NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES, KARACHI

PROJECT REPORT

Automation of Minutes of Meeting

Author:
Mehdi Raza K16-3904,
Hussain Yousuf K16-3805,
Shehryar Naeem K16-3950

Supervisor:
Dr. Muhammad Rafi

Submitted in the partial fulfilment of the requirements for the degree of Bachelor of Science

Department of
Computer Science

June 7, 2020

June 2020

National University of Computer & Emerging Sciences
(FAST-NUCES)

Project Supervisor	Dr. Muhammad Rafi
Project Manager	Mehdi Raza Rajani
Project Member	Shehryar Naeem
	Hussain Yousuf
Submission Date	June 7th, 2020

Supervisor:
Dr. Muhammad Rafi _____

Head of Department:
Dr. Muhammad Atif Tahir _____

Department of Computer Science

National University of Computer & Emerging Sciences (FAST-NU)

Main Campus, Karachi

ACKNOWLEDGEMENT

We are Thankful to Allah and His blessings.

We would like to express our utmost gratitude to our supervisor Dr. Muhammad Rafi, for his enthusiasm, patience, insightful comments, helpful information, practical advice, and unceasing ideas that have helped us tremendously at all times especially during research and development of this project. His immense knowledge, profound experience, and professional expertise in Data Science and Natural Language Processing has enabled us to complete this project successfully. Without his support and guidance, this project would not have been possible.

We are also thankful to our FYP Jury and FYP Committee for their guidance at each evaluation.

Document Information

Category	Information
Customer	FAST-NU
Project	Automation of Minutes of Meeting
Document	Project Report
Document Version	2.0
Status	Final Version
Author(s)	Mehdi Raza Rajani, Shehryar Naeem, Hussain Yousuf
Approver(s)	Dr. Muhammad Rafi
Issue Date	7th June 2020

Definition of Terms, Acronyms, and Abbreviations

Term	Description
MOM	Minutes of Meeting
EMA	Enterprise Meeting Assistant
NLP	Natural language Processing
DB	Database
ERD	Entity Relationship Diagram
AWS	Amazon Web Service
SSL	Secure Socket Layer

ABSTRACT

Minutes of the Meeting (MOM) is an AI-powered voice note application for automated note-taking and summary preparation for in-person meetings. MOM is a voice and document collaboration app that delivers Enterprise Meeting Assistant - EMA MOM's EMA will join your session and record the meeting, observe the speaker, and extract essential pointers from the attached documents. MOM will require some facts regarding Meeting. MOM will use these references to generate the Minutes of Meeting. Apart from summarizing the meeting, MOM clumps the meeting points according to the agenda of the meeting, and one can easily recall the update on a specific program. Also, it will assign tasks to attendees, which will help the manager to take follow up. MOM uses speech to text for producing the meeting transcript. National Language Processing and heuristics Techniques for highlighting the critical point from meeting transcript and attached documents, clustering the talk according to agenda and assigning tasks to attendees.

Contents

List of Figures

Chapter 1

Introduction

1.1 Need for Product

Every organization holds routinely scheduled assemblages as part of the everyday work. For an organization, it is a technique to keep everyone updated with work matters and a way of keeping track of individual and teamwork progress through a healthy discussion. The minutes of the meeting are recorded to make the meeting productive.

Meeting minutes keep an account of what was done or discussed about at an assemblage, including any decision shaped or move taken. Typically, a secretary or assistant records a meeting, but any appointed individual can do it. Meeting of the minutes comprises of date, time, and a venue of the meeting, the objective of the meeting. Minutes also include information of attendees and those who were absent and agenda items. Settlements that were made and that need to be performed are an integral part of the document. The meeting's report also includes deadlines and assigned personnel and tasks.

Although it may appear like record-keeping of assemblage minutes can be time taken, but for sure it will save an organization's time and money. Meeting minutes provide a transcribed document of what was considered and recognised at a meeting. This document will bring every attendee on the same page. If in case someone fails to attend the meeting, still it will be able to get aligned with the team. It also offers legal protection in some cases. The minutes of the meeting also acts as a measuring stick. With high-grade meeting minutes, the company makes sure everyone knows the things that were decided and what needs to be achieved by what date.

Technology is getting advance and have automated several tasks, so we aimed to digitalize the Minutes of the meeting generation. Formulating meeting minutes using technology can be a fast and easy way of recording resolutions. As a critical legal document, consideration should be taken to ensure the effectiveness of the decisions and legal compliance.

There are multiple benefits of using technology to generate meeting minutes. An intuitive document creation process which assists a business to turn its mind to relevant aspects of the decisions, thus helping with the creation of legally capable resolutions. A proficient drafting of templates, making the formulation effortless. Professional formatting and advancing clarity of the recommendations made. The solution is obtainable 24/7 for assistance. A quick and effortless solution, which makes the contemporaneous and reliable recording of the decisions more likely. The result is fine document - Minutes of the Meeting.

1.2 Benefits to Users

One challenge during all meetings is to create a summary of the meeting, usually, this is done manually by an employee designated for this task. This project aims to develop a product that will automatically create minutes of the meeting with annotated text from the discussions of the meeting requiring minimal human intervention and bootstrapping.

The project not only provides a summary but also highlights key features discussed, and clusters meeting points according to the agenda.

