিউদ্দীন উ	.wav	3	0:50:48	https://drive.google.com/drive/folders/1Jl6yztCJdPqs-Wp62ofJggufHBFrf2ah?usp=sharing
11	পশ্চিমবঙ্গের নি	e.com/watch	bn_v10	দ মুহিউদ্দীন উ	.wav	3	0:54:52	https://drive.google.com/drive/folders/1X4DfJrZXzjopMRVLG0D0aReA4opOD1T?usp=sharing
12	মামলা, দমন, প	e.com/watch	bn_v11	দ মুহিউদ্দীন উ	.wav	3	0:58:38	https://drive.google.com/drive/folders/1_eY3AUR0v_v3UkvcOb_rNO487zUjDZOB?usp=sharing
13	লকডাউন: বাং	e.com/watch	bn_v12	দ মুহিউদ্দীন উ	.wav	3	0:50:38	https://drive.google.com/drive/folders/1CMq3fsnX7tc-X5qe2ONW08H4GQvCGUJ0?usp=sharing
14	বাংলাদেশে বি	e.com/watch	bn_v13	দ মুহিউদ্দীন উ	.wav	3	1:01:41	https://drive.google.com/drive/folders/1xACTapTkglySlnfzUvOpnKWc24qJXn?usp=sharing
15	স্বাধীনতার ৫০	e.com/watch	bn_v14	দ মুহিউদ্দীন উ	.wav	3	0:55:07	https://drive.google.com/drive/folders/1_85CqatHGk8pkL6hrM45jvUwQMDXIVB?usp=sharing
16	মোদীর সফর ও	e.com/watch	bn_v15	দ মুহিউদ্দীন উ	.wav	3	0:57:24	https://drive.google.com/drive/folders/1i7lg9zy8ZnmhgCmGoLLixTixAm2lb6?usp=sharing
17	প্রধানমন্ত্রীর চ	e.com/watch	bn_v16	দ মুহিউদ্দীন উ	.wav	3	0:50:55	https://drive.google.com/drive/folders/1ImxoKxHyMjzSmGkG2H02K7zaR3Kq?usp=sharing
18	মুশতারফের মু	e.com/watch	bn_v17	দ মুহিউদ্দীন উ	.wav	3	0:55:41	https://drive.google.com/drive/folders/13wHeuWYmNnVzeMit_Rwe7L2V0yZ0WP8D?usp=sharing
19	মৃত্যুর ডিজিটাল	e.com/watch	bn_v18	দ মুহিউদ্দীন উ	.wav	3	0:57:49	https://drive.google.com/drive/folders/1oYVC8sr2pEKcZJcXhyPCNV_xxy9SPWHe?usp=sharing
20	মুক্তিযুদ্ধের খে	e.com/watch	bn_v19	দ মুহিউদ্দীন উ	.wav	3	0:50:58	https://drive.google.com/drive/folders/11W6zFxXAhRDFBFinZr6pPqulM3FCuqesO?usp=sharing
21	আল জাজিরার	e.com/watch	bn_v20	দ মুহিউদ্দীন উ	.wav	3	0:58:25	https://drive.google.com/drive/folders/1WkmzJpEXz3Jl5csUJqlm9cxU6mF_lczJ?usp=sharing
22	টিকা কি রাজনী	e.com/watch	bn_v21	দ মুহিউদ্দীন উ	.wav	3	0:55:24	https://drive.google.com/drive/folders/1WmbTaSXSmJ0uTL4C-SryZxb09QJ2SS4?usp=sharing
23	ক্ষমতালালীদের	e.com/watch	bn_v22	দ মুহিউদ্দীন উ	.wav	3	0:50:44	https://drive.google.com/drive/folders/14mlhb1FyX4b0Jh4f9C40lBvKqlet_pb9?usp=sharing
24	বাংলাদেশ-ভার	e.com/watch	bn_v23	দ মুহিউদ্দীন উ	.wav	3	0:44:35	https://drive.google.com/drive/folders/1pc235ZVGMymrkNLCX5EoiV-cueUpeH?usp=sharing
25	ট্রাম্পের হার, বা	e.com/watch	bn_v24	দ মুহিউদ্দীন উ	.wav	3	0:48:49	https://drive.google.com/drive/folders/1znigr2l_xvMe-CoGOgaGniWEE3WT7K03?usp=sharing
26	বাংলাদেশের ও	e.com/watch	bn_v25	দ মুহিউদ্দীন উ	.wav	3	0:55:50	https://drive.google.com/drive/folders/1nrEN4QSHBcfwnLU8-EgXdyJhUhlYXhH?usp=sharing

Figure 10: Collection of Audios for Bangla dataset with relevant annotated information.

Here, we have kept the records of our data. Then after labeling, all the data will be stored in the drive link for saving computer internal storage and easy access by everyone. The labelled data look like this-



bn_v61_sp1.wav
bn_v61_sp1_2.wav
bn_v61_sp1_3.wav
bn_v61_sp1_4.wav
bn_v61_sp1_5.wav
bn_v61_sp1_6.wav
bn_v61_sp1_7.wav
bn_v61_sp2.wav
bn_v61_sp2_2.wav
bn_v61_sp2_4.wav
bn_v61_sp2_5.wav
bn_v61_sp2_6.wav
bn_v61_sp2_7.wav
bn_v61_sp3.wav
bn_v61_sp3_2.wav

Figure 11: Speaker utterances for a single Bangla meeting audio

The same procedure has been used for all the hundred audio files and in this way, we will construct our dataset and upload it in google drive for use. These data are now ready for pre-processing, noise reduction etc.

Chapter 6:

Methodology and Model Architecture

6.1 Proposed method

Wan et al. created a speaker embedding using an LSTM network [45] for both text-dependent and text-independent speech verification [8]. As a starting point, we employ the Embedding module for our Speaker Diarization system a pre-trained instance of their model (trained on fixed-length segments collected from a big corpus).

Then we try out a CNN architecture, an LSTM Model (on the AMI-Corpus Dataset, henceforth referred to as AMI-CNN and AMI-LSTM), a BiLSTM Model (on the Bangla Dataset, henceforth referred to as Bangla-BiLSTM), and three different variants of a BiLSTM model (using Transfer Learning, henceforth referred to as BiLSTM-TF) with the Triplet Loss Function as an embedding module.

The input audio signals are segmented into uniform small chunks using ground truth labels (each chunk containing approximately only one speaker), a VAD is used to select only those segments that contain speech, and raw level features (MFCC and log-Mel Spectrum Features) are extracted from each frame as the network input for the embedding model (BiLSTM-TF/LSTM/CNN).

For each input sequence, the embedding model generates speaker embeddings. As a result, arbitrary-length audio streams are reduced to a series of fixed-size embeddings. We then use a clustering approach to calculate both the overall number of unique speakers in the audio input as well as assign each audio chunk to a specific speaker using these embeddings.

