A
PROJECT REPORT
ON

Conversation State Prediction System

SUBMITTED TO

SHIVAJI UNIVERSITY, KOLHAPUR
IN THE PARTIAL FULFILLMENT OF REQUIREMENT FOR THE
AWARD OF DEGREE BACHELOR
OF

ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING

SUBMITTED BY

Ms. Divya Patil - 18UCS073
Mr. Shubham Rangate - 18UCS096
Mr. Sujay Uday Rittikar - 18UCS097
Ms. Mugdha Sathe - 18UCS103
Ms. Vaidehi Rathor - 18UCS122

UNDER THE GUIDANCE
OF

Prof. U. A. NULI

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
DKTE SOCIETY'S TEXTILE AND ENGINEERING
INSTITUTE,ICHALKARANJI

2021-2022

D.K.T.E. SOCIETY'S
TEXTILE & ENGINEERING INSTITUTE,
ICHALKARANJI
(AN AUTONOUMOUS INSTITUTE)

DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING

CERTIFICATE

This is to certify that, project work entitled

Conversation State Prediction System

is a bonafide record of project work carried out in this college by

Ms. Divya Patil - 18UCS073
Mr. Shubham Rangate - 18UCS096
Mr. Sujay Uday Rittikar - 18UCS097
Ms. Mugdha Sathe - 18UCS103
Ms. Vaidehi Rathor - 18UCS122

is in the partial fulfillment of award of Bachelor Degree of Technology in Computer Science & Engineering prescribed by Shivaji University, Kolhapur for the academic year 2021-2022.

Prof. U. A. NULI
(PROJECT GUIDE)

PROF.(DR.) D.V.KODAVADE
(HOD CSE DEPT.)

PROF.(DR.) P.V.KADOLE
(DIRECTOR)

EXAMINER:

ABSTRACT

The evolution of the concept of Speaker Diarization using LSTM facilitated the process of understanding the speaker identities for specific segments of input audio stream data without manually tagging the data. Our study focuses on the possibility of the emergence of the concept of Speaker State Prediction based on the sequence of speaker states and their contexts in a conversation. In this study, the Markov Chains are used to identify and predict the Speaker State for the next dialogue among speakers on the basis of previous contexts. The application of distance metric on speaker-specific contextual similarity enables the possibility of prediction of the speaker states in the most natural and long conversations specifically. The system has a prediction accuracy of 61% calculated using live audio conversations. The findings imply that the proposed method is effective to predict a speaker's state during a conversation.

DECLARATION

We hereby declare that, the project work report entitled **Conversation State Prediction System** which is being submitted to D.K.T.E. Society's Textile and Engineering Institute Ichalkaranji, affiliated to Shivaji University, Kolhapur is in partial fulfillment of degree B.Tech (CSE). It is a bonafide report of the work carried out by us. The material contained in this report has not been submitted to any university or institution for the award of any degree. Further, we declare that we have not violated any of the provisions under Copyright and Piracy / Cyber / IPR Act amended from time to time.

ACKNOWLEDGEMENT

With great pleasure we wish to express our deep sense of gratitude to U. A. Nuli for his valuable guidance, support and encouragement in completion of this project report. Also, we would like to take opportunity to thank our head of department Dr. D. V. Kodavade for his co-operation in preparing this project report. We feel gratified to record our cordial thanks to other staff members of Computer Science and Engineering Department for their support, help and assistance which they extended as and when required.

Thank you,

Ms. Divya Patil - 18UCS073
Mr. Shubham Rangate - 18UCS096
Mr. Sujay Uday Rittikar - 18UCS097
Ms. Mugdha Sathe - 18UCS103
Ms. Vaidehi Rathor - 18UCS122

INDEX

Contents

1 Introduction

Speaker Recognition (Zhongxin Bai, et al., 2020) is a process to recognize the identity of a speaker as well as verify the identity. Although, the scopes of Speaker identification and Speaker verification are different and may merge based on the applications. Speaker identification has evolved with several text-dependent methods like HMM-Based Methods and text-independent methods such as the VQ-Based approach (Lei Z., et al., 2005) followed by SVM classification. Speaker diarization(Quan Wang, et al., 2017) is a process used to perform Speaker identification using partitioning an input audio stream into homogeneous segments according to the speaker's identity. According to Speaker diarization, it's possible to identify the multiple speaker states in an audio stream.

The speaker diarization system is based on the use of Audio embeddings in form of text-independent d-vectors(Jung, J., et al., 2018) to train the LSTM-based (Sepp Hochreiter and J urgen Schmidhuber, 1997) speaker verification model and furthermore, combine the model with a spectral clustering algorithm to obtain the text-independent states. Thus, the study in this report is based on the live speech audio streams having firmly speaker voices with minimum noise levels.

In recent years, Conversational AI (P. Kulkarni, et al., 2019) is used in various applications such as Chatbots and other virtual assistants. It works on the principles of Natural Language Processing (Khurana, et al., 2017) and is dependent on a set of steps essential to identify, retrieve and predict the characteristics of a conversation. The concepts of preprocessing NLP techniques such as tokenization and lemmatization, intent classification, entities extraction, featurizer, and response selector are based on various text-related and vector-based operations and, Machine Learning and Deep Learning-based models such as SVM, Bayesian Networks (Weissenbacher, Davy, 2006), Word2Vec and LSTM.

Conversational AI richly deals with intent identification and its usage to recognize the flow of conversation and response selection. Although, with advancements in such AI systems (Sungdong Kim, et al., 2019), the speakers are no more limited to the speaker and the system. The count of speakers in such conversations is higher than two and may cause a problem in selecting the response for a specific user on the basis of intents alone. Thus, our problem focuses on providing a solution for such situations.

1.1 Problem Definition and Description

Problem Statement: To develop the conversation state prediction system on the basis of live conversational audio data and the extracted context.

Problem Description: The current chatbots and robots in multi-user environments need contexts in order to be created and thus have a dependency on in-advance context knowledge and no existence of user personalization (Javier Cebrian, et al., 2021). The problem of Speaker State Prediction focuses on creating a system that can converse with multiple users at the same time and without the existence of hardcoded training data based on knowledge of contexts. In this system, we initially use the process of Speaker diarization to identify the speaker identities in a conversation and use these as speaker states to find the probable speaker state for the next conversation holding account of the same speakers. This will improvise the Conversational AI systems to identify the speaker's state of a future dialogue in conversation in advance.

1.2 Aim and objective of the project

Aim: Predict the next speaker state on the basis of speaker-mapped live accumulated contextual data.

Objectives:

1. Recognize the speaker's identities from the conversation: This objective refers to identifying who is speaking in the current conversation. The speaker diarization process aids in the identification of speakers. Using these speakers' identities, further prediction of the next speaker will be done.

2. Next speaker state prediction: The goal of this objective is to predict the next speaker who will continue the further conversation. The recognized speakers along with their speaker-specific contexts are used to predict the future speaker using the Markov chains, based on distance scores between previous contexts and current contexts.

1.3 Scope and limitation of the project

Scope:

The project highlights the gain of possible predictive power to the currently growing Conversational AI systems, based on voice-based improvements. The current evolving algorithms in the small domain of Speaker Recognition have gained momentum with the rise in their applications in the real world, such as

Speaker Diarization using LSTM-based embeddings which are actively being used by Google Inc. in their popular product: Google Assistant. Using the same intuition, we build a predictive extension to the speaker diarization system, making it more reliable to find speaker identities and computationally less extensive. In addition, the system can have more applications associated with it, considering the idea of knowing the state of the future speakers in advance. With such consideration and thorough research on its applications which include building predictive movie transcripts based on some initial inputs, easier authentication systems even in multi-user environments in case of small-scale applications, in the legal matter based on predicting the speaker state of a multi-user conversation where we can identify a speaker based on the proof of context, providing proof of the existence of a certain speaker in a conversation related to crime, entertainment purpose while building games as well as in phonetic customer service systems. With such a wide range of applications and much more, it is considerate to build such a system and develop its efficiency to make it reliable.
Input: Speech data, in form of live audio stream
Output: The predicted speaker state

Limitations:

1. Lower performance on large audio data, unless the dimensionality is increased.
2. Can only perform text-dependent speaker verification.
3. Not very adaptive to the applications with subject-independent conversations, considering continued flows of the same speakers.