The project also allows users to share and collaborate the summary of the meeting via a user-friendly web-based interface, thereby providing maximum interoperability across multiple devices.

1.3 Gap Analysis with existing solution

There are several applications available like Otter.ai, Reason 8, Voicea dedicated to tasks that align with our project objective, but many of them simply lack the features which we have outlined and the bootstrapping process involved is simply too complex which defeats the purpose of an automation app. Furthermore, these applications are dedicated to a subset of the features which we have outlined, as a consequence their results are better than ours. Our objective was not to compete with these existing apps but to provide a viable product which encapsulates all the requirements and features desirable in an automated meeting application, understand how they interrelate with each other and see how they all work in conjunction to provide a user-friendly and complete user experience.
The further details of Existing Solution are added in Appendix **??**.

Chapter 2

Literature Review

2.1 Improved Keyword and Keyphrase Extraction from Meeting Transcripts

The paper [3] aims to extract important keywords from meeting transcripts, which may help users to quickly comprehend the gist of the discussion. The classifiers used for extraction here are Max Entropy Classifier and Support Vector Machine Classifier. After transcription, it subjects the document to preprocessing steps (stemming and removal of stop words) which removes irrelevant information. Max Entropy classifier fits well for this task as it does not presume that features are mutually exclusive of each other. Another advantage is that it starts with maximum entropy (least information), thus preventing bias. It trains SVM to map feature vectors to their respective category. Words belonging to categories deemed important are labelled as keywords, Key Phrases are a combination of keywords, they are identified before preprocessing. Compound words such as words containing hyphen or words in the quotation or in bold are considered important key phrases.

2.2 Rapid Keyphrase Extraction from Audio Transcripts of Video Lectures

In this paper [5], we use a supervised machine learning algorithm to train the domain-specific features. In addition, the decision tree classifier model is used to enhance the efficiency of the feature engineering and to extract relevant key phrases. Preprocessing steps includes
POS TAGGING: The noun tags(NN, NNS, NP), verb tags(VBZ, VBG, VBN) and adjective tags(JJ) are the most important tags. UPenn TreeBank II tagset is used for POS tagging.
STOP WORDS REMOVAL: The common words such as prepositions, articles, conjunctions, symbols, numbers, and non-alphanumerics that are irrelevant to the context are removed. eg (“the”, “off”, “is”, ‘ “a”, “an”, “in”).
N-GRAM EXTRACTION: The input string is tokenized as per the phrase boundaries. N-Gram takes a list of words and builds a set of consecutive word pairs.
STEMMING: Stemming or Lemmatization reduces a word to its base form.

The system uses a supervised machine learning algorithm to train and test

datasets using a decision tree classifier model. Gini index is used along with the ID3 algorithm for the classification and creation of a decision tree. The dataset is trained using the ID3 algorithm to produce a decision tree for identifying keyphrases. Gini index is used to calculate the possible split values of attributes. Experimentation shows that the Decision tree model overcomes the drawback of Naive Bayes classifier that assumes feature independence.

2.3 Text categorization with Support Vector Machines

The paper [6] discusses the working and benefits of commonly used text categorization technique i.e. Support Vector Machine. Firstly, the paper describes the steps of preparing data for categorization that mainly includes, tokenization, stemming, stop word removal, computing term, and inverse document frequency and selecting the words with minimum three occurrences in the corpus.
Then the paper defines the working of SVM which i.e. in this algorithm we plot each data item as a point in n-dimensional space (where n is the number of features - in this case, features are distinct words) with the value of each feature being the value of a particular coordinate. Then, the algorithm performs classification by determining the hyper-plane that distinguishes the two groups very well. The SVM finds the hypothesis which minimizes the bound on the true error. The paper also defends that SVM works well for text classification because its performance is based on the margin that separates the data nor on a number of features as the text has a huge number of features. Also, there are few irrelevant features so we can easily drop them. Moreover, as the document vectors are too sparse which supports SVM. Most importantly the most text categorization problems are linearly separable so SVM best fits in this case.

2.4 Speaker Diarization With LSTM

This paper [7] refers to the process of separating audio dialogues based on who spoke when, in essence, it classifies audio segments according to their speaker. The paper suggests the use of MFCC (Mel-frequency cepstrum) to convert audio segments into a power spectrum, a domain suitable for input for later stages. The input is then fed into recurrent (LSTM) multi-layer perceptron to produce vector embeddings called d-vectors which are then subjected to spectral clustering which was preferred over KMeans as it works well for non-Gaussian data.

The clustering is used to aggregate segments based on a common speaker. Here offline clustering is preferred which requires embeddings of all segments to be available before the clustering process, whereas the process of distinguishing between audio features is taken care of by comparing the intermediary d-vectors.

The comparison is made by finding the cosine similarity and checking it against a threshold.

2.5 Sentence Extraction Based Single Document Summarization

The paper [8] aims to generate a summary of the document based on extracting important sentences from the document. To locate sentence boundaries heuristics such as period (.) exclamation, question mark, and uppercase letters are employed to demarcate sentences. Features such as sentence number, sentence length, sentence relative position, presence of noun and verbs, word count, and term frequency of the sentence are vectorized and normalized on 0-1 scale. The score is the sum of the score of the words with sentence-level features in context. The word features are the combination of its relative position, length, frequency, and type. The selection of sentences based on its score also depends on already selected sentences. If multiple sentences are belonging to the same category then the one with the greater length is removed. The selected sentences are then inserted according to their previous relative order.