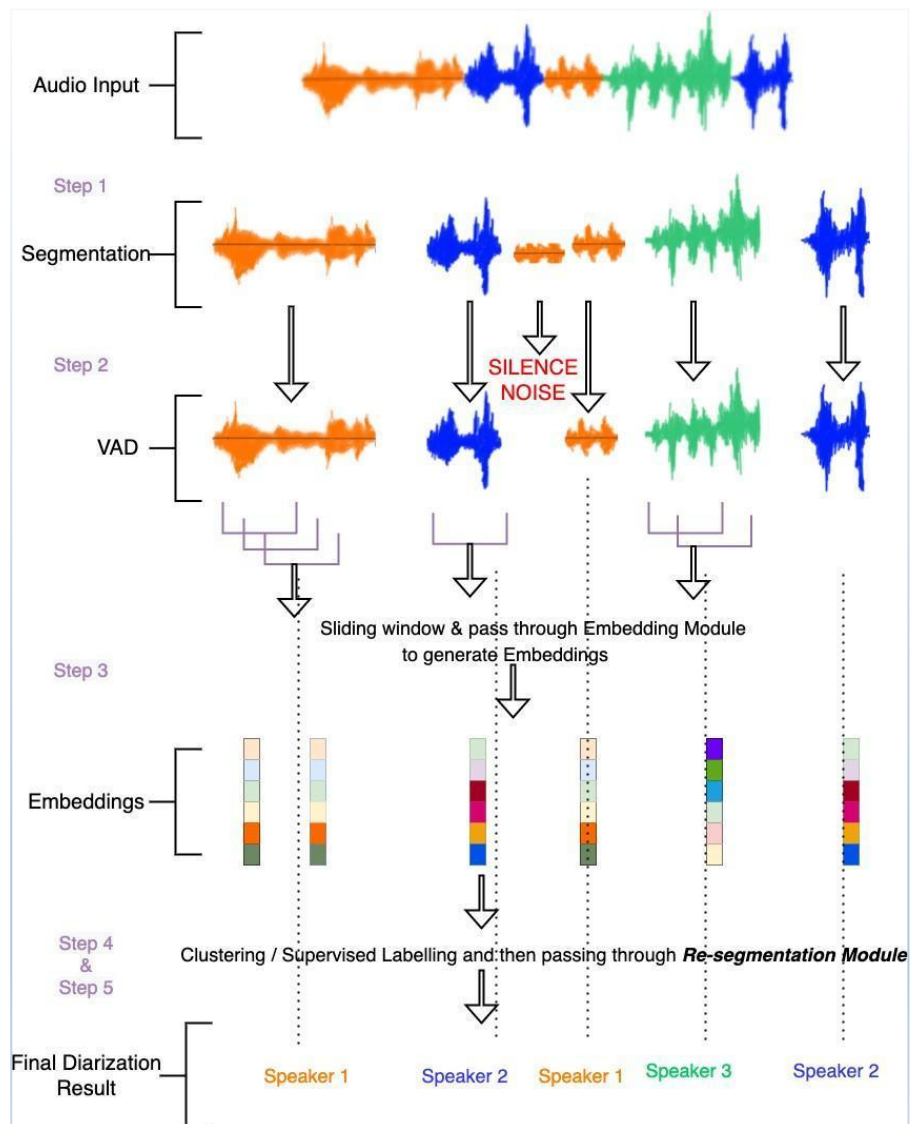


Figure 12: Proposed Approach for Diarization.

6.2 Pre-processing

The audio wav files have several speakers, each with a corresponding text annotation file that lists the timestamps of each speaker as well as their speaker id. The Librosa library is used to load it into the system. We next parse the annotation files that we prepared using audacity software for the appropriate time stamp, break the wav file into small chunks, then concatenate them according to the time stamp to their id as a speaker. If the consecutive chunks do not belong to the annotation file, we additionally eliminate the noise throughout this step.

Librosa [46] is a Python tool that analyzes music and audio. Librosa is mostly utilized when working with audio data, such as in the creation of music (using LSTMs) and automatic speech recognition. It provides the necessary building elements for the development of music information retrieval systems.

We separated the audio into 1 second long chunks for the testing step because we didn't know which speaker said what at what time (i.e. whether parts of the wav file actually include utterances). We then conducted the denoising low-level feature extraction as before. For each of the models, we employ the following routines:

6.2.1 AMI-CNN and AMI-LSTM

We use a self-implemented Mel-log spectrum and MFCC feature extractor using the `python_speech_feature` library as well as a denoiser to remove the silence parts and speech noise for the AMI-CNN model. The log magnitude spectrum on the nonlinear frequency scale, which is similar to the mel scale, can well depict the spectral envelope of speech (called the mel-log spectrum). The mel spectrum, as described by its Fourier coefficients, is also seen to have a useful property for representing the spectral envelope as a parameter. The mel scale is a scale of pitches judged by listeners to be equal in distance one from another.

Windowing the signal, applying the DFT, taking the log of the magnitude, and then warping the frequencies on a Mel scale, followed by using the inverse DCT is all part of the MFCC feature extraction technique. We used the log-mel spectrum of the wav chunks

as the input vectors (features) to the Embedding module for the AMI-LSTM model, which is essentially the resemblyzer. Figure 13 shows the model architecture of AMI-CNN.

We used the model architecture: input (160 x 64) \rightarrow 2D Convolution (3,3) \rightarrow Relu activation \rightarrow 2D Maxpool (2,2)) $\times 2 \rightarrow$ Dense Layer (32) \rightarrow Relu activation \rightarrow Dense Layer (16). Therefore, the output embedding is a 16 floating values long vector.

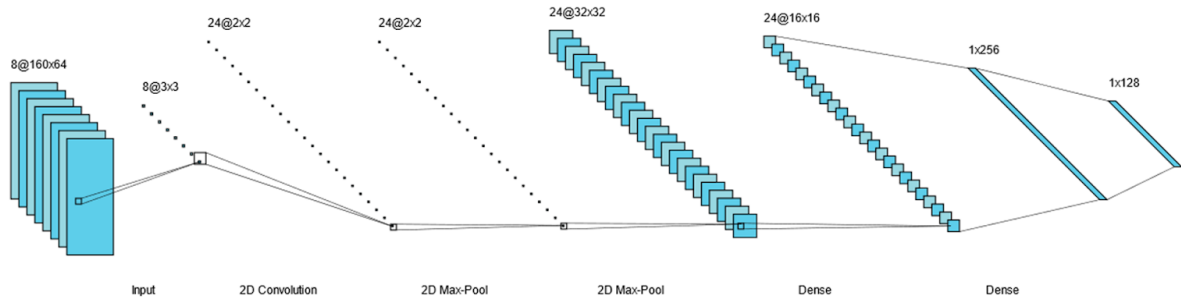


Figure 13: Model Architecture of AMI-CNN.

We use three LSTM Layers (768 nodes large) followed by 2 Dense Layers with tanh activation and Dropout of 0.01. The 256 long output vector is normalized using l2 distance metric. Figure 14 shows the model architecture for AMI-LSTM.

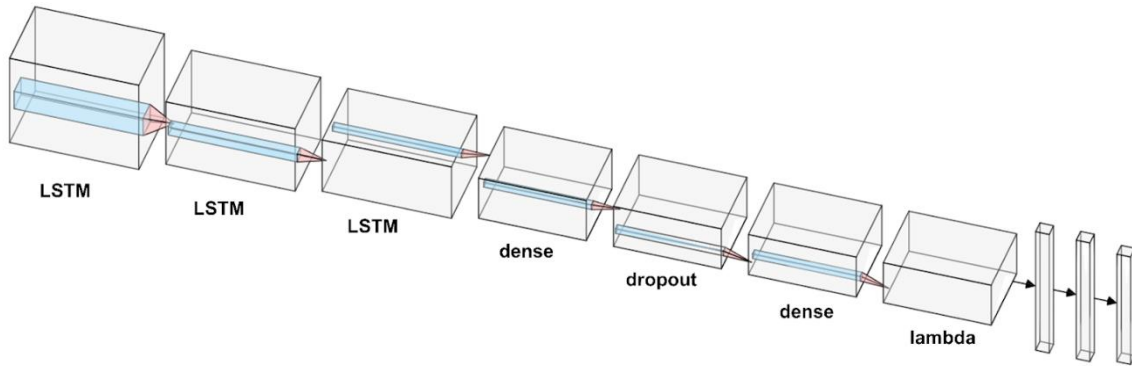


Figure 14: Model Architecture of AMI-LSTM.

6.2.2 Bi-LSTM Transfer learning

We used Librosa's MFCC and MFCC-Delta feature extractors for the AMI-LSTM and all three BiLSTM-TL variants. We use Librosa's MFCC and MFCC-Delta feature extractors, as well as Pyannote's Embedding generation function, for the Bangla-English-BiLSTM model. We use two BiLSTM Layers (128 nodes large) followed by 2 Time Distributed Dense Layers (32 nodes large) with tanh activation and Dropout of 0.01.