1.4 Timeline of the project

Month	Task performed	Description
June	Project Domain selection and finalization. Selection of Problem Statement	Analyzed various project domains and finalized our project domain. Considered various problems related to that domain and selected one to resolve.
July	Studying	Study on various Research papers.
August	Synopsis Documentation	Documentation of synopsis and requirement Analysis.
September / October	System Requirement. Module Identification and study	Identified the system requirements. Studied and identified the required modules in detail.
November	SRS Documentation and presentation	Created the Software Requirement Specification Document (SRS Document) and Powerpoint
December /January	Coding 30%. Data Collection. Implementation 30%. Testing and Improvements	30% coding done. Collection of the required data for code implementation. Testing and improvements in the required part.Analysis of limitations and drawbacks.
February	Data Collection. Coding 70%.Implementation 70%.Testing and Improvements. Code updation.	Collection of data in different formats due to change of approach.70% coding done according to updated approach. Implementation and Testing. Updation of code for meeting the desired requirements.
March	Coding 100%. project report is created.	100% coding done. Overall system testing is performed.

1.5 Project Management Plan

Task	Duration	Start Date	End Date	Priority
Domain Selection	7 days	02-07-2021	08-07-2021	High
Domain Finalization	7 days	09-07-2021	15-07-2021	High
Selection of Problem Statement	14 days	16-07-2021	29-07-2021	Medium
Finalization of Problem Statement	7 days	30-07-2021	05-08-2021	High
Study on Research Paper	14 days	06-08-2021	19-08-2021	Medium
Documentation of Synopsis	14 days	20-08-2021	02-09-2021	High
Requirement Analysis	7 days	03-09-2021	09-09-2021	High
System Requirement	7 days	10-09-2021	16-09-2021	High
Module Identification and study	7 days	17-09-2021	22-09-2021	Medium
SRS Documentation and presentation	7 days	24-09-2021	01-10-2021	Medium
Coding 30%. Data Collection.	14 days	02-12-2021	15-12-2021	High
Implementation 30%.	7 days	16-12-2021	22-12-2021	High
Data Collection. Coding 70%	14 days	23-12-2021	05-01-2022	High
Implementation 70%.	14 days	06-01-2022	24-01-2022	High
Testing and Improvements.	14 days	25-01-2022	07-02-2022	High
Code updation	7 days	08-02-2022	14-02-2022	High
Coding 100%	14 days	16-02-2022	1-03-2022	High
Testing Implementation 100%	7 days	2-03-2022	8-03-2022	High

1.6 Project Cost

COCOMO RESULTS for conversation state prediction system								
MODE	**"A" variable**	**"B" variable**	**"C" variable**	**"D" variable**	**KLOC**	**EFFORT, (in person-months)**	**DURATION, (in months)**	**STAFFING, (recommended)**
organic	2.4	1.05	2.5	0.38	0.280	0.631	2.098	0.301

Explanation: The coefficients are set according to the project mode selected on the previous page, (as per Boehm). Note: the decimal separator is a period.

The final estimates are determined in the following manner:

effort = a*KLOC^b, in person-months, with KLOC = lines of code, (in thousands), and:

staffing = effort/duration

where a has been adjusted by the factors:

Product Attributes	
Required Reliability	1.00 (N)
Database Size	1.00 (N)
Product Complexity	1.00 (N)
Computer Attributes	
Execution Time Constraint	1.00 (N)
Main Storage Constraint	1.00 (N)
Platform Volatility	1.00 (N)
Computer Turnaround Time	1.00 (N)
Personnel Attributes	
Analyst Capability	1.00 (N)
Applications Experience	1.00 (N)
Programmer Capability	1.00 (N)

2 Background study and literature overview

2.1 Literature overview

The Speaker Recognition Technology is evolving through several methodologies. Speaker Recognition means finding the identity of speakers in an audio stream, involving both speaker identification and speaker verification. Clark D et al., 2016 [1] highlight the evolution of the technology of Speaker Recognition through various advancements in Audio Signal Processing. The evolution of concepts: Spectrograms, Cepstrum-based systems, and GMM-based systems in order to provide additional evidence of the possibility of the Speaker Recognition system.

While the GMM-based systems produced commendable results, using SVM and joint factor analysis, the technology further evolved through the use of probabilistic linear discriminant analysis (PLDA) as the state-of-the-art technique for many years. Noor Salwani Ibrahim et al., 2018 [2], proposed I-vector Extraction for Speaker Recognition Based on Dimensionality Reduction, which emphasizes performing speaker recognition using the subspaces of channel variation and speaker variation that are further interpreted in low dimensional space, called "total variability space". The i-vector embeddings can be further used to obtain speaker identities by performing clustering using the Gaussian Mixture Model (GMM). While the proposal has effective speaker recognition, it observes lower performance on large audio data, unless the dimensionality is increased. Also, it can only perform text-dependent speaker verification and is not very adaptive to real-world applications, considering continued flows of the same speakers.

The use of Deep Learning techniques for Speaker Recognition has recently taken a pace. Zhongxin Bai et al., 2020 [3], illustrate a proper outlook of usage of Deep Learning in Speaker Recognition, through the evolution from i-vectors to the use of DNN for producing embeddings.

The term Speaker Diarization evolved through the work of Quan Wang et al., 2017 [4], "Speaker Diarization with LSTM". It proposes the speech embeddings based on d-vectors, which are the embeddings generated using an LSTM model. The Speaker diarization can be performed on the text-independent audio data and the d-vectors are known to be better performing than the i-vectors in terms of recognizing the speaker identities. While there are not many limitations in this system design considering the overall efficiency to identify speaker identities, the model is computationally extensive. Therefore, the problem of a slower response rate is a given. The system can be further used to develop predictive systems that will be able to predict the flow of future conversations considering the given sample of speakers.

Furthermore, there are many current problems when it comes to Humanoid Robot Interaction (HRI), a research field that specifically focuses on how we can make Human-Robot interactions easier and more human-like, including the perceptually more personalized aspects. Specifically, the Long-term human-robot interactions and how these interactions can become relatively real-life, based on the work of Irfan et al., 2019 [5], is a vital subject for the ever-evolving field of Robotics.

In addition, such ideology can also be applied to chatbot systems, since these are also becoming more focused on the Multi-agent environments as included by Zojan Memon et al., 2018 [6], which include applications such as Virtual Reality.

Considering these aspects, we propose a possible implementation of the Speaker State Prediction System, i.e., a system that can predict "Who will speak next?". We implemented several methodologies to find the most accurate way to predict, including the Pattern-based State prediction using Markov Chains as mentioned by Sujay Rittikar, 2021 [7], and some context-based methods including entities-based prediction system and, the whole context-based prediction system at the word level. The context-based prediction system at the word level was found more reliable and accurate considering the use-cases. While it has some flaws, it stands as a fundamental approach to the problem and the basis of future developments in this research.

2.2 Critical appraisal of other people's work

The approach used for Speaker Diarization by Quan Wang et al., 2017 [4] represents very new work for the domain of Speaker Recognition since, there hasn't been any work that represents the speakers by their identities and time intervals, especially given a live audio data as an input. However, this technique encircles around the Spectral Offline Clustering which in the implementation phase has problems of not considering the previous clusters created. The Clustering methodology, while going on updating the clusters as it receives more data, these clusters can't be confirmed to be the same as an iteration before. Therefore, during the process of Speaker Diarization, in a live audio experimental setting, the speaker labels that we are eager to assign for each iteration of recording in a live recording, may get mismatched to the clusters. Therefore, producing wrong speaker identities.

The reason behind this problem to be considered is because, in a live audio setting, we need to process the audio data as we receive it dynamically and then perform Speaker Diarization and, further on the Speech to Text Recognition as well as our proposed method of Speaker State Prediction.

This processing may become tedious and inaccurate because of the wrong speaker identities as a result of the Clustering method in the first place. It impacts the whole system that we are proposing. Therefore, the Spectral Offline Clustering method has to be modified, with respect to its clustering centroids so as to make sure that it doesn't change in the conditions of new data being fed. Or, another method of clustering is recommended that will encompass all the speaker identities while identifying the number of speaker identities. The solution to this problem will increase our developed system's accuracy and overall speed.

2.3 Investigation of current project and related work

We started our project with the aim of "Conversation State Prediction System". This means we had objects to identify the speaker states, identify the flow of the speakers in the conversation and predict the future flow of speakers in a conversation. We also aimed to make the system computationally less expensive and make it optimal for conversational AI. The whole system we designed was pattern-based. The system predicted the speakers based on the previous patterns of their involvement in the conversation.