Chapter 3

Requirements Analysis

3.1 Functional Requirements

The user will be able to navigate to the web application and provide the meeting recording and metadata which includes the document presented in the meeting, date and time, venue of the meeting, agenda list of meeting, and attendees of the meeting. The data will be preprocessed and the audio transcriber will send an annotated transcript (speech to text and speech diarization) to the engine. Then the transcript will then be categorized according to the meeting's discussion and clustered according to the meeting's agenda. Then important features will be extracted and formatted to get a first version of the Minutes of Meeting document. Then the user will proofread the document and upon discrepancy, he will be able to make the changes accordingly. Upon satisfaction, he will be able to share and download the document.

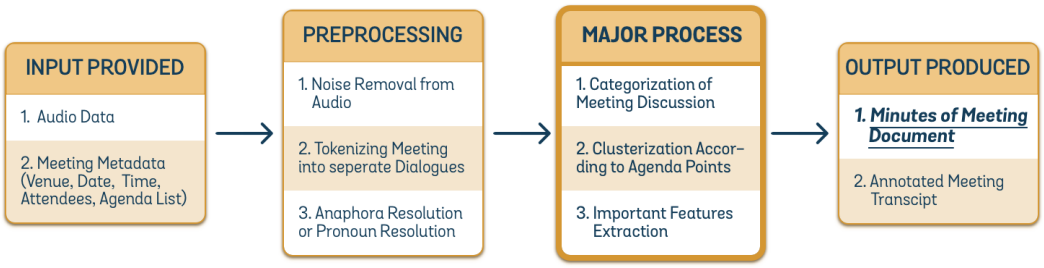
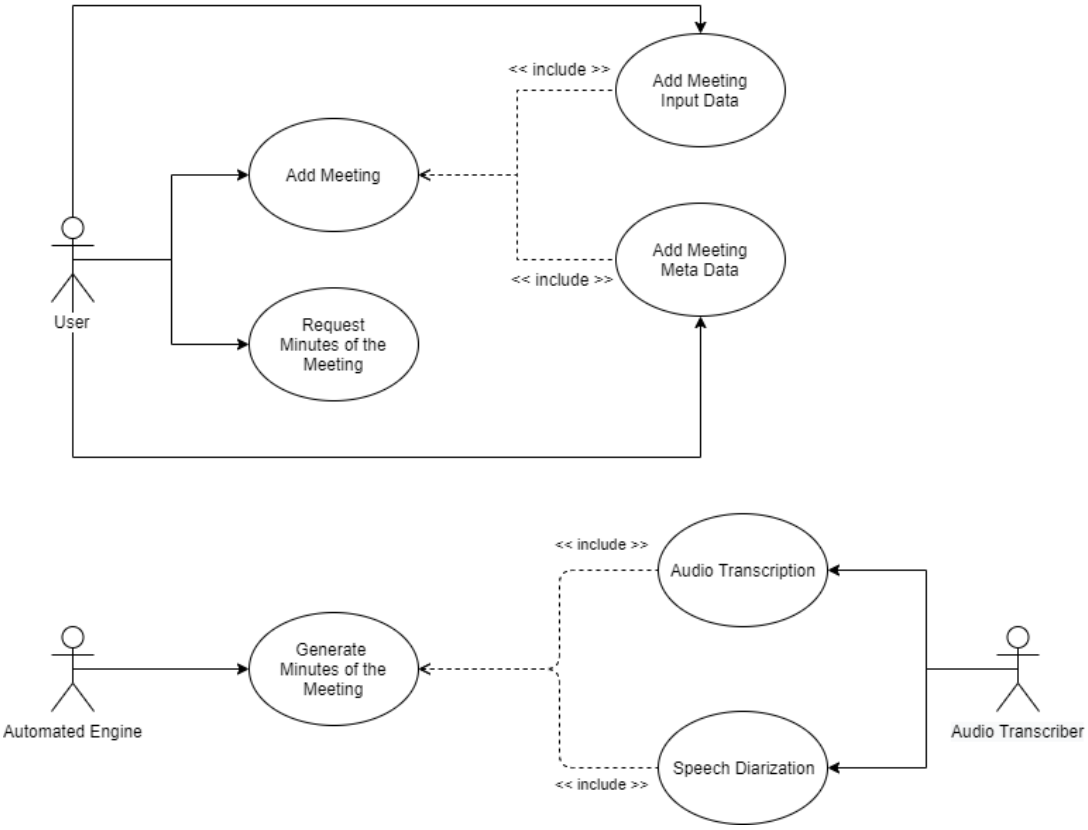


FIGURE 3.1: The Bird View of MOM Solution

3.1.1 Use Cases



The description of the Use Case Diagram is presented in the following subsection.

3.1.2 Use Case: Add Meeting

Use Case 1	**Add Meeting**
Use case Id:	User1
Actors:	User
Feature:	User wants to request the minutes of meeting document.
Pre-conditions:	• User is already registered. • User is signed in the system.