6.3 Voice activity detection

6.3.1 Energy-based VAD (log-mel spectrum)

An energy detector is used in the most basic VAD designs. When the signal's energy climbs over a certain threshold above the noise floor, it's considered that the increase is due to speech [49]. Because the noise level in most applications is unknown in advance and changes over time, it must be calculated throughout the conversation. The dual time constant integrator is an adaptive approach for determining the noise floor. To put it another way, if the signal's energy at point N is less than the noise floor, the noise floor will drop fast, and if the energy is more than the noise floor, the noise floor would rise slowly. When the noise floor fluctuates more slowly than the speech envelope, this strategy comes in handy. State machines with variable thresholds can be utilized to further improve the performance of the VAD depending on energy thresholds. Speech, for example, has a significant relationship with time. As a result, prior frame states can be used to predict the likelihood of speech in the present frame.

6.3.2 Neural Network-Based VAD (LSTM)

Output from VAD each speech and non-speech segment is assigned a label of 1/0 and VAD is regarded as a binary classification issue. Recurrent Neural Network or RNNs may capture temporal connections by integrating feedback loops between the neural network's input and output [50]. The Long Short-Term Memory (LSTM) network is a kind of RNN that has gained popularity as a result of its capacity to capture long and short-term dependencies concurrently. Bidirectional LSTMs (BLSTMs) are a type of LSTM that combines two separate LSTM networks that function in the same order: the first does a causal analysis, while the second performs an anti-causal analysis. The

effectiveness of these models in the VAD task has been demonstrated in prior studies. The capacity to simulate long-range relationships between the inputs is the driving force for the deployment of LSTM-RNN. Other typical data-driven VAD techniques, such as those based on GMM or Feed-Forward Neural Networks, ignore temporal relationships. These challenges are addressed using delta features, modulation, or long-span features. Recurrent Neural Networks can only model a small degree of time dependency. LSTM-RNN, on the other hand, goes beyond simply using context information by introducing the concept of a memory cell that can be read, written, and reset based on feature context and previous outputs, using multiplicative input, output, and forget units with multiplicative weights learned automatically during training. They solve the vanishing gradient problem of classic RNNs by learning when to access which bits of the prior context.

6.3.3 GMM Based VAD (WebRTC)

A voice activity detector based on GMMs that employs GMM models of speech and non-speech sounds as input features, with log energies of six frequency bands ranging from 80Hz to 4000Hz. The goal of voice activity detection (VAD) is to recognize human speech in an audio stream that contains both speech and noise. Typical VAD approaches include feature engineering based on power spectral density and periodicity, which is paired with trained statistical models for classification, such as gaussian mixture models (GMM) or artificial neural networks. A VAD can be used in a variety of ways, such as a front-end for automatic speech recognition, to filter out non-speech sounds, or to change coding schemes in codecs like Opus. Because these applications are often real-time, rapid low-latency VADs are required. WebRTC, which employs a GMM architecture, has a common version of a VAD. It is computationally efficient and can run in low-resource environments, although its accuracy isn't the best.

We implemented VAD into our project in three different ways. WebRTC-VAD makes use of GMM models of speech and non-speech sounds, with input characteristics represented as log energies in six frequency bands ranging from 80Hz to 4000Hz. The energy-based method employs normalized energies (calculated using log Mel Spectrum and MFCC Features) and a user-supplied threshold to identify bands of vocal activity. The LSTM-based one is similar to the one used for training on input files except that it labels each

speech/non-speech segment 1/0 and handles VAD as a binary classification issue. Figure 15 shows the wave signal for any input audio. Figure 16, 17 and 18 are some example outputs of audio signals for different VAD's used in this project. Wave signal of different VAD gives a proper visual reference of the audio including speech and non-speech segments

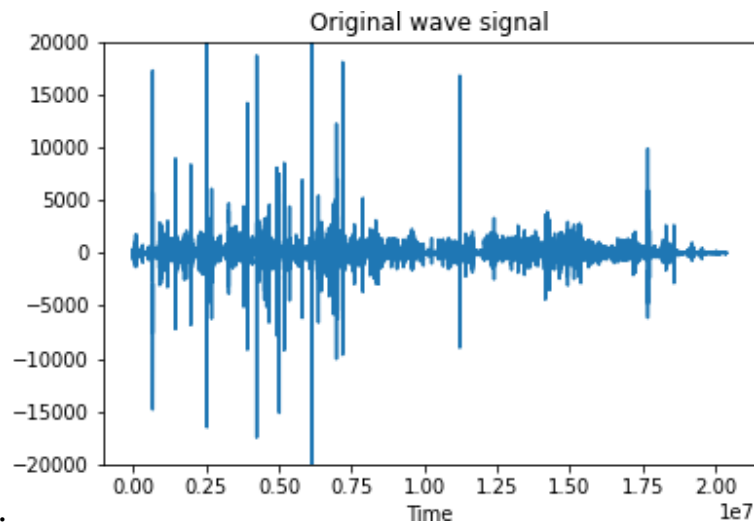


Figure 15: Wave signal for an input audio.

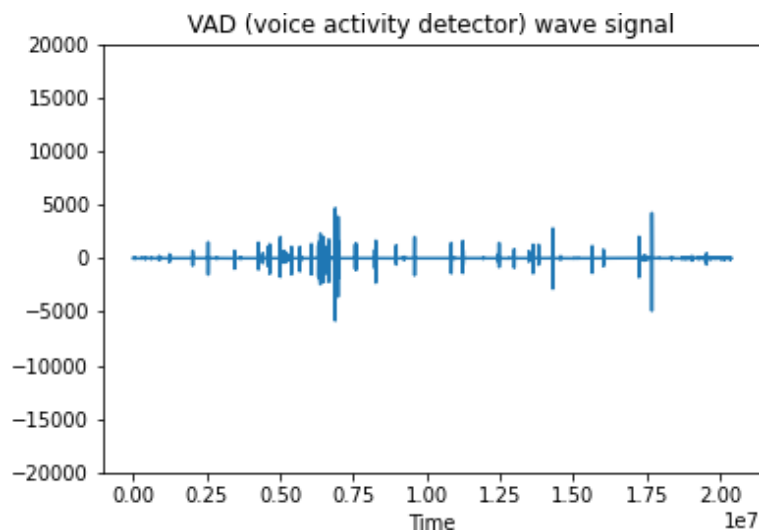


Figure 16: Energy based VAD (log-mel spectrum)

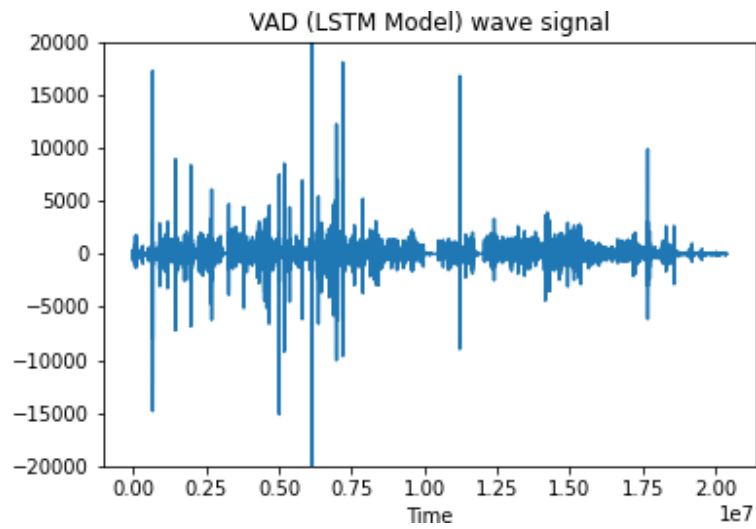


Figure 17: Neural Network-Based VAD (LSTM)