Thus, we opted to use speaker diarization, clustering and Markov chains. But the Markov chains are dependent on the following hyperparameters:

1. Sensitivity of error.
2. Window size/ size of labels

3 Requirement analysis

3.1 Requirement Gathering

1. A user should be able to access the mic of the system.
2. A user should be able to start conversations with multiple speakers and trigger the system to perform operations.
3. A user can expect the next predicted speaker state as the final result of the system.

3.2 Requirement Specification

Sr. No.	Requirements	Essential	Description
RS1	The system should have access to a mic to record	Essential	Record live audio conversation
RS2	The system should have an efficient Speech to Text Converter	Essential	Convert live audio conversations into text
RS3	The conversations used to use the system should have the specific language	Essential	Speech to Text Converter language is required for the conversation
RS4	The system should identify the speaker identities	Essential	The Speaker Diarization requires proper identification of speaker identities

Cont'd on following page

RS5	The system should generate text export file	Essential	A text export file should be generated with properly mapped conversations to corresponding speakers
RS6	The system should create an efficient Word2Vec model	Essential	Word2Vec model is required to be efficient enough to make sure the prediction process is proper
RS7	The current speaker should be identified	Essential	In the live audio, the current speaking person should be identified to decide whether to predict a speaker state
RS8	The system should predict accurate speaker state	Essential	Markov Chains are used to predict the accurate speaker state using the distance metrics in its transition matrix

3.3 Use case Diagram

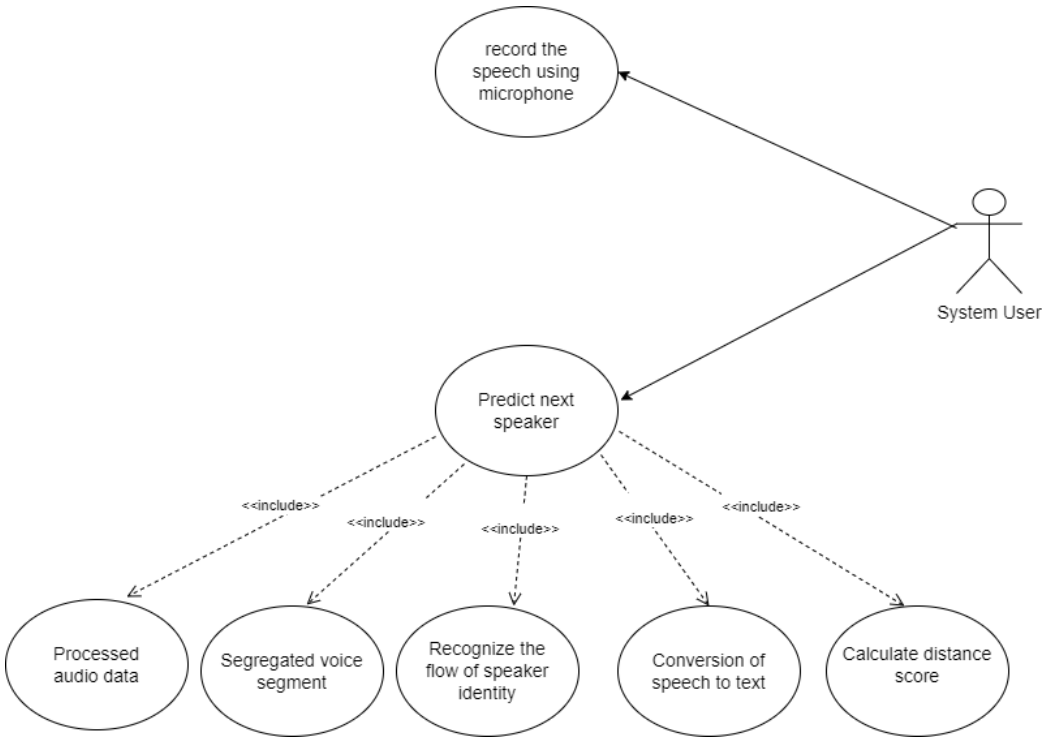


Figure 1: Use Case Diagram

4 System design

4.1 Architecture Design

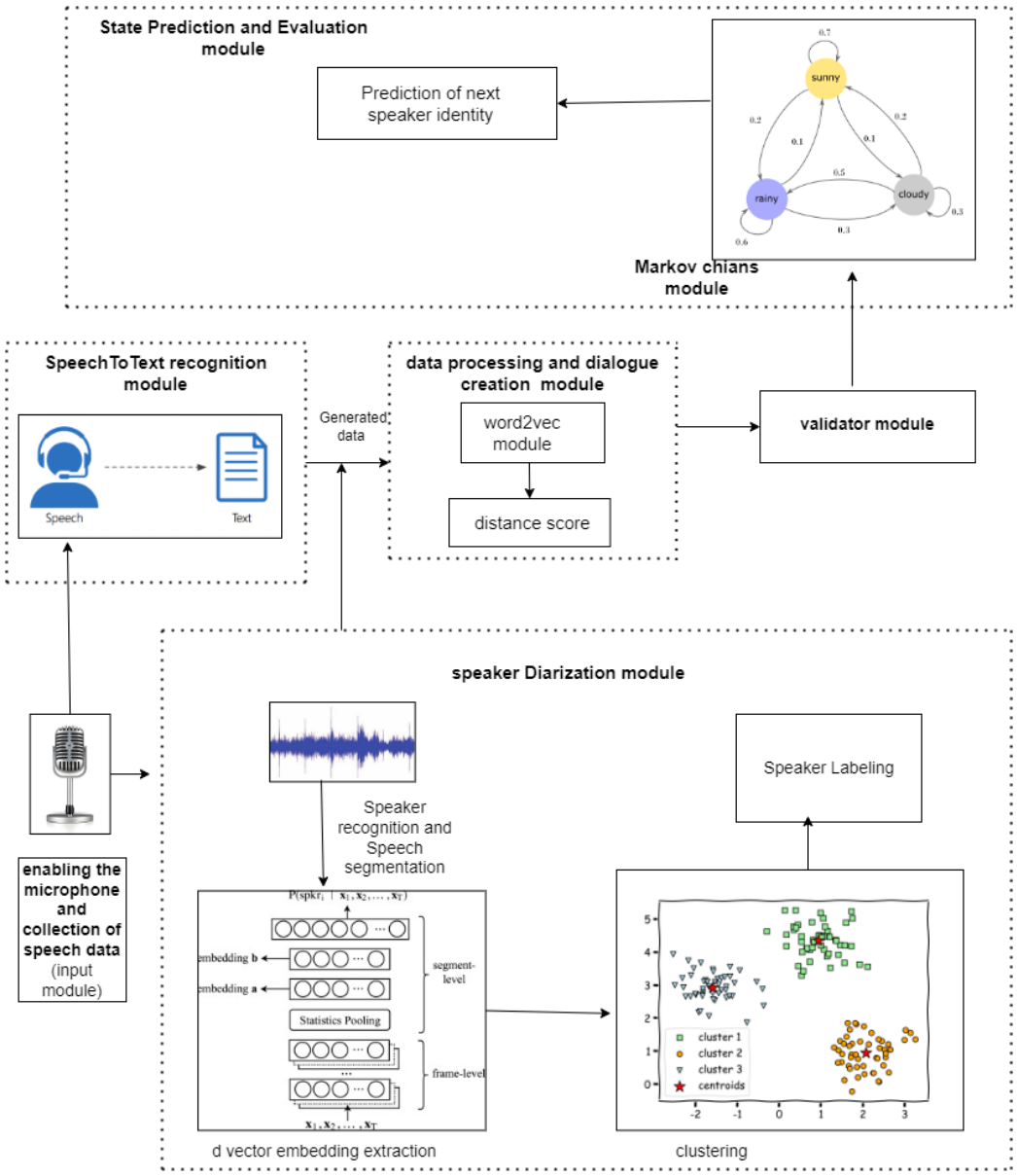


Figure 2: Architecture Design

4.2 Algorithmic Description of each module

Below is the algorithmic description of all the important algorithms used in the project:

K-means clustering algorithm:

This algorithm is an unsupervised machine learning algorithm used for clustering. It groups the dataset into different clusters. The number of clusters 'k' needs to be predefined and for each group, a centroid is found. The main aim of the algorithm is to reduce the sum of distances and create the clusters.

Initially k is defined. Then randomly the centroids are found and each data point is assigned to their closest centroid, which will form the predefined K clusters. Then variance is calculated and then new centroids are assigned to each cluster. Again the previous steps of reassigning the data points w.r.t their nearest centroids is done and final clusters are obtained.