Main Success Scenario:

1. The user selects the date and time of the meeting.
   {Will be represented on screen.}
2. The user enters the venue of the meeting.
   {Will be represented on screen.}
3. The user selects the attendees from the list.
   {Will be represented on screen.}
4. The user will add the agenda list for the meeting.
   {Will be represented on screen.}
5. The user will add the attached document which will be presented in the meeting.
   {The progress bar will be shown to track the uploading document. Afterward, it will be represented on screen.}
6. The user can add the video of the meeting.
   {This is an optional step}
7. The user will add the audio of the meeting.
   {The progress bar will be shown to track the uploading audio. Afterward, it will be represented on screen.}

Alternate Scenarios:

2.a Venue not present:

- Venue will be added to database.

3.a Attendee name is present:

- User can select it.

3.b Attendee name is not present:

- Name will be added to database.
- User can select it.

Post Conditions:	• Users can now generate the request for generating the minutes of the meeting.
Use Case Cross referenced	User2

3.1.3 Use Case: Request Generation of Minutes of Meeting

Use Case 1	**Request Generation of Minutes of Meeting**
Use case Id:	User2
Actors:	User
Feature:	User wants to request the minutes of meeting document.
Pre-conditions:	• User is already registered. • User is signed in the system. • The meeting is added with all input and metadata.
Main Success Scenario: 1. The user will request the generation of minutes of the meeting. {Activity Indicator will be presented, to show that the process is running in the background. Afterward, the document will be shown on screen.}	
Post Conditions:	• Automation Engine will start processing and generate the document.
Use Case Cross referenced	• User1 • Server2

3.1.4 Use Case: Generate Minutes of the Meeting

Use Case 1	**Generate Minutes of the Meeting**
Use case Id:	Server1
Actors:	• Server (Automation Engine) • Audio Transcriber • User
Feature:	Automation Engine will generate Minutes of Meeting automatically.
Pre-conditions:	• The user has initiated document generation request of the legitimate meeting record.

Main Success Scenario:

1. The server will ask audio transcribers to do speech to text and speech diarization.
   {The server will wait until the annotated transcript is generated. Pass the resultant transcription for further processing. The annotated transcript will also be shown to user.}

2. The server will then perform categorization and clustering according to the meeting's agenda.
   {The server will wait until the categorization is performed. Then it will pass the categorized data to be clustered according to agenda points. Then the clustered data will be passed for further processing.}

3. The server will then extract important features from each agenda's data.
   {The server will until important features are extracted. Later it will pass for the final step.}

4. The formatted document will be generated.
   {The document will be shown to user.}

5. The user will check it confirm the document.
   {System will accommodate the changes done by user and show the final copy of the Minutes of Meeting document.}

Alternate Scenarios:

5.a No changes made by User:

- Add the document to database.
- Show the final version of document to user.

5.b User finds some discrepancy:

- Allow user to make changes.
- Add the document to database.
- Show the final version of document to user.

Use Case Cross referenced User2

3.2 Non-Functional Requirements

3.2.1 Performance Requirements

The web server should be able to handle multiple users in parallel, also the response time should not be too long and the server should provide periodic notification regarding the progress. The system expects that the user will have a standard Internet connection. Since the application depends on third parties, also the system assumes that third party services will also be consistent.

3.2.2 Safety Requirements

The web server and database are expected to have enough capabilities to prevent data breaches and to provide periodic backups. User profiles shall not be disclosed to any third party, and minimal pertaining information of the user will be kept.

3.2.3 Security Requirements

The application should communicate with a remote server using SSL. This assures the confidentiality, integrity, and non-repudiability of the payload. Furthermore, all data on the database shall be encrypted and third parties would not be allowed to persist or distribute data.

3.2.4 User Documentation

The application should include elaborate and intuitive controls that will enable the user to use the application without expert knowledge of the domain. However, instructions should be included in the application for completeness's sake.