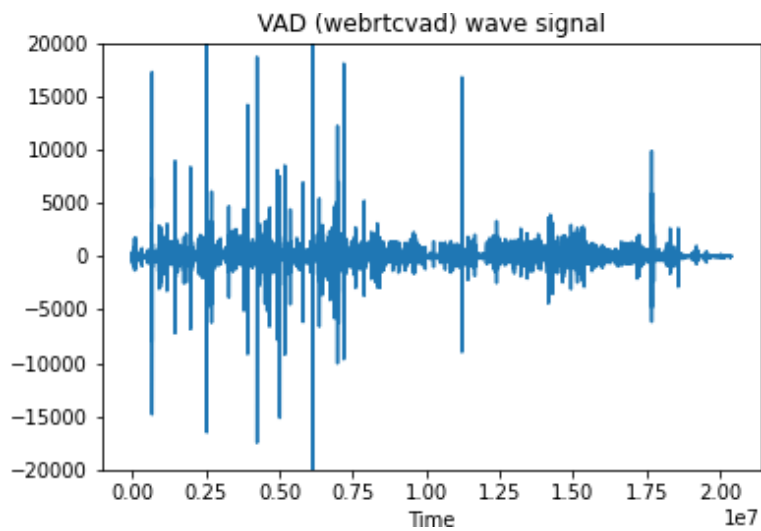


Figure 18: GMM based VAD (WebRTC)

6.4 Embeddings

We used the embedding module (Resemblyzer[47]/AMI-CNN/AMILSTM/Bangla-BiLSTM/BiLSTM-TF) to produce the embedding vector after reducing noise. These embedding vectors are designed to determine the degree of similarity between two speakers based on their distance or cosine similarity.

6.4.1 Resemblyzer

Resemblyzer [47] uses a deep learning model to provide a high-level representation of a voice (referred to as the voice encoder). It builds a summary vector of 256 values (an embedding, typically shortened to "embed" in this repo) from an audio file of speech that describes the features of the voice uttered [51].

Resemblyzer offers a wide range of applications:

6.4.1.1 Voice similarity metric

compare two voices and provide a score based on how similar they sound. By comparing voice profiles with the continuous embedding of a speech segment, you can figure out who is speaking when.

6.4.1.2 High-level feature extraction

the embeddings created can be used as feature vectors in machine learning models. Component analysis of the embeddings can be used to determine accents, tones, prosody, gender, and so on.

6.5 Clustering Algorithms

Clustering scenarios can be divided into two categories [16]. The first, named Bottom-Up, is by far the most popular. The method begins by separating the entire audio stream into a series of clusters, then gradually tries to combine the redundant clusters until each cluster belongs to a real speaker. The second clustering approach, known as top-down clustering, begins with a single cluster for all audio data and works its way down to a number of clusters equal to the number of speakers. We used the following clustering of embeddings using the following algorithms.

6.5.1 Spectral Clustering

Keras

Clustering is applied to a normalized Laplacian projection in this case. In fact, Spectral Clustering is especially beneficial when the individual clusters' structure is substantially non-convex, or when a measure of the cluster's center and spread is insufficient to describe the entire cluster, such as when clusters are nested circles on the 2D plane. This approach may be used to locate normalized graph cuts if the affinity matrix is the graph's adjacency matrix. When calling fit, an affinity matrix is built using either a kernel function with Euclidean distance, such as the Gaussian (aka RBF) kernel, or a k-nearest neighbors connection matrix. When a measure of the cluster's center and spread isn't enough to describe the entire cluster, it's helpful. The exact number of expected speakers is required for this strategy.

Google

Uses KMeans++ on the features in the eigenvector space after performing eigen decomposition which is used to decompose a matrix into eigenvectors. As an input, it accepts the minimum and maximum predicted cluster numbers.

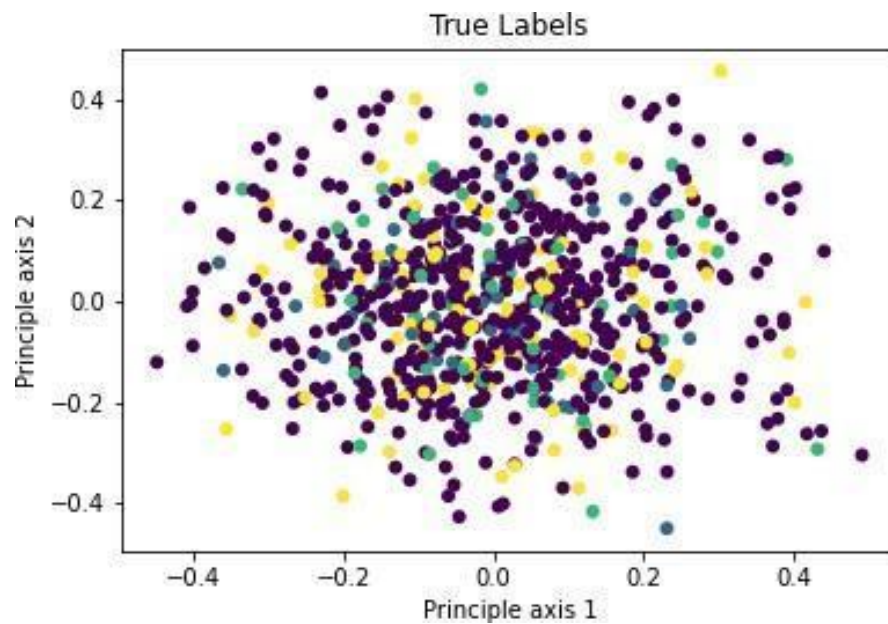
6.5.2 Hierarchical Density-Based Spatial Clustering (HDBSCAN)

Hierarchical Density-Based Spatial Clustering or HDBSCAN is a hierarchical clustering technique that is based on DBSCAN or Density-Based Spatial Clustering. We tuned the other settings manually and set min cluster size to 4 for our model. Because HDBSCAN is just a DBSCAN implementation for variable epsilon values, the minimum cluster size is the only input parameter. This, unlike DBSCAN, allows it to detect clusters of varied densities without first deciding on an acceptable distance threshold.

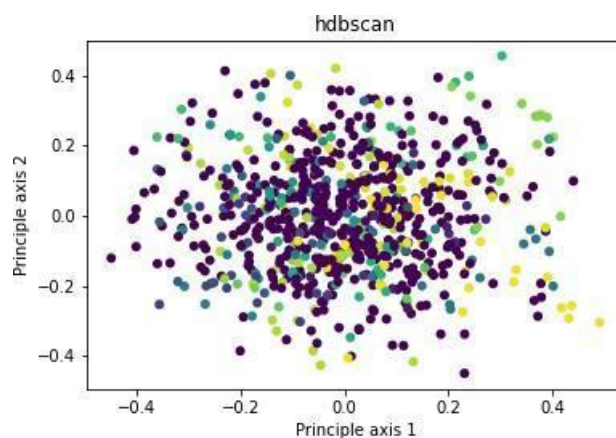
6.5.3 MeanShift

The goal of mean-shift clustering is to locate and adapt centroids based on the density of instances in the feature space. It's extremely similar to K-means clustering, with the exception that it doesn't require a number of clusters to be specified. The algorithm allocates each data point to the nearest cluster centroid iteratively given a series of data points, with the direction to the closest cluster centroid defined by where the majority of

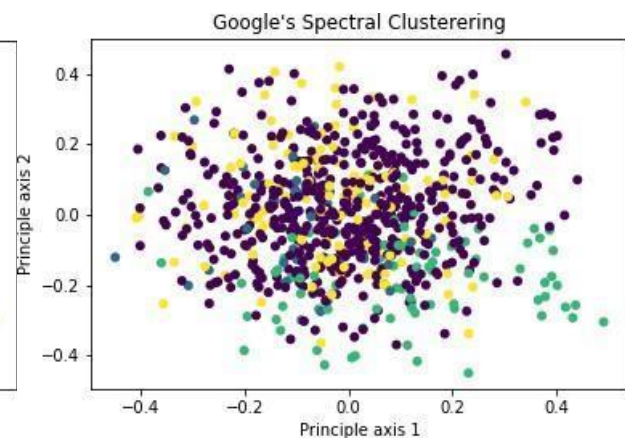
the adjacent points are. As a result, each iteration will bring each data point closer to the cluster center, which is or will be the location of the most points. Each point is assigned to a cluster after the algorithm ends. The benefit of the method is that it distributes clusters to data without automatically specifying the number of clusters depending on bandwidth. As a result, we decided to use this approach to cluster the embeddings. Figure 16 shows different clustering algorithm used in the project for better understanding of the data. All of them are generated based on the Bangla dataset.