Markov Chains algorithm:

It is an algorithm used to show the sequence of possible events. The probability of the next event is dependent on the previous event only. This property is known as the markov property or memory-lessness. The probabilities of each state are represented in the matrix form. This matrix is known as the transition matrix. If the Markov chain has N possible states, the matrix will be an NxN matrix. Each row of this matrix should sum to 1. In addition to this, a Markov chain also has an Initial State Vector of order Nx1. These two entities are a must to represent a Markov chain.

Large Audio Transcription Algorithm:

This algorithm is used to break the large audio file in smaller chunks and then perform the speech recognition over those chunks. This is done by splitting the audio at the point of silence between the sentences and forming smaller chunks of data. These chunks are then fed to the speech recognition API.

PCM to Float Data Conversion Of Audio Signal Algorithm:

This conversion is done to transform the pulse code modulated data, i.e the audio data into the float format .16 bit PCM has a range - 32768 to 32767. So to convert it into float form, multiply each of the PCM samples by (1.0f/32768.0f) into a new array of floats, and it is passed to the resample module.

Recording Algorithm: This algorithm is used to record and save the live recordings. The recordings are recorded and saved in the wav file. And as the recordings are live, they are updated and saved after specific intervals of time. It returns the bytesize and bytearray of the wav file.

Voice Activity Detection Algorithm:

In Voice Activity Detection the presence or absence of speech in an audio segment is detected. It is done by taking the audio segments which are split into segments of 5-40ms duration as an input. Then two features namely signal energy and spectral centroid are extracted and compared to the threshold value. Then depending upon the value, the decision about whether or not speech is present is taken. The wav_splits containing only the speech part are returned.

Offline Spectral Clustering Algorithm:

This function is used to generate the speaker labels of all the embeddings that are produced by the d-vectors. Clustering algorithm consists of following steps:

1. Construction of affinity matrix.
2. Applying the sequence of refinement operations on the affinity matrix which includes Gaussian blur with standard deviation, row-wise thresholding, symmetrization, diffusion, row-wise max normalization.

getMappedTranscript Algorithm:

In this algorithm, the identified chunks of text (identified by STT model) are assigned to the speaker labels. This is done by comparing the probabilities of the chunks and the maximum probable chunk is assigned to the specific speaker label. A map with the mapping of speaker labels and the chunks is generated with a name transcriptMap.

SpeakerDiarization and STT Algorithm:

Speaker Diarization algorithm is a combination of multiple sub modules which include data collection, speech detection, speech segmentation, embedding extraction and clustering. STT (Speech to text) Algorithm is used to convert the audio signals (vibrations) into the text format using digital converters and the concept of phonemes.

getSpeakerSpecificDialgoues Algorithm:

This algorithm is used to map the identified dialogues to the respective speaker label in a dictionary in a key:value format.

getTransitionMatrix Algorithm:

In this algorithm, all the texts for a speaker are aggregated into one single string and then it is used to find the transition matrix. To form the transition matrix, the distance between all the pairs of aggregated text lists of speakers is found and then added in the transition matrix. This gives the distances between the speakers' text in matrix form.

getConversationDistances Algorithm:

This algorithm is used to find the distance between all the dialogues of the speakers and the new dialogue which gets added in the dialogue list as the speakers continue the speech. Whenever any new dialogue is added, its distance with all the already available dialogues is found. This algorithm returns the dictionary of distances.

getAveragedDistancesVector Algorithm:

This algorithm initially checks if any nan or null values are present to avoid considering the outliers in the calculations. Then the average of distances for each speaker state is found. It returns an averaged distance vector.