Chapter 4

Methodology and Implementation

4.1 Process Diagram

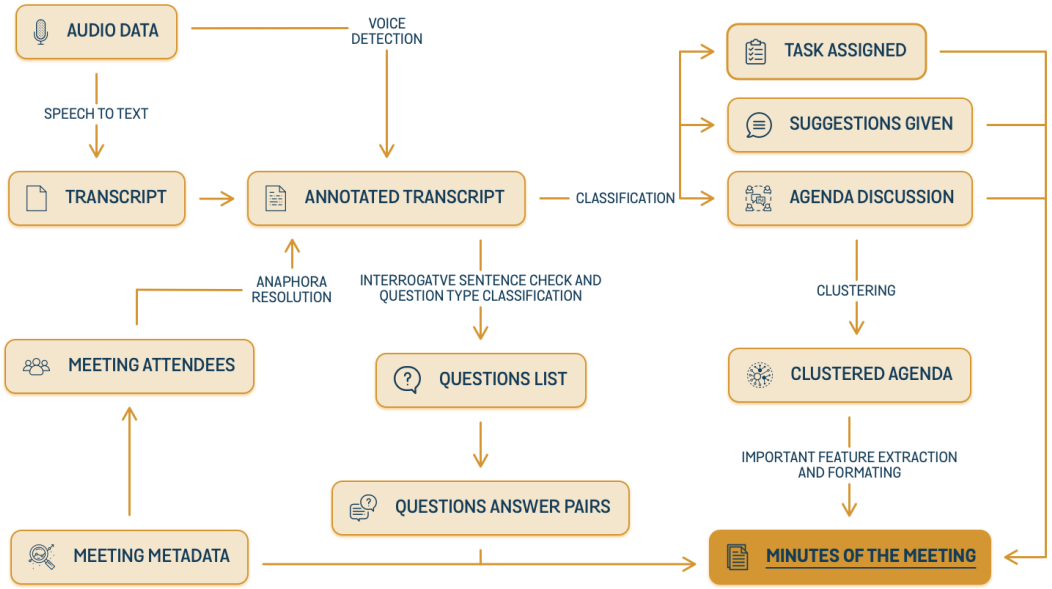


FIGURE 4.1: Process Diagram of MOM Solution

4.2 Meeting Data Collection

Meeting Audio is uploaded on the web interface which is used to generate the Minutes of Meeting along with some other data. Meeting metadata like venue, time, agenda list, attendees, chairperson are also uploaded to the application. These are used as auxiliaries for the process.

4.3 Transcription and Speaker Diarization

The uploaded audio data is then processed by AWS Cloud Service which uses Audio Transcribe to perform transcription and speaker diarization modules to produce an annotated transcription of the audio feed.

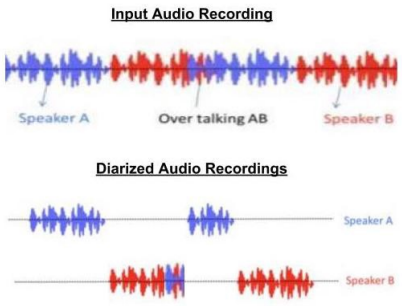


FIGURE 4.2: Speaker Diarization

4.4 Anaphora Resolution

To avoid repetition, often during conversations pronouns are used to refer back to a word used earlier in a conversation, Anaphora resolution resolves what a pronoun in an earlier context referred to, this becomes important as we separate and cluster individual meetings dialogues. Without this resolution, the separated content would lack context. We used Stanford CoreNLP Coref Annotator to resolve this issue.

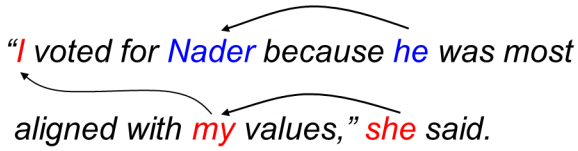


FIGURE 4.3: Anaphora Resolution Example

4.5 Clustering of Meetings Discussion according to Agenda

Before doing clustering we tokenized the sentences.

4.5.1 Tokenization of Sentences

The document must be parsed to eliminate certain words for later stages, this method is known as tokenization. The lexemes may be alphanumerics or special characters. This process is performed by positioning word boundaries.

The following tokenizers were considered during development:

1. **Term Frequency–Inverse Document Frequency:**

   Tf-idf is a mathematical measure that is dedicated to determining how relevant a term is to a document in a corpus. The score allotted to every lexeme depends on the frequency of the term in the document and the frequency in the entire corpora. The words that are present in every document with higher frequency will be having less score. On the contrary, the words that are present in some specific document will be having a higher score.

2. **Count Vectorizer :**

   Frequency of occurring words in a document is the score of count vectorization.

3. **The Universal Sentence Encoder (USE):**

   USE is a pre-trained model provided by tensor-flow which transforms the text into 128-dimensional vectors. The accuracy of the model is directly proportional to the length of the string. USE is trained on diversified datasets and is generally recommended as a tokenizer for NLP tasks.

4.5.2 Clustering

The following algorithms were used to cluster meeting points according to the agenda:

1. K-Means Algorithm
2. Agglomerative Clustering
3. Gaussian Mixture Model
4. Balanced Iterative Reducing and Clustering using Hierarchies, or BIRCH

Data used in clustering was taken from the following link:
https://www.opm.gov/policy-data-oversight/pay-leave/pay-systems/federal-wage-system/#url=FPRAC

4.5.3 Selection of Clustering Algorithm

Universal Sentence Encoder (USE) was used to perform tokenization, generated vector embeddings (128 x 1 dimension) for each sentence were then subjected to ***Principal Component Analysis (PCA)*** which was used to select top features. The top features of a sentence vector were then clustered using the above-listed algorithms. The topic keywords for each cluster were found using ***Latent Dirichlet Allocation Model (LDA)***. Then we used ***Stochastic Gradient Descent*** and cross-validation to calculate the mean score.

Following were the results for the experiment.

Clustering Algorithm	Purity Score
K-Means	90.223
Agglomerative	**91.014**
Gaussian Mixture	88.682
Birch	89.383

Of all the above clustering algorithms Agglomerative Clustering performed best as it had the highest mean score.

All of the relevant code and sample clustering plots are attached in Appendix B.

4.6 Classification of Meetings Points

The process involves segregating sentences based on their semantics, the sentence may be suggestive, interrogative, imperative or declarative.

First of all we identified that a sentence is interrogative or not by using Stanford CoreNLP Sentence parser.