(a)



(b)



(c)

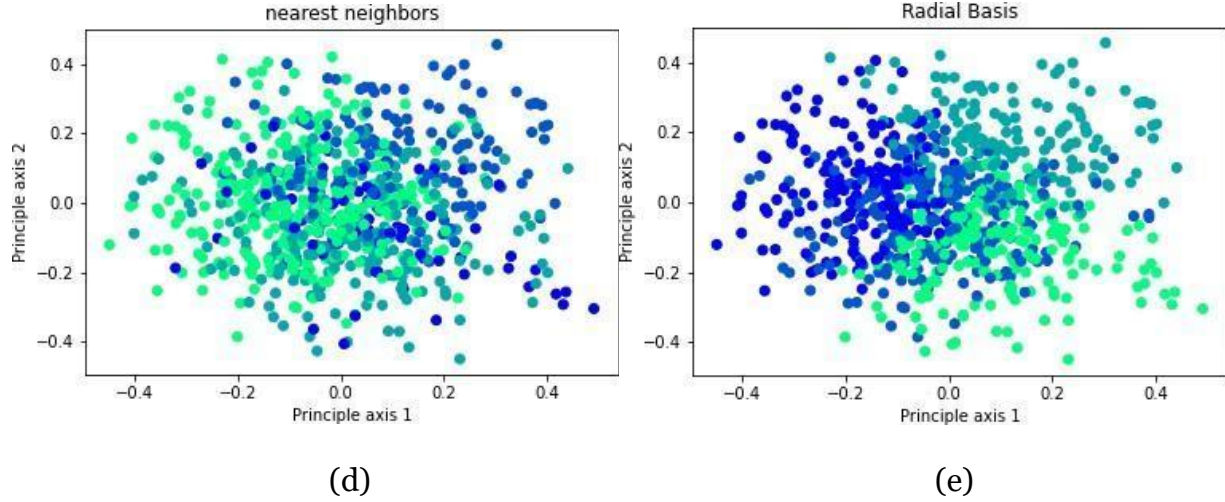


Figure 19: Clustering of embedding vector using different algorithm –
 (a) True Labels (b) Google’s Spectral Clustering (c) HDBSCAN
 (d) Nearest Neighbors (e) Radial Basis

6.6 DER calculation

The Diarization Error Rate (DER), as specified and utilized by NIST in RT evaluations, is the key metric used for speaker diarization tests. It is calculated as the percentage of time that is incorrectly assigned to a speaker or non-speech. We next use Diarization Error Rate to compare it to the given speaker id (ground truth) to get results.

The system hypothesis diarization output does not need to identify the speakers by name or definite ID, hence the ID tags provided to the speakers in both the hypothesis and reference segmentation do not have to be the same. Non-speech tags, on the other hand, are marked as non-labeled spaces between two speaker segments and hence must be explicitly identified.

The evaluation script begins by performing the best one-to-one mapping of all speaker label IDs between the hypothesis and reference files. This permits distinct ID tags to be scored between the two files. The Diarization Error Rate score is computed by

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

where S is the total number of speaker segments where both reference and hypothesis files contain the same speaker/s pair/s. It is obtained by collapsing together the hypothesis and reference speaker turns. The terms $N_{ref}(s)$ and $N_{hyp}(s)$ indicate the number of speakers speaking in segment s , and $N_{correct}(s)$ indicates the number of speakers that speak in segment s and have been correctly matched between reference and hypothesis. Non-speech segments are assumed to have no speakers in them. The error for a segment is 0 when all speakers/non-speech in that segment are appropriately matched. Comparison of Diarization Error Rate (DER) for both train and test data are shown in Table 1. The scores are for the method where we used resemblyzer that uses voice encoders and spectral clustering.

Table 1: DER comparison for Resemblyzer, Spectral combined approach.

Blur σ	Train DER	Test DER
0.25	0.1733	0.6143
0.5	0.1464	0.5575
0.75	0.1825	0.5613
1.00	0.4617	0.5644

6.7 Experimental Setup

Hardware and software details for the project is stated below -

6.7.1 Hardware configuration

- Windows: 10
- Nvidia GPU
- RAM 16 GB

6.7.2 Software configuration

- Python 3.6
- Jupyter notebook

We used the following libraries:

Pydub: Python library to work with only .wav files. By using this library, we can play, split, merge, edit ours .wav audio files.

Xmltodict: Parse the given XML input and convert it into a dictionary.

Resemblyzer: Allows to derive a high-level representation of a voice through a deep learning model (referred to as the voice encoder).

Pyannote: Advanced data structures for handling and visualizing temporal segments with attached labels.

Noisereduce: Reduces noise in .wav files.

Spectralcluster: Allows to implement spectral clustering algorithms.

PyTorch: An open source machine learning framework that accelerates the path from research prototyping to production deployment.

Pyannote.metrics: A toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems.

Pyannote.core: Advanced data structures for handling and visualizing temporal segments with attached labels.

Hdbscan: Hierarchical Density-Based Spatial Clustering of Applications with Noise. hdbscan library is a suite of tools to use unsupervised learning to find clusters, or dense regions, of a dataset.

Keras: Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. In this project we used the latest version of Keras.

Tensorflow_addons: TensorFlow Addons is a repository of community contributions that conform to well-established API patterns. TensorFlow addons supports a large number of operators, layers, metrics, losses, optimizers, and more.

Python_speech_features: Provides common speech features for ASR including MFCCs.

Chapter 7

Transfer Learning Variants

We developed three other variants of the LSTM based model using transfer learning and hyper parameter tuning to examine the feasibility difference of all the models in terms of result. The transfer learning variants were developed based on feature extraction and combining the models.

7.1 Transfer learning

Transfer learning focuses on preserving and transferring information learned while addressing one problem to a different but related one. The idea is that a model trained on a big and general enough dataset would effectively serve as a generic model of the whole domain, and that these pre-learned feature mappings will eliminate the need to train a large model on a large dataset from start. We integrate Transfer Learning by pre-training the BiLSTM model on the Bangla Dataset and then fine-tuning it on the AMI-Corpus Dataset using the variants Feature extraction, Combined models and Hyperparameter tuning.

7.1.1 Feature extraction

Feature Extraction is a technique for reducing the number of features in a dataset by generating new ones from existing ones and then discarding the original features. The original set of features should then be able to summarize the majority of the information in the new reduced set of features. From a combination of the original set, a summarized version of the original features may be generated. Other benefits of employing Feature Extraction methods include improved accuracy, less danger of overfitting, faster training, improved data visualization, and increased explain ability of our model. Because the representations learnt by the pre-trained network would extract meaningful features [48], we run the raw features (MFCC) of the data sample through the Bangla-BiLSTM Model to produce "refined" features. The resulting "refined dataset" is then utilized to train a new Embedding Module (AMI-LSTM) from the ground up. This is akin to running the dataset through a series of two models, one after the other, to align the data.