4.3 System Modeling

4.3.1 Dataflow Diagram

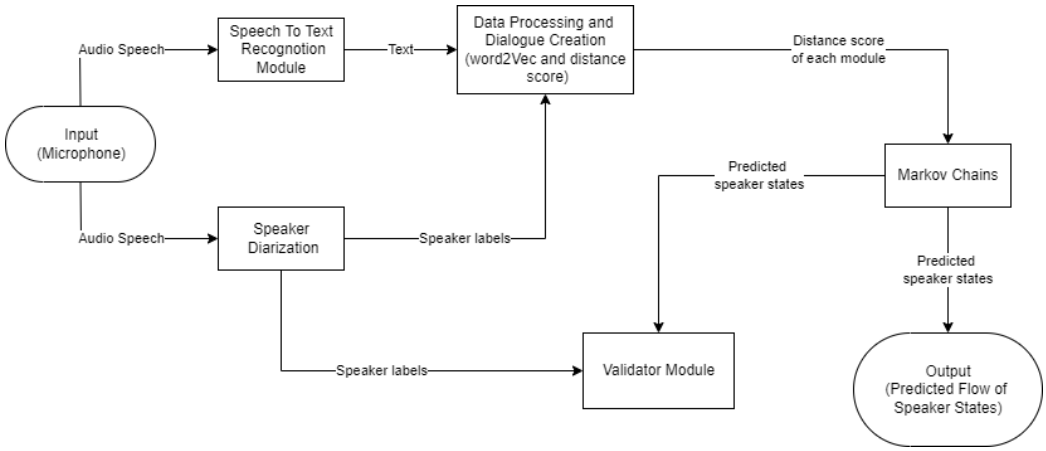


Figure 3: Dataflow Diagram

4.3.2 Sequence Diagram

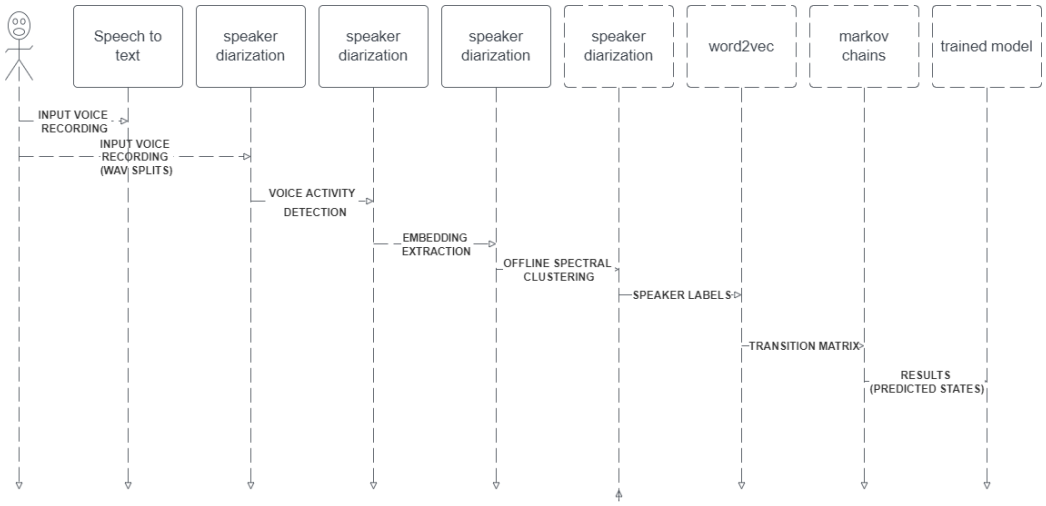


Figure 4: Sequence Diagram

4.3.3 Activity Diagram

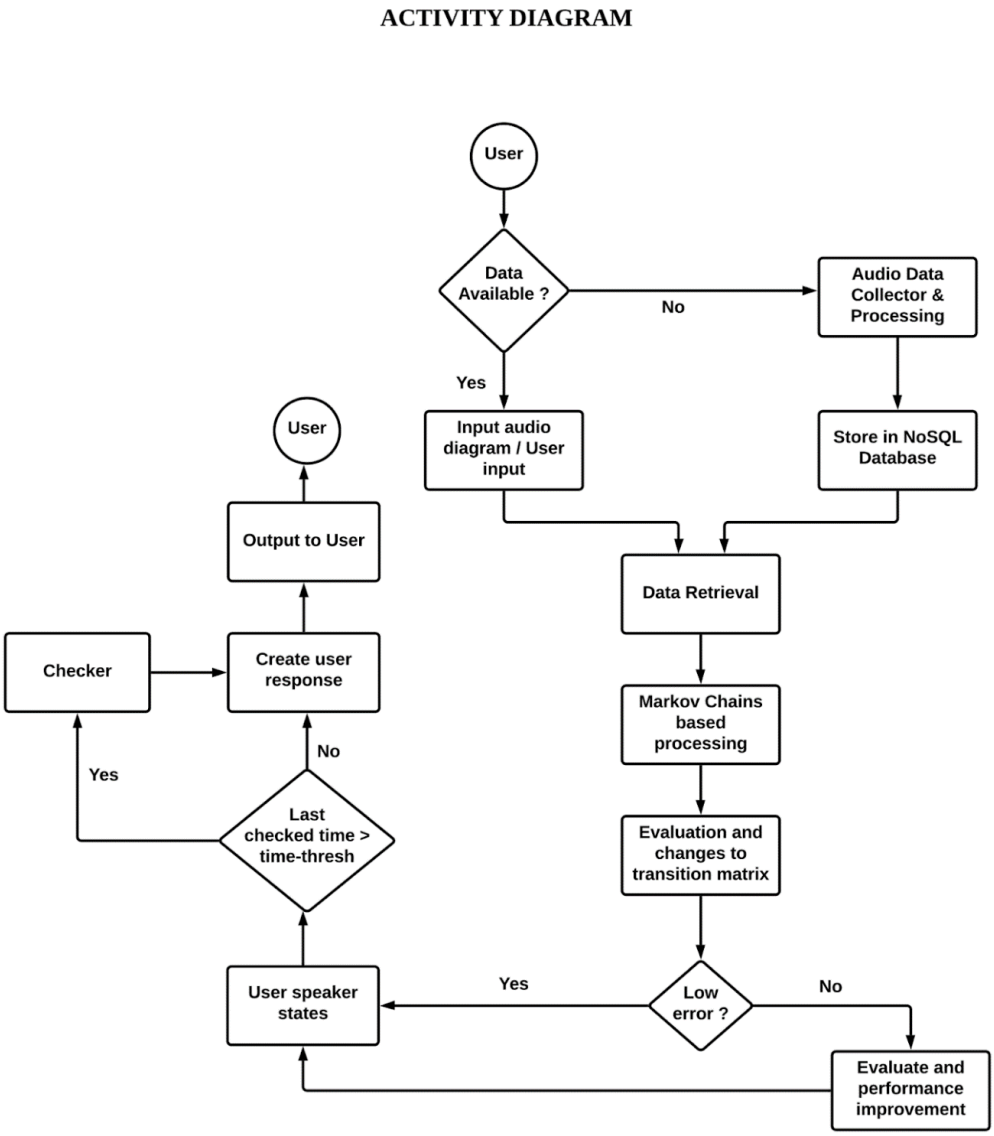


Figure 5: Activity Diagram

5 Implementation

5.1 Environmental Setting for Running the Project

For running the project, a python environment is necessary. The libraries required to be installed are as follows:

1. resemblyzer
2. pydub
3. spectralcluster
4. audioread
5. SpeechRecognition
6. librosa
7. pandas
8. numpy
9. from gensim.models import Word2Vec The command that can be used to install these libraries is:

pip install ¡library_name¿
The other internal libraries that are needed are:

1. soundfile
2. speech_recognition
3. from pydub import audio segment
4. from pydub.silence import split_on_silence
5. from pydub import AudioSegment
6. from resemblyzer import preprocess_wav, VoiceEncoder

5.2 Detailed Description of Methods

The system consists of a lot of methods like speaker Diarization, speech to text, markov chains and distance score etc. description of each method is provided as below:

1. **Speaker Diarization method**
This method is a combination of multiple sub modules which include data collection, speech detection, speech segmentation, embedding extraction and clustering.

- **Data Collection:** In this, live data is collected continuously and dynamically through enabling the microphone of the computer. And storing it for further processing. This process takes place for an infinite number of times until a specific time that is provided by us.
- **Speech Detection:** Speech detection is a method in which actual speech part from a conversation is detected. For detecting the speech voice activity detection system is used. This system separates the speech part and non-speech part from each other by removing the silences and non-speech data. This VAD system is based on a neural network which distinguishes the speech signals from non-speech signals. After that noise is removed because the speaker recognition system deals only with the speech data. VAD takes input as overall audio data. And check the data which have values greater than some threshold value. If that audio signal has value greater than that of threshold value then this signal is termed as speech signal otherwise signal is non-speech signal.
- **Speech Segmentation:** after detecting the speech next step is to segment it. Speech segmentation is a process in which audio data is divided into small parts. These segments are continuous. This process is done because if dividing the audio data into small segments then it is easy to perform the operations on it. After dividing the audio data, applied the windowing to reduce the loss of data. Hamming window function is used as window function. This function creates overlapped windows of segments. This window consists of some size which can vary based on the size of audio data. This windowing method is important because when the segments are created then there is a possibility of missing the speech data. So, the overlapping segment concept is applied to reduce such loss.

- **Embedding Extraction:** Embedding extraction is performed on segments that got from the previous step. Embeddings are nothing but the numeric vector representation of the audio data. Embedding extraction is text-dependent method but in this system embedding extraction is text independent. These embeddings are generally based on the features of audio segments. a deep neural network i.e., LSTM (long short-term memory) is used which contains one input layer, more than one hidden layer, and one output layer. The input layer of this LSTM takes the input as the MFCCs. These MFCCs are nothing but the features or characteristics of our speech segments. These features depend on both the time and frequency domain. These MFCCs are represented by matrices in terms of numbers. Then hidden layers of LSTM perform numeric computation on the inputs. After performing the operations, at the last d-vectors are generated. D-vector is the averaged output of the last hidden layer. This d- vector is termed as the embedding. The target labels are formed as a 1-hot N-dimensional vector with the only non-zero component corresponding to the speaker identity.

- **Clustering:** Final step is clustering. In this step the clusters of embeddings are created. In short, simply groups of the embeddings are created. These groups are formed based on speaker identity. The embeddings which have the similar speaker identities are placed in the same group. There are some clustering algorithms available to perform grouping of embeddings. Here in the current system spectral offline clustering and k-means clustering to create the groups. Finally, speaker labels are given to the groups so that identify from which group the corresponding speaker identity is present.

2. Speech to text recognition method

When humans speak then some sound and vibrations are created. These vibrations together generate some words having meanings. But there are some difficulties while identifying the meaningful words. For that, a concept known as phoneme is used. A phoneme is a unit of sound that helps to distinguish one word from another. For example, consider the words pat, pan, bad, and bat. In these words, the letters p, b, d, and t are known as phonemes. The speech-to-text technology uses vibrations to recognize the audio signals and translate them into digital signals or digital language. This is done using an Analog-to-digital converter(ADC). The ADC identifies the sound, then segments it into thousands of small chunks of seconds, and then these chunks are matched with the phonemes mentioned above to find the distinct words

and try to build meaningful sentences. In the English language, there are approximately 40 phonemes. These phonemes are then matched with identified chunks and meaningful words, phrases, and sentences are built based on their relevance. Chunks that are created may be overlapped or non-overlapped. Overlapped chunks are used to reduce the loss of Data. In the current system, Google's speech recognizer is used to get a text from audio speech. This module takes input as an audio file and provides output as text which is stored in some file. This file contains text chunks of specific intervals of time.

3.Word2Vec model

After the speech to text conversion, the text data needs to be processed numerically in order to perform multiple operations. Word2vec model can be used to process this text data.

Word2Vec is a word embedding technique in Natural Language processing. It was developed by Tomas Mikolov in 2013 at Google. In NLP, the words are represented in the form of real-valued high dimensional vectors that hold (encode) the meaning of the word. This representation using vectors is such that vectors of words with similar meaning are closer to each other in the vector space.

For example, "King" and "Man" will be closer to each other than "King" and "Horse".

Word2vec represents words as semantically meaningful vectors and also preserves the relationship between words. The process of prediction of speaker state is characterized by two main components: Vectorization and Distance Metric.

a. **Vectorization:** The process of converting text data into numerical data is Vectorization. Word2Vec is a two-layered neural network. It inputs a structured set of words (text corpus) and returns a set of vectors that represent input words accurately. It uses either of the algorithms – CBOW or Skipgram, to generate vectors from the words where vectors of words with similar meanings have similar embeddings.

b. **Distance metric:** A distance metric can be referred to as the distance between two vectors. It tells how similar two words are by the distance between their respective vectors. The maximum the distance between two vectors is, the more dissimilar the words are and vice versa.For example, if there are three speakers [0,1,2] and the speaker state of a new dialogue is to be predicted, then the distance metric of the new dialogue with respect to dialogues of each speaker states is calculated, then the speaker state with least distance metric with the new dialogue is considered as the label for the new dialogue.It is represented like:

For speaker state 0, average distance score of the new dialogue is 0.0084.
For speaker state 1, average distance score of the new dialogue is 0.00973.
For speaker state 2, average distance score of the new dialogue is 0.00976.