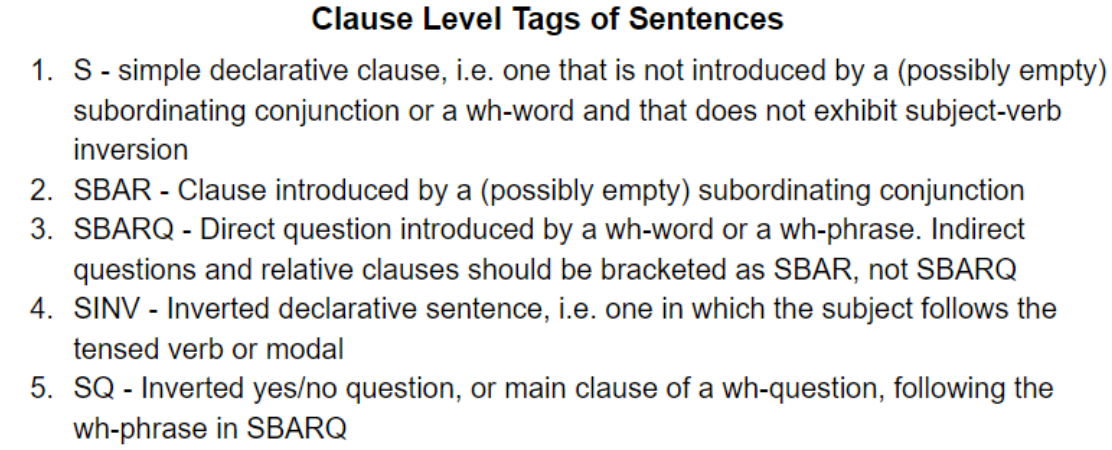


FIGURE 4.4: Stanford NLP Sentence Types

If the parser marks a sentence as SQ or SBARQ then the it is furter classified as *Affirmation (YES or NO), What, When, Who, Unknown*. On the contrary, if the sentence non-interrogative so it is classified as *Task Assigned, Suggestion or Normal*.

```
def classify (sentence):
    if isSentenceInterrogative (sentence):
        sentenceClass = interrogativeModel,predict (sentence)
    else:
        sentenceClass = sentenceModel.predict (sentence)
    return sentenceClass
```

4.6.1 Sentence Classification

In order to classify a sentence as *Task Assigned, Suggestion or Normal*, several supervised classifiers were tested, some of which are:

1. Logistic Regression
2. Support Vector Machine (SVM) Classifiers
3. Gradient Boosting Ensemble Tree Classifier
4. Hybrid Neural Network Classifiers

Of all the above classification algorithms Hybrid Neural Network performed best as it had the highest f1 score.

The confusion matrix and classification report for each classifier is attached

in Appendix A.

Data for training the classification model was taken from multiple sources which is as following;

1. Normal and Suggestive sentences were extracted and compiled from the following link [9]
   https://github.com/Semeval2019Task9/Subtask-A
2. Task Assigned sentences were compiled and extracted from the following link [10]
   https://github.com/RevRameshkumar/EPADataset

Sentence Type	Sentence Count
Normal	7457
Suggestive	2468
Task Assigned	6734
Total	**16659**

4.6.2 Interrogative Classification

In order to classify an interrogative sentence as *Affirmation (YES or NO), What, When, Who, Unknown*, Support Vector Machine classifiers was tested and its confusion matrix and classification report is attached in Appendix A.

Data for training the Interrogative sentence classification model was taken from the following link; https://github.com/mohitRohatgi/what_who_when

Interrogative Sentence Type	Sentence Count
What	609
Who	402
Affirmation	104
Unknown	272
When	96
Total	**1483**

4.7 Extracting Important Features from Discussion

The following features were extracted:

- Sentences with “no” words
- Sentences with statics
- Sentences with date
- The sentiment of a sentence and its score

We predominantly used regular expressions to possibly capture all permutations of the above features. Regular expressions proved to be extensible and adequate for our purpose. Context-free grammar was not a suitable choice as natural language does not always follow structure but can still be meaningful, furthermore Context-free grammar takes exponential time

$$O(n^3)$$

to compute for natural languages. Also Type 0 and Type 1 grammars would require a Turing machine, making them infeasible in practice.

The code snippets of each feature extractions is attached in Appendix **??**.

4.8 Text Summarization

We have used the TextRank algorithm for this purpose. TextRank is an extractive and unsupervised text summarization method. Extractive summarization relies on selecting certain tokens from a document based on the rank and value of that phrase. Then it produces a summary by combining all of the valuable sentences.

The working of TextRank Algorithm is as follows:

1. Merge all the strings that appeared in the document.
2. Tokenize the concatenated string into proper sentences.
3. Transform the tokenised sentences into vectors
4. Compute the similarity score of each vector
5. Convert the similarity matrix into a graph. The network graph edges are similarity rate and vertices are tokenized sentences.
6. Lastly, a specified percentage of sentences were marked an important according to their values.