7.1.2 Combining Models

We've combined the two models which are AMI-LSTM and Bangla Bi-LSTM, into one. The notion is that the pre-trained network's BiLSTM layers already have features that can be used to create embeddings. The dense layers, on the other hand, are unique to the original dataset. As a result, the weights of the BiLSTM layers in the Bangla-BiLSTM Model are frozen, the Time Distributed Dense Layers are removed and replaced with one LSTM + Simple Dense Layers, and the model is retrained using MFCC features from the AMI-Corpus Dataset, allowing only the top layers to be trained.

7.2 Hyperparameters tuning (Knob tuning)

Our method includes a variety of hyperparameter "knobs" that may be adjusted to fine-tune and improve the performance of the Transfer Learning Models stated above. For example, we noticed that the number of speakers recognized by spectral clustering was sensitive to the amount of Sigma used in the Gaussian Blur operation used throughout our research. As can be seen below in Figure 20 and 21, the output embedding clustering varies as Sigma changes. As a result, after training on a single dataset, we may finetune Sigma manually to better adapt our Model to new datasets and get better outcomes.

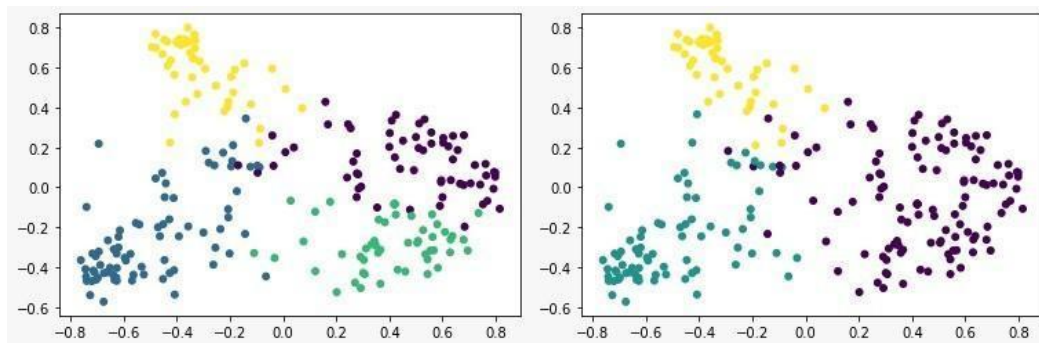
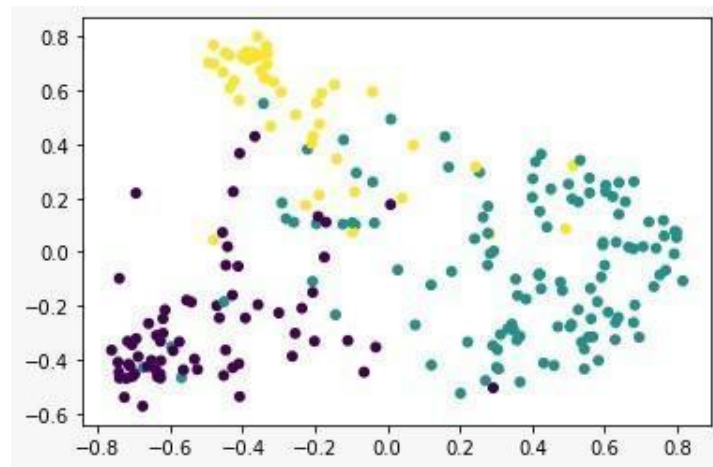


Figure 20: Spectral Plots with their respective Gaussian Blur Sigma Values ($\sigma = 0.25$ and $\sigma = 0.50$)



=

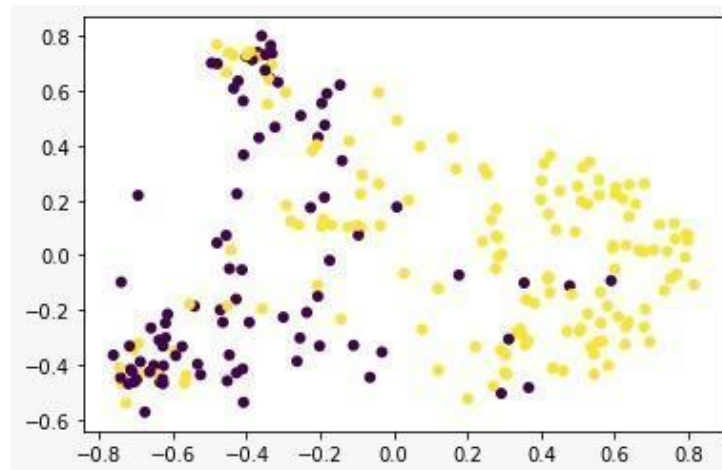


Figure 21: Spectral Plots with their respective Gaussian Blur Sigma Values ($\sigma = 0.75$, $\sigma = 1$ and Ground truth)

Chapter 8

Result analysis

We used Pyannote's metric library to calculate the DER. It's part of the pyannote toolbox, which makes creating diarization pipelines simple. The DER computation is done in Python, and the best speaker mapping is done with `scipy.optimize.linear sum assignment` with a "greedy" assignment option. The DER function can be invoked straight from Python without the requirement to save the results to a file. Other metrics included in the toolbox include cluster purity and coverage.

8.1 Result visualizations

Figure 17 and 18 describes the Plots of Hypothesis & Reference for the BiLSTM model which was ran on the Bangla dataset. Figures 19 and 20 give a simple visual representation of the output diarization, referred to as Hypothesis ("who spoke when"), in comparison to the ground truth, referred to as Reference (who said what when). The findings of the Diarization Error Rate (DER) test are reported in Table 1.

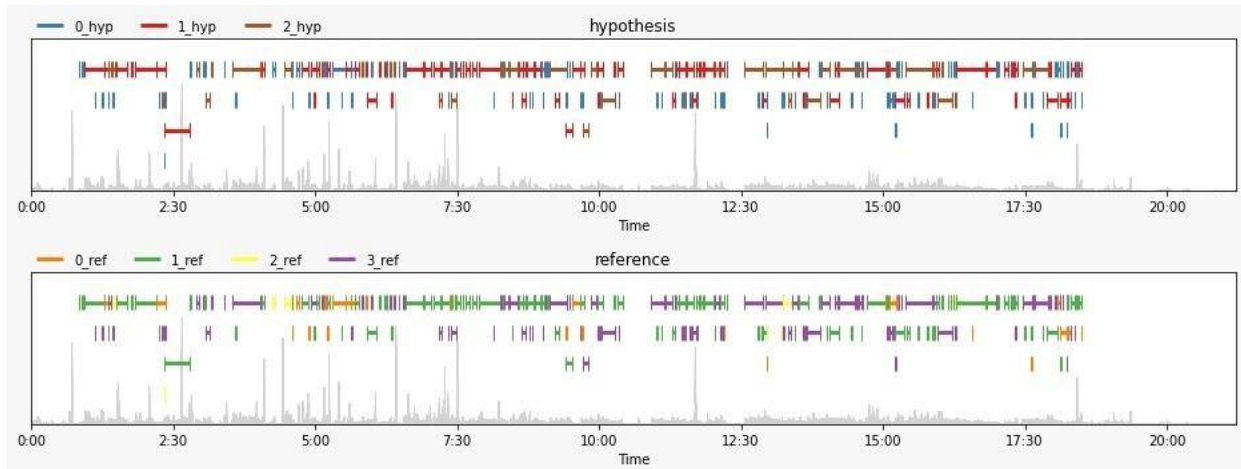


Figure 22: Plots of Hypothesis & Reference for BiLSTM model (Train DER: 0.114)