4. Markov chains and state

After obtaining the distance scores, Markov chains are used to obtain the transition matrix.

Markov chains was named after Andrey Markov and it's a model that describes a sequence of possible events. The conversation can be seen as a sequence of words spoken by a group of people. So, using Markov chains the probability of possible next speaker state is obtained, and hence the next speaker state is calculated.

For example, a person visits a restaurant often and always orders the same food- pizza or sandwich or burger. Based on the sequence that he has ordered this food on the previous days, the Markov chain can predict what food will be ordered today. Suppose, the pizza has been ordered previously, then it gives the probability that which food will be ordered next time. If the probability of the sandwich being ordered next is greater, then there is a high chance that the person orders the food in the sequence – pizza ¿ sandwich.

In this project, Markov chains are used to form the transition matrix. Transition matrix is basically the matrix of probabilities of possible next speaker states.

Consider a matrix:

	Spst1	SpSt2	SpSt3
SpSt1	0	0.0028	0.0012
SpSt2	0.0028	0	0.0024
SpSt3	0.0012	0.0024	0

Here, every value of element represents the average distance scores between the corresponding speakers that can be considered as probabilities.

Consider a value in the above transition matrix (SpSt1, SpSt2) =¿ 0.0028

	Spst1	SpSt2	SpSt3
SpSt1	0	0.0028	0.0012
SpSt2	0.0028	0	0.0024
SpSt3	0.0012	0.0024	0

The distance between all the vectors of sentences spoken by speaker 1 and 2 are calculated and the average of these distances is calculated and it is probability that speaker2 will be speaking after speaker1. For eg, here the probability of speaker2 to speak after speaker1 is 0.0028. The probability of speaker3 to speak after speaker1 is 0.0012 and so on.Now this Transition Matrix of distance score is used for the further predictions.

5. Prediction of next speaker state

The dot product of this average distance score matrix with the transition matrix gives a list of distance scores that can be considered as the probability of the next speaker state. The least distance means greater probability and vice versa.

For Example:

For Speaker1, an average distance score with respect to new dialogue obtained is 0.00840

For Speaker2, an average distance score with respect to new dialogue obtained is 0.00973

For Speaker3, an average distance score with respect to new dialogue obtained is 0.00976

So, the average distance score matrix becomes [0.00840 0.00973 0.00976].

Here, after the dot product, the distance scores obtained were .

[4.39*10-5 5.1*10-5 3.7*10-5].

Thus, Speaker3 has the least distance score and hence a higher probability of speaking next.

5.3 Implementation Details

1. Implementation of Speaker Diarization module

In the current System Speaker Diarization module is used to find speaker state identity. This Speaker diarization module provides the speaker state sequences. The implementation includes the various functions and methods from which some are directly available and some are newly defined.

For the Implementation of this module function named SpeakerDiarization is created. This function takes input as two arguments such as embeddings_obj and minimum_clusters respectively. Here, embedding_obj is the object created of the class Embeddings which contains fields for the voice data, sampling rate, voice segments etc and functions such as VoiceEncoder, VoiceActivityDetection, OfflineSpectralClustering etc. Using this object VoiceActivityDetection function is called. This function will return the voice segments that only contain the voice data which is already preprocessed. These voice segments are stored as wav_splits. After that the next method executed is ExtractEmbeddings. This method is used to get embeddings from voice segments. For obtaining the embeddings predefined embed_utterance function is used. These embeddings are stored in cont_embeds. And finally, ExtractEmbeddings returns this cont_embeds to us. offlineSpectralClustering is the next method used to create the clusters of the embeddings. This function takes the input as the number of minimum clusters that have to be created. Output of this method is nothing but the cluster labels.

Final step in the speaker diarization module is to get speaker labels. SpeakerLabeling method is used for this purpose. This method will take input as cluster labels. Then after executing the function speaker labels are obtained. The speaker labels that are returned by the functions are stored in SpeakerLabelsOutput.

2. Speech to text

As this prediction system is based on context and words that are spoken by each speaker. So, the SpeechToText module plays an important role in the system. Main aim of this module is to generate text from audio speech. For this purpose, one one function is created and named as get_large_audio_transcript.

This function includes some predefined functions and libraries that are required to convert speech to text. In this function Google's speech_recognition module is imported. This module is used for conversion.

Then again, a pydub module is imported that will handle the other stuff like using wav file for extracting audio data, splitting audio file, merging the data etc. After that, a speech recognition object is created using the Recognizer method of the speech-recognition module. This object is named as r. Then using the AudioSegment module the audio file is accessed. After it, chunks of audio data are generated and stored in some directory. Using the object, the audio chunks are recorded and stored as audio -listened. Then this audio-listened is passed to the function recognize-google which actually tries to convert audio data to text. And for every audio chunk the text is generated and appended to the whole-text variable. Finally, the get-large-audio-transcription method returns the whole-text.

3. Implementation of Markov chain and state prediction module

In this module, the transition matrix and distance score matrix is used to get the list of distance scores that is considered as the probability of the next speaker state.For implementing this module one class is created named as contextualSpeakerState. This class has fields like dataset, listOfTexts, listOfSpeakerStates. Methods such as getTransitionMatrix, getConversationDistances, identifiedSpeaker, predictedSpeaker.

One method is created for calling this all the function as SpeakerValidationAndPrediction. In this method the object of the above -mentioned class is created. Using this object, the getTransitionMatrix method is called. Transition matrix is returned by this method as transisionMatrix. This method created the transition matrix and stores the distances between the dialogues of one speaker and another speaker. But initially the dialogue of a specific speaker is aggregated into one string. For calculating the distance, the distance score method is used. After that, the getConversationDistances method is executed in which distances of each speaker's dialogue with input text is calculated. List Distances is returned by this method. After that, the getAveragedDistancesVector method is executed in which the average of distances of each speaker is calculated. averagedDistancesVector as a list is returned by this method. Next method invoked is identifiedSpeaker. Input to this method is the average of distances. Then minimum distance from the list of averagedDistancesVector is found and referred to as current speaker. In the next step, the current speaker and speaker identified from the speaker diarization module is checked. If both are same, then the next speaker's prediction will happen. For the next prediction, the predictSpeaker method is called. In this method the dot product of averagedDistancesVector and transitionMatrix is calculated and the result is stored as obtainedDotProductVector which is nothing but the list. Finally, the minimum value is calculated and returned as the next predicted speaker state.

6 Integration and Testing

6.1 Description of the Integration Modules

Output of module				Input to module			
Sr. No.	**Module**	**Output**	**Type**	**Module**	**Input**	**Type**	**Status**
1.	Speech to text	Conversations in the form of text	String	Data Processing and Dialogue creation	Text generated from speech to text module	String	Verified
2.	Data Processing and Dialogue creation	Distance score of each speaker state in the form of list	Float	Markov Chains	Distance score values	Float	Verified
3.	Speaker Diarization	Speaker labels in the form of list	Integer	Validator Module	Speaker labels	Integer	Verified
4.	Markov Chains	Predicted speaker state	Integer	Validator module	Predicted speaker state	Integer	Verified

6.2 Testing

Sr. No.	Test Case Title	Description	Expected Outcome
1.	Successful live Audio data Input	The live audio to the system is successful if the Audio data length and number of speakers are as specified.	Audio data Input must be successful.

Cont'd on following page

2.	Unsuccessful live Audio Input due to the wrong input.	Wrong input audio data, such as number of speakers are not as mentioned.	Prompt "wrong Audio Input error" and ask for input again.
3.	Successful Speaker Diarization process	The Speaker Diarization process is successful when labels (speaker identities) are generated.	The Speaker Diarization process must be successful.
4.	Unsuccessful Speaker Diarization process due to non-execution of Speaker Recognition and Speech Segmentation sub-process	Errors during the execution of Speaker Recognition and Speech Segmentation	Prompt "Errors while processing the VAD module" and exit.
5.	Unsuccessful Speaker Diarization process due to non-execution of d-vectors extraction sub-process	Errors during extraction of d-vector embeddings.	Prompt "Errors during d-vector extraction" and exit.