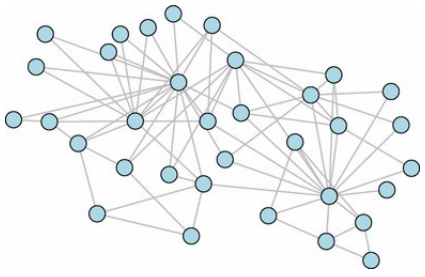


FIGURE 4.5: Network Graph Example

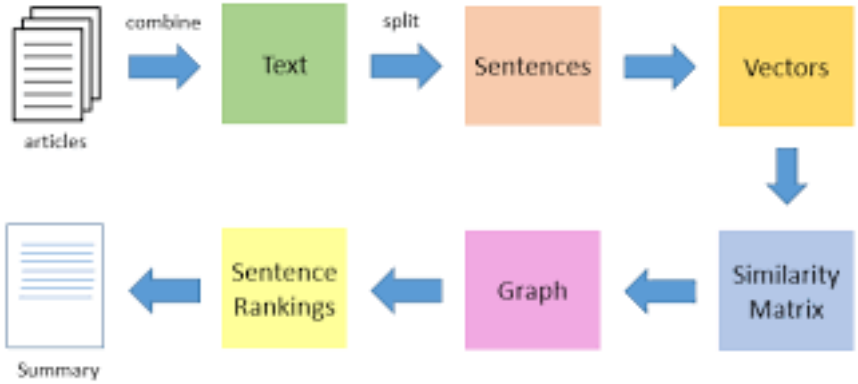


FIGURE 4.6: TextRank Algorithm Working

4.9 User Decision

After Classification, Clustering, Feature Extraction, and Text Summarization the result is presented on the web interface where the user can select and modify the results, upon satisfaction he may download and collaborate.

Chapter 5

System Design

5.1 Rationale for System Design

To allow maximum devices to leverage the provided services the user interface is a web-based, this not only allows for high maintainability but also enables upgrades to be pushed out seamlessly as a user does not have to manually upgrade rather the process is transparent to the user, furthermore the web-based interface allows multiple users to collaborate interactively.

Python and JavaScript languages are being used to develop back-end and front-end of the system, respectively. Both languages are object-oriented and allow a plethora of software engineering designs, tools and patterns to be incorporated in the application. Community support and documentation for both languages are mature to account for best practices and hindrances.

Popular NOSQL database, MongoDB is being used for persistence of data, the DBMS has gained a reputation for being reliable, fast and robust.

NodeJs platform will handle the application logic, the environment is fully asynchronous and provides a non blocking api, this mitigates synchronization issues which would be inevitable in threaded based environments.

Amazon Web Service (AWS) is used to transcribe meeting audio into transcript and perform Speech Diarization which is used as an input for our solution.

Stanford NLP is one of the famous group working on Natural Language Processing and the have provided number of parser for commonly occurring task. We have used Coref Parser to perform Pronoun Resolution task, Sentence-type Parser to identify interrogative sentences and Sentiment Parser in feature extraction.

Tensor flow is a renowned library for Deep Learning in Python. We have used Universal Sentence Encoder (USE) in our system. USE is a pre-trained embedding model for English Language.