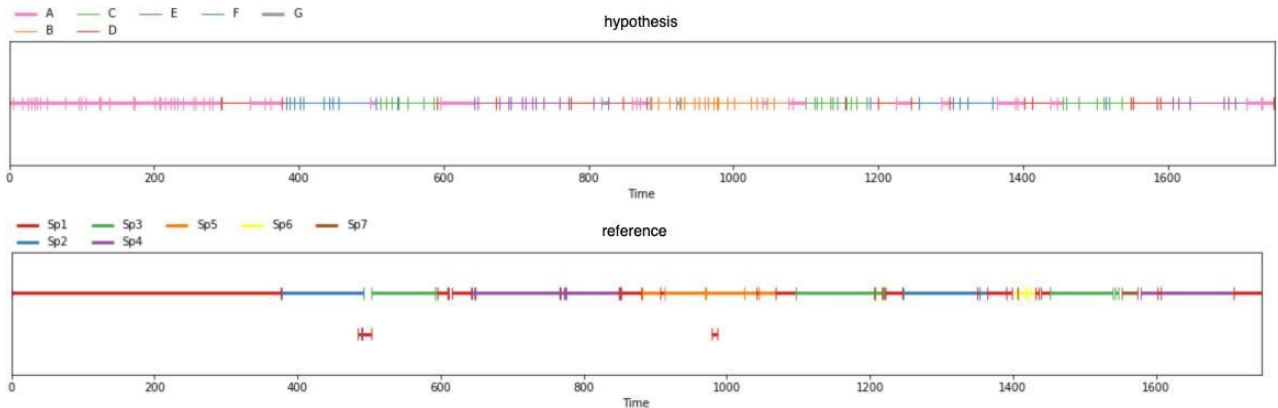


Figure 23: Plots of Hypothesis & Reference for BiLSTM model Test (DER: 0.24)

8.2 Performance comparison

We compared all three models that trained on AMI and Bangla dataset. Among them the Bangla-BiLSTM achieved a test DER of 0.238.

Table 2: Train and Test DER comparison between all three Embedding architectures

Model	Train DER	Test DER
<i>AMI-CNN</i>	0.32	0.37
<i>AMI-LSTM</i>	0.287	0.304
<i>Bangla-BiLSTM</i>	0.114	0.238

Transfer learning variants did not achieve as promising result as the Bangla-BiLSTM model. Among the three variants, variant 3 achieved a test DER score of 0.28.

Table 3: Comparison between the variants

Model	Train DER	Test DER
<i>TL Variant 1</i>	0.41	0.45
<i>TL Variant 2</i>	0.41	0.29
<i>TL Variant 3</i>	0.42	0.28

Chapter 9

Project Summary

9.1 Conclusion

The goal of this work was to build a speaker diarization system for Bengali meeting recordings. Speaker diarization refers to the task of dividing audio with multiple speakers into separate audio by individual speakers. Our goal includes to make a labeled dataset of Bengali meeting recordings, then run experiments to the existing speaker diarization systems for possible improvements. These improvements must be made tailored to our dataset and Bengali speech. And finally, train the improved model with our dataset. Comparison with other systems may be constrained by the dataset as other similar systems are not yet found for Bengali speech.

This research presents the findings from our experiments of a Speaker Diarization with a Transfer Learning model. We show our research on VADs, Pre-processing Algorithms, Embedding Modules, and Clustering Algorithms. Additionally, we demonstrated how Transfer Learning may significantly reduce the computation time required to train an end-to-end Speaker Diarization System at a very minimal cost of accuracy, and in some cases, how it can outperform non-Transfer Learning models.

In this research we first run the AMI-CNN and AMI-LSTM models so that we can use the weights and embeddings later on in our Bangla-Bi-LSTM model. In order to fine tune our model and look for possible improvements we implemented three variants of the Bi-LSTM model on our Bangla dataset using transfer learning. In our first variants we took feature and embeddings extracted from the AMI-LSTM. In the second one we combined the AMI-LSTM model and Bangla-Bi-LSTM model and extracted MFCC features. In our third variant we changed the values of our hyperparameter sigma. Then we calculated DER of all our experiments and compared the results. From our experiments our Bangla Bi-LSTM model obtained a DER score of 0.24.

9.2 Future work

Our future work would be to choose a baseline architecture are the old i-vector, relatively old d-vector, and the state-of-the-art x-vector architecture. The i-vector architecture is quite old, though we see methods [6] formulated for the i-vector architecture used effectively on newer architecture. But the baseline i-vector architecture is quite ineffective

today, also the computational method for the i-vector is very complex. The d-vector architecture is promising, showing good results and having room for improvements. But d-vector being a Google built system documentation, and information is scarce. Pretrained models, or implementation are hard to come by. However, on the state-of-the-art x-vector architecture, the pre-trained models are available on Kaldi, an ASR toolkit. The full x-vector system was implemented with Kaldi, so we can retrieve implementation details too. We then decided to use x-vector architecture as our baseline system. We also tested Google's LSTM based model to have some insight on basic diarization approaches. We tested the model on a Bangla dataset, which we had to collect from the owner. We have prepared our own dataset of Bengali meetings. So, we will be training the existing state-of-the-art model with our dataset and bring various changes in the hyperparameters and other features to bring possible improvements.

Chapter 10

Bibliography

Reference

- [1] D. Reynolds and P. Torres-Carrasquillo, "Approaches and Applications of Audio Diarization", Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.
- [2] K. Selvan, A. Joseph and K. Anish Babu, "Speaker recognition system for security applications", 2013 IEEE Recent Advances in Intelligent Computational Systems (RAICS), 2013.
- [3] P. Verma and P. Das, "i-Vectors in speech processing applications: a survey", International Journal of Speech Technology, vol. 18, no. 4, pp. 529-546, 2015.
- [4] Eda Kavlakoglu, "AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference?", IBM.com. <https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks> [Accessed: 08- Jan- 2022]
- [5] S. Saha, "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. Towards Data Science.", Medium. <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53> [Accessed: 08- Jan- 2022]
- [6] M. Phi. "Illustrated Guide to LSTM's and GRU's: A step by step explanation | by Michael Phi | Towards Data Science.", Medium. <https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21> [Accessed: 08- Jan- 2022]
- [7] Y. Verma, "Complete Guide to Bidirectional LSTM (With Python Codes)", Analytics India Magazine. <https://analyticsindiamag.com/complete-guide-to-bidirectional-lstm-with-python-codes/> [Accessed: 08- Jan- 2022]
- [8] Wikipedia contributors, "Speech processing", Wikipedia. https://en.wikipedia.org/wiki/Speech_processing [Accessed: 08- Jan- 2022]
- [9] S. Prasad, "Different Types of Clustering Methods and Applications", Analytixlabs. <https://www.analytixlabs.co.in/blog/types-of-clustering-algorithms/> [Accessed: 08- Jan- 2022]
- [10] Xavier A. "Diarization Error Rate", Xavieranguera.com. <http://www.xavieranguera.com/phdthesis/node108.html> [Accessed: 11- Dec- 2021]
- [11] K. Weiss, T. Khoshgoftaar and D. Wang, "A survey of transfer learning", Journal of Big Data, vol. 3, no. 1, 2016.
- [12] D. Martinez, "Is Transfer Learning the final step for enabling AI in Aviation?", Datascience.aero. <https://datascience.aero/transfer-learning-aviation/> [Accessed: 11- Dec- 2021].