Cont'd on following page

6.	Unsuccessful Speaker Diarization process due to non-execution of Clustering sub-process	Errors during performing Offline Spectral Clustering	Prompt "Error during Clustering process" and exit.
7.	Successful Markov Chains process	The Markov Chains process is successful when the transition matrix is generated.	The Markov Chains process must be successful.
8.	Unsuccessful Markov Chains process due to null-valued transition matrix	The labels (speaker identities) are not interpreted correctly.	Prompt "Error while running Markov Chains process, please check the output of Speaker Diarization" and exit.
9.	Unsuccessful Markov Chains process due to zero-valued transition matrix	There is no output sequence of labels generated by speaker diarization.	Prompt "No speakers found in the audio data provided, thus the Markov Chains process is unsuccessful" and exit.

7 Performance Analysis

Initially, a text-independent approach was used. After data collection (audio files), VAD (VoiceActivityDetection) inspects the presence of voice in audio. The speech segments obtained by the previous step undergo Embedding extraction process using d-vector feature extraction. Then the speaker labels are found once embeddings of all the audio segments are extracted. This is done using OfflineSpectralClustering. The speakers labels are then fed to the markov chain module to get the flow of conversation. Later on a text dependent approach proved to be better, where after obtaining the speaker labels, a speech to text conversion module (that gives a data frame including the speech in the audio file corresponding to the speaker) is used to get distance matrices. These are then used to get the transition matrix and fed to the markov chain module to get the flow of conversation.

7.1 Evaluation Metrics

Validation Percentage

The validation percentage refers to the percentage of the number of times context-based speaker-states were identified for the current context i.e., validated per conversation window.

$$Validation\% = 100 \times \frac{NumberOfValidationsPerConversationWindow}{ConversationWindowLength}$$

Prediction Accuracy:

The Prediction accuracy refers to the percentage of the correct predictions out of the total predictions.

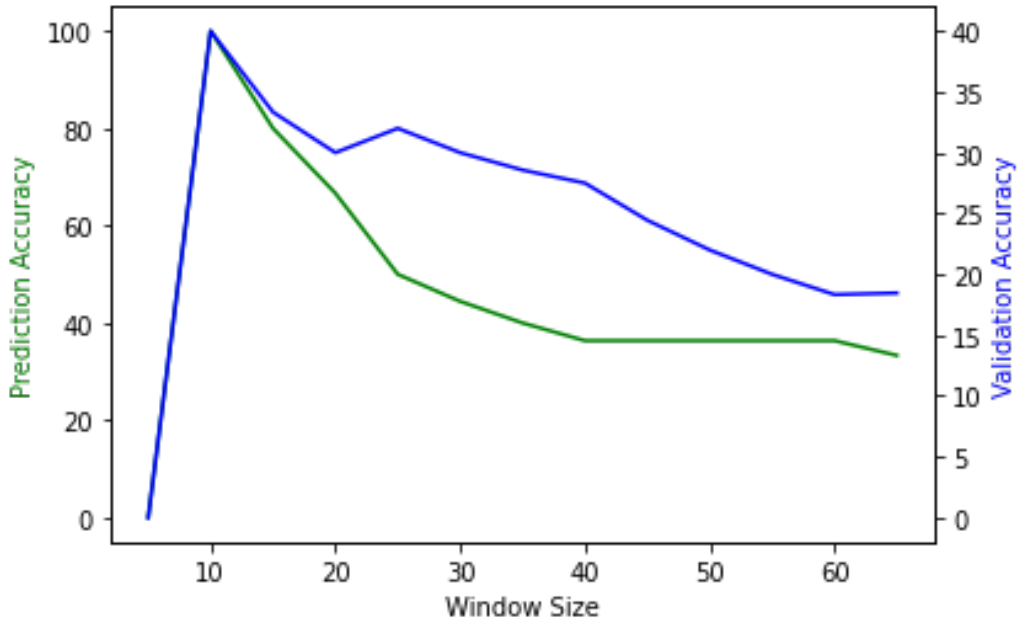
$$PredictionAccuracy = 100 \times \frac{NumberOfCorrectPredictions}{TotalPredictions}$$

7.2 Results

Audio Tran script	Number of Dialogues	Number of Speakers	Prediction Accuracy	Conversation Window Length	Validation Percentage
1	68	6	66.667	20	30
2	36	5	75	30	13.333
3	27	4	60	15	33.333
4	228	6	54.84	65	47.692
5	331	8	48.2	80	90

The table illustrates the evaluation results of four conversations, each evaluation performed on a live audio conversation having an average length of 2 minutes. The Audio Transcripts obtained were used to further evaluate the validation percentage and prediction accuracy. The results depicted in the table confirm that there is an average of 61% prediction accuracy.

The results also conclude that the validation percentages are low for lower number of dialogues. As the number of dialogues increases, the validation percentage per window increases. But the tradeoff between the validation accuracy and prediction accuracy occurs since, with the increase in number of dialogues, there is a rapid decrease in the accuracy of predictions while observing an increase in validation percentage.


Prediction Accuracy
Validation Accuracy
Window Size
100
80
60
40
20
0
40
35
30
25
20
15
10
5
0
10
20
30
40
50
60

8 Applications

1. **Chatbots:** In recent times, chatbots are used extensively by people for various reasons, to get assistance in solving a particular problem (and getting it done fast) or to search information from the internet and much more. But these chatbots require a lot of training data. This data is needed to be in a specific format like yaml/json and also it is needed to be hardcoded which can become hectic. Sometimes they may give 'out of context' answers. This system can help resolve this issue. It requires minimal training data and will make sure that the answers will not go out of context. Also, the replies from the bots would be faster.

2. **Robots:** Our system can be fitted in the robots as well. As our system predicts the future speaker and context, the robots will in advance, know how to react or reply to the people. Also, the robots will be able to talk with more people at a time. As they'll be knowing about the upcoming topic, the robots won't reply in an out of context way and the communication will be smoother and faster.

3. **Entertainment Industry:** In the entertainment industry, the dialogues and scripts are very important for a particular movie or series to be successful. Our system can be used here also. If some of the initial dialogues and characters are fed to the system, it can predict future dialogues and can generate a script for a movie or show. It would definitely be useful here.

4. **Conferences or youtube videos:** In conferencing, our system will be useful to know about the next speaker and what topic will be spoken by the next speaker.

9 Installation Guide and User Manual

For running the project, a python environment is necessary. The libraries required to be installed are as follows:

1. resemblyzer
2. pydub
3. spectralcluster
4. audioread
5. SpeechRecognition
6. librosa
7. pandas
8. numpy
9. from gensim.models import Word2Vec The command that can be used to install these libraries is:

pip install ¡library_name¿
The other internal libraries that are needed are:

1. soundfile
2. speech_recognition
3. from pydub import audio segment
4. from pydub.silence import split_on_silence
5. from pydub import AudioSegment
6. from resemblyzer import preprocess_wav, VoiceEncoder

10 Publication Details

Development of a Conversation State Prediction System

Sujay Uday Rittikar
SUJAYRIT20@GMAIL.COM
DKTE's Textile and Engineering Institute
Maharashtra, IN 416115

Abstract

With the evolution of the concept of Speaker diarization using LSTM, it's relatively easier to understand the speaker identities for specific segments of input audio stream data than manually tagging the data. With such a concept, it's highly desirable to consider the possibility of using the identified speaker identities to aid in predicting the future Speaker States in a conversation. In this study, the Markov Chains are used to identify and update the Speaker States for the next conversations between the same set of speakers, to enable identification of their states in the most natural and long conversations. The model is based on several audio samples from natural conversations of three or greater than three speakers in two datasets, with overall total error percentages for recognized states being lesser than or equal to 12%. The findings imply that the proposed extension to the Speaker diarization is effective to predict the states for a conversation.

Keywords: Conversational AI, Sound, Audio Signal Processing, Automatic Speech Recognition, Speaker Recognition, Conversation States, LSTM, Markov Chains

1. Introduction

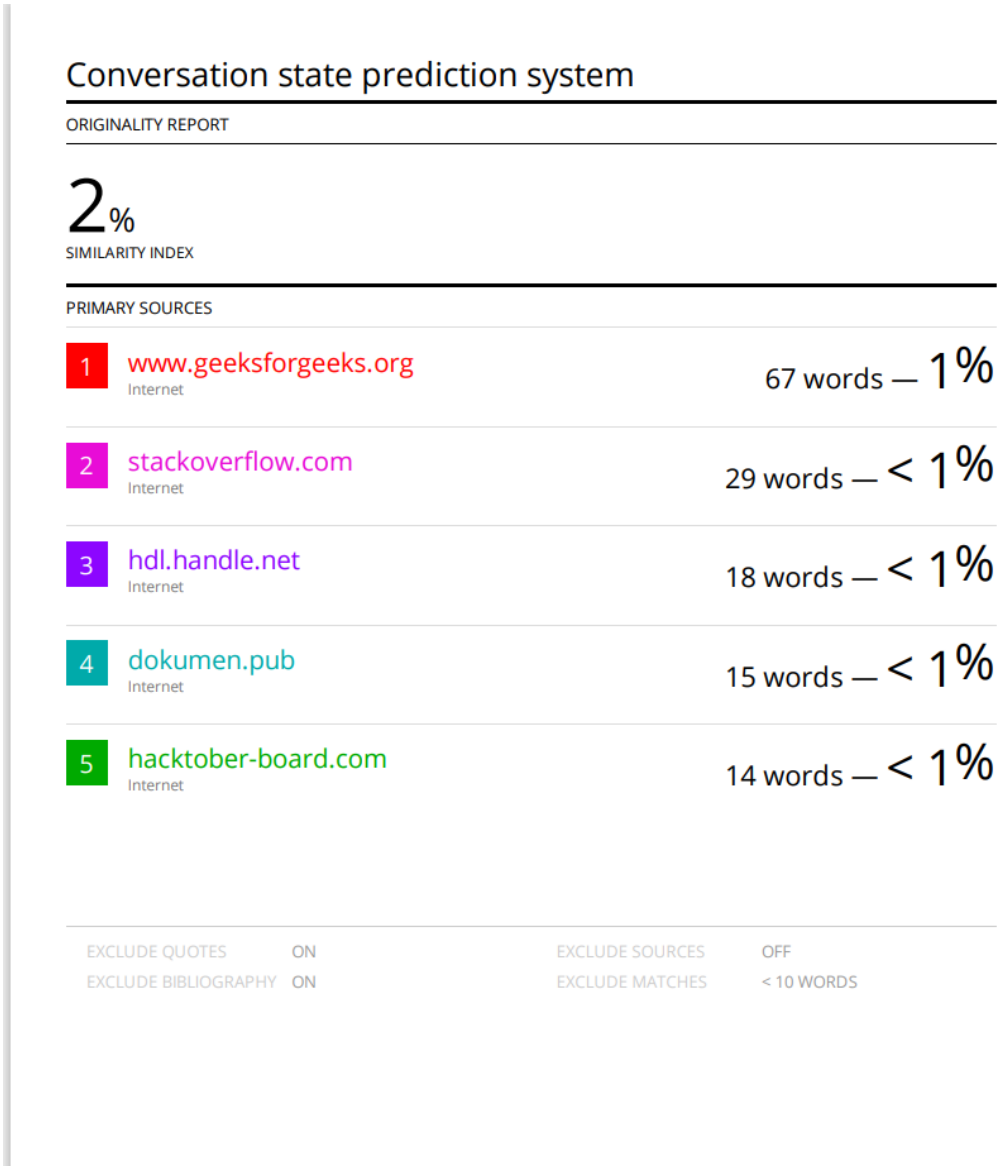
Speaker Recognition (Zhongxin Bai, et al., 2020) is a process to recognize the identity of a speaker as well as verify the identity. Although, the scopes of Speaker identification and Speaker verification are different and may merge based on the applications. Speaker identification has evolved with several text-dependent methods like HMM-Based Methods and text-independent methods such as the VQ-Based approach (Lei Z., et al., 2005) followed by classification using SVM. Speaker diarization(Quan Wang, Carlton Downey, Li Wan, Philip Andrew Mansfield, Ignacio Lopez Moreno, 2017) is a process used in order to perform Speaker identification using the process of partitioning an input audio stream into homogeneous segments according to the speaker identity. According to the findings of Speaker diarization, it's possible to identify the multiple speaker states in an audio stream.

The speaker diarization system is based on the use of Audio embeddings in form of text-independent d-vectors(Jung, J., et al., 2018) to train the LSTM-based (Sepp Hochreiter and J urgen Schmidhuber, 1997) speaker verification model and furthermore, combine the model with a spectral clustering algorithm to obtain the text-independent states. Thus, the study in the paper is based on the speech audio datasets having firmly speaker voices with minimum noise levels taking into consideration the possible noises to be detected as individual speaker states.

In recent years, Conversational AI (P. Kulkarni, et al., 2019) is used in various applications such as Chatbots and other virtual assistants. It works on the principles of Natural Language Processing (Khurana, et al., 2017) and is dependent on a set of steps essential to identify, retrieve and predict the characteristics of a conversation. The concepts of preprocessing NLP techniques such as tokenization and lemmatization, intent classification, entities extraction, featurizer, and response selector are based on various text-related and vector-based operations and, Machine Learning and Deep Learning-based models such as SVM, Bayesian Networks (Weissenbacher, Davy, 2006), and LSTM.

The Conversational AI richly deals with intent identification and its usage to recognize the flow of conversation and response selection. Although, with advancements in such AI systems

11 Plagiarism Report



Conversation state prediction system

ORIGINALITY REPORT

2%
SIMILARITY INDEX

PRIMARY SOURCES

1	www.geeksforgeeks.org Internet	67 words — 1%
2	stackoverflow.com Internet	29 words — < 1%
3	hdl.handle.net Internet	18 words — < 1%
4	dokumen.pub Internet	15 words — < 1%
5	hacktober-board.com Internet	14 words — < 1%

EXCLUDE QUOTES	ON	EXCLUDE SOURCES	OFF
EXCLUDE BIBLIOGRAPHY	ON	EXCLUDE MATCHES	< 10 WORDS

Figure 6: Plagiarism Report

12 References

(a) Lei, Zhenchun and Yang, Yingchun & Wu, Z.. (2005). Speaker Identification Using the VQ-Based Discriminative Kernels. 3546. 797-803. 10.1007/11527923_83.

(b) Jung, J., Heo, H., Yang, I., Yoon, S., Shim, H., & Yu, H. (2018). D-vector based speaker verification system using Raw Waveform CNN.

(c) Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/ neco.1997. 9.8.1735.

(d) P. Kulkarni, A. Mahabaleshwarkar, M. Kulkarni, N. Sirsikar and K. Gadgil, "Conversational AI: An Overview of Methodologies, Applications & Future Scope," 2019 5th International Conference On Computing, Communication, Control And Automation (ICCUBEA), 2019, pp. 1-7, doi: 10.1109/ICCUBEA47591.2019.9129347

(e) Khurana, Diksha & Koli, Aditya & Khatter, Kiran & Singh, Sukhdev. (2017). Natural Language Processing: State of The Art, Current Trends and Challenges

(f) Weissenbacher, Davy. (2006). Bayesian Network, a model for NLP?. 10.3115/1608974.1609007

(g) Sungdong Kim, Sohee Yang, Gyuwan Kim and Sang-Woo Lee. 2019. Efficient Dialogue State Tracking by Selectively Overwriting Memory. arXiv:1911.03906

(h) Javier Cebrián, Ramón Martínez, Natalia Rodríguez, Luis Fernando and D'Haro. 2021. Considerations on creating conversational agents for multiple environments and users. 10.1609/aaai.12007

(i) Shaver, Clark D. and Acken, John M., "A Brief Review of Speaker Recognition Technology" (2016).

(j) Noor Salwani Ibrahim and Dzati Athiar Ramli, "I-vector Extraction for Speaker Recognition Based on Dimensionality Reduction" (2018).

(k) Zhongxin Bai, Xiao-Lei Zhang. 2020. Speaker Recognition Based on Deep Learning: An Overview. arXiv:2012.00931arXiv:2004.01559

(l) Quan Wang, Carlton Downey, Li Wan, Philip Andrew Mansfield, Ignacio Lopez Moreno, Google Inc., USA, Carnegie Mellon University. 2017. Speaker Diarization with LSTM. arXiv:1710.10468

(m) Irfan, Bahar & Ramachandran, Aditi & Spaulding, Samuel & Glas, Dylan & Leite, Iolanda & Koay, Kheng. (2019). Personalization in Long-Term Human-Robot Interaction. 685-686. 10.1109/HRI.2019.8673076.

(n) Memon, Zojan & Jalbani, Dr & Shaikh, Mohsin & Memon, Rafia & Ali, Ahmed. (2018). Multi-Agent Communication System with Chatbots. Mehran University Research Journal of Engineering and Technology. 37. 663-672. 10.22581/muet1982.1803.19.

(o) Rittikar, Sujay Uday. "Development of a Conversation State Prediction System." (2021). https://arxiv.org/abs/2107.01462