- [13] H. Maheshwari. "Understanding Domain Adaptation", Medium. <https://towardsdatascience.com/understanding-domain-adaptation-5baa723ac71f> [Accessed: 11- Dec- 2021].
- [14] A. Zheng, "Evaluating Machine Learning", O'Reilly Online Learning. <https://www.oreilly.com/library/view/evaluating-machine-learning/9781492048756/cho4.html> [Accessed: 11- Dec- 2021].
- [15] Multimodal Information Group, "Rich Transcription Evaluation", Nist.gov. <https://www.nist.gov/itl/iad/mig/rich-transcription-evaluation> [Accessed: 11- Dec- 2021].
- [16] T. Park, N. Kanda, D. Dimitriadis, K. Han, S. Watanabe and S. Narayanan, "A review of speaker diarization: Recent advances with deep learning", *Computer Speech & Language*, vol. 72, p. 101317, 2022.
- [17] H. Gish, M. Siu, R. Rohlicek, Segregation of speakers for speech recognition and speaker identification, in: *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 1991, pp. 873–876.
- [18] M.-H. Siu, Y. George, H. Gish, An unsupervised, sequential learning algorithm for segmentation for speech waveforms with multiple speakers, in: *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 1992, pp. 189–192.
- [19] J. R. Rohlicek, D. Ayuso, M. Bates, R. Bobrow, A. Boulanger, H. Gish, P. Jeanrenaud, M. Meteer, M. Siu, Gisting conversational speech, in: *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 1992, pp. 113–116.
- [20] U. Jain, M. A. Siegler, S.-J. Doh, E. Gouvea, J. Huerta, P. J. Moreno, B. Raj, R. M. Stern, Recognition of continuous broadcast news with multiple unknown speakers and environments, in: *Proceedings of ARPA Spoken Language Technology Workshop*, 1996, pp. 61–66.
- [21] M. Padmanabhan, L. R. Bahl, D. Nahamoo, M. A. Picheny, Speaker clustering and transformation for speaker adaptation in large-vocabulary speech recognition systems, in: *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 1996, pp. 701–704.
- [22] J.-L. Gauvain, L. Lamel, G. Adda, Partitioning and transcription of broadcast news data, in: *Proceedings of the International Conference on Spoken Language Processing*, 1998, pp. 1335–1338.
- [23] D. Liu, F. Kubala, Fast speaker change detection for broadcast news transcription and indexing, in: *Proceedings of the International Conference on Spoken Language Processing*, 1999, pp. 1031–1034.
- [24] S. S. Chen, P. S. Gopalakrishnan, Speaker, environment and channel change detection and clustering via the Bayesian Information Criterion, in: *Tech. Rep., IBM T. J. Watson Research Center*, 1998, pp. 127–132.

- [25] AMI dataset, 2022. [Online]. Available: <http://www.amiproject.org/index.html>. [Accessed: 05- Mar- 2022].
- [26] Ning, H., Liu, M., Tang, H., & Huang, T. A spectral clustering approach to speaker diarization. In INTERSPEECH 2006 and 9th International Conference on Spoken Language Processing, INTERSPEECH 2006 - ICSLP, 2006. (Vol. 5, pp. 2178–2181). International Speech Communication Association.
- [27] Tranter, Sue & Reynolds, Douglas. An overview of automatic speaker diarization systems. IEEE Transactions on Audio, Speech & Language Processing, 2006. 14. 1557-1565. 10.1109/TASL.2006.878256.
- [28] Castaldo, Fabio & Colibro, Daniele & Dalmaso, Emanuele & Laface, Pietro & Vair, Claudio. Stream-based speaker segmentation using speaker factors and eigenvoices. ICASSP, IEEE International Conference on Acoustics, Speech, and Signal Processing - Proceedings, 2008. 4133 - 4136. 10.1109/ICASSP.2008.4518564.
- [29] Sun, H., Nwe, T. L., Ma, B., & Li, H. Speaker diarization for meeting room audio. In Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH 2009. (pp. 900–903).
- [30] Valente, Fabio & Vijayaseenan, Deepu & Motlíček, Petr. Speaker diarization of meetings based on speaker role n-gram models, 2011. 4416-4419. 10.1109/ICASSP.2011.5947333.
- [31] Wang, Q., Downey, C., Wan, L., Mansfield, P. A., & Moreno, I. L. Speaker diarization with LSTM. In ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings (Vol. 2018-April, pp. 5239–5243). Institute of Electrical and Electronics Engineers Inc.
- [32] Zhang, A., Wang, Q., Zhu, Z., Paisley, J., & Wang, C. Fully Supervised Speaker Diarization. In ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings (Vol. 2019-May, pp. 6301–6305). Institute of Electrical and Electronics Engineers Inc.
- [33] Sharma, V., Zhang, Z., Neubert, Z., & Dyreson, C. Speaker Diarization: Using Recurrent Neural Networks. arXiv preprint, 2020. arXiv:2006.05596.
- [34] N. Dawalatabad, S. Madikeri, C. C. Sekhar and H. A. Murthy, "Novel Architectures for Unsupervised Information Bottleneck Based Speaker Diarization of Meetings," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 29, pp. 14-27, 2021,
- [35] P. Kenny, D. Reynolds, F. Castaldo, Diarization of telephone conversations using factor analysis, IEEE Journal of Selected Topics in Signal Processing 4 (2010) 1059–1070.

- [36] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, P. Ouellet, Front-end factor analysis for speaker verification, *IEEE Transactions on Audio, Speech, and Language Processing* 19 (2011).
- [37] S. Shum, N. Dehak, E. Chuangsuwanich, D. Reynolds, J. Glass, Exploiting intra-conversation variability for speaker diarization, in: *Proceedings of the Annual Conference of the International Speech Communication Association*, 2011.
- [38] S. Shum, N. Dehak, J. Glass, On the use of spectral and iterative methods for speaker diarization, in: *Proceedings of the Annual Conference of the International Speech Communication Association*, 2012, pp. 482–485.
- [39] S. H. Shum, N. Dehak, R. Dehak, J. R. Glass, Unsupervised methods for speaker diarization: An integrated and iterative approach, *IEEE Transactions on Audio, Speech, and Language Processing* 21 (May 2013) 2015– 2028.
- [40] M. Senoussaoui, P. Kenny, T. Stafylakis, P. Dumouchel, A study of the cosine distance-based mean shift for telephone speech diarization, *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 22 (2013) 217–227.
- [41] G. Sell, D. Garcia-Romero, Speaker diarization with plda i-vector scoring and unsupervised calibration, in: *Proceedings of IEEE Spoken Language Technology Workshop*, IEEE, 2014, pp. 413–417
- [42] Ghahremani, Pegah, Vimal Manohar, Daniel Povey and S. Khudanpur. “Acoustic Modelling from the Signal Domain Using CNNs.” *INTERSPEECH*, 2016
- [43] Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., ... Vesely, K. *The Kaldi Speech Recognition Toolkit* (2011). IEEE Signal Processing Society.
- [44] Wan, L., Wang, Q., Papir, A., & Moreno, I. L. Generalized end-to-end loss for speaker verification. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* (Vol. 2018-April, pp. 4879–4883). Institute of Electrical and Electronics Engineers Inc.
- [45] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667.
- [46] McFee, Brian, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. “librosa: Audio and music signal analysis in python.” In *Proceedings of the 14th python in science conference*, pp. 18-25. 2015.
- [47] C. Jemine, “Resemblyzer”, Github Repository. <https://github.com/resemble-ai/Resemblyzer> [Accessed: 05- Mar- 2022]
- [48] Stevo Bozinovski. “Reminder of the First Paper on Transfer Learning in Neural Networks, 1976”. In: *Informatica* 44 (Sept. 2020).

- [49] A. d. S. P. Soares et al., "Energy-based voice activity detection algorithm using Gaussian and Cauchy kernels," 2018 IEEE 9th Latin American Symposium on Circuits & Systems (LASCAS), 2018, pp. 1-4.
- [50] X. -L. Zhang and J. Wu, "Denoising deep neural networks-based voice activity detection," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 853-857.
- [51] C. Jemine, "Resemblyzer allows you to derive a high-level representation of voice through a deep learning", CuratedPython. <https://curatedpython.com/p/resemblyzer-allows-resemble-ai-resemblyzer/index.html> [Accessed: 05- Mar- 2022]