5.2 System Architecture

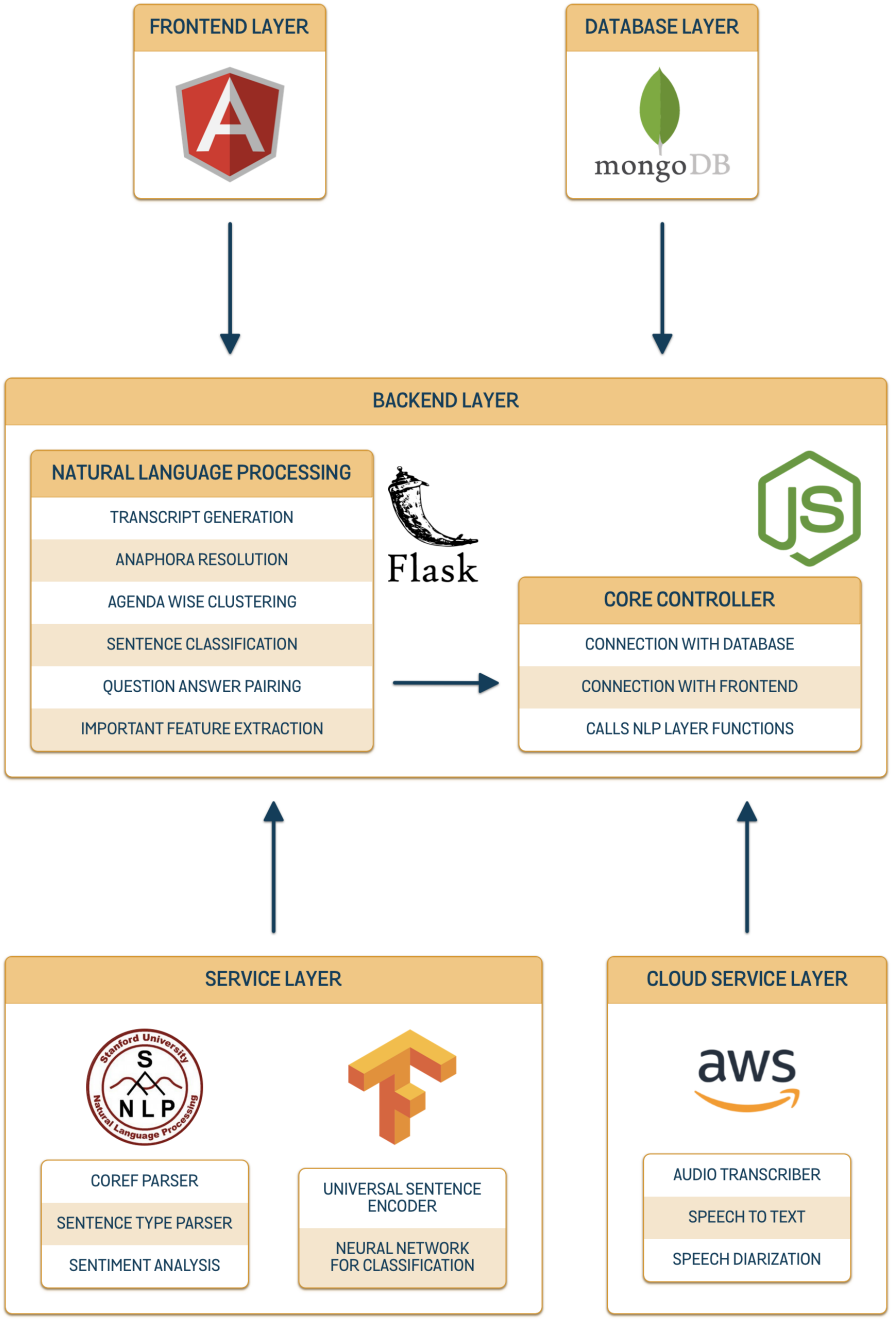


FIGURE 5.1: The Detailed System Architecture of MOM Solution

Chapter 6

Issues Faced

Following are the issues which we faced during this project.

- Dataset used to train was too heterogeneous, different data sources were used, hence some of our techniques performed poorly in some instances, therefore we selected the most generally applicable techniques.
- As data came from different formats we had to extract data individually from each format.
- Textual data was found in abundance. Models were primarily trained on textual data as opposed to spoken, as many iterations and bulk of data were required.
- KMeans did not work well on non-gaussian data.
- As agglomerative was used, we had to perform Topic Modelling in addition.
- Agenda list provided in some cases did not reflect the discussed agendas in the meeting.
- Agenda list (the dataset we used to perform agenda clustering) used of too many abbreviations.
- In many cases, the agenda list focused too much on company-specific jargon hence it was difficult to produce a general model.
- Agenda list would often consist of numerical values, hence trivial approaches were used against such agendas as there was a lack of context.
- The local accent was a major problem during speech transcription.
- For unsupervised processes like clustering, we had no ground truth values to compare with and to assess how well the model performed.
- There was no Dataset that maps a meeting transcription to the minutes therefore we were not able to compute the combined accuracy of our system.

Chapter 7

Testing Plan

7.1 Purpose of the Test Plan Document

The Test Plan document reports and tracks the essential information required to effectively represent the procedure to be used in the project's outcome testing. The Test Plan document is produced during the Planning Phase of the Project. This record is designed for Project Supervisor, Manager and the team. Some divisions of this document are occasionally shared with the client and other stakeholders whose consent is necessitated.

7.2 Environmental Needs

1. Personal Computer with at least 8 GB RAM and 256 GB Hard Disk.
2. Stanford CoreNLP Server running at Local Machine.
3. Python 3.7 (minimum) installed.
4. MongoDB 4.0 (minimum) installed.
5. NodeJS 13.0.0 (minimum) installed.
6. An Amazon EC2 instance.

7.3 Validation Testing

7.3.1 Items to be Tested

Items to Test	Test Description	Test Date	Test Responsibility
UI Integration with API's	As the UI was developed separately, the integration between Frontend and Backend of our system.	5th-Jun-2020	Alpha Testing by Developers acting as users.

7.3.2 Test Deliverable

A test summary showing all the conducted tests and their respective outcomes. Collectively reinforcing a working and validated system.

7.3.3 Test Cases

UI Integration with API's

As the UI was developed separately, the integration between Frontend and Backend of our system.

No.	Steps	Expected Results	Actual Results	Pass/Fail
1	Scripts and other images testing, to investigate whether all the resources, materials, scripts are in place and correctly associated.	All the assets would be accurately integrated with their respective handlers and no reference errors are reported on test play.	All the assets are properly integrated and no errors are reported on test play.	Pass
2	Navigation between tabs on button clicks and events is tested.	All the navigations would be performing properly without giving any reference errors.	All the navigations perform properly without giving any reference errors.	Pass

7.4 Functional Testing

7.4.1 Items to be Tested

Items to Test	Test Description	Test Date	Test Responsibility
Signup	User credentials would be fed into the system in order to sign up for a new user account.	10-Feb-2020	Developers
Login	User credentials would be fed into the system to log the user into the application.	10-Feb-2020	Developers
Add Meeting and Upload Meeting Recording	User can add meeting details that includes time, date, venue, attendees and agenda. Then the User can upload the meeting audio recording in mp3 format so that it can be processed in future.	20-Mar-2020	Developers
Minutes of Meeting Generation	After the transcription of audio is completed. User will follow to multiple iterative steps and at the end Minutes of the Meeting will be generated.	5th-Jun-2020	Developers

7.4.2 Test Pass / Fail Criteria

Since the tests are performed internally by developers, the passing criterion is based on binary decisions, that is, a test item is regarded PASSED if it completely matches the expected results.

7.4.3 Test Deliverable

A test summary showing all the conducted tests and their respective outcomes. Collectively reinforcing a working and validated system.

7.4.4 Test Cases

Signup

User credentials would be fed into the system in order to sign up for a new user account.

No.	Steps	Expected Results	Actual Results	Pass/Fail
1	User enters the URL in field.	The Login Page will be loaded.	The Login Page will be loaded.	Pass
2	User clicks on Create an Account..	User will be shown fields to enter the essential information.	User will be shown fields to enter the essential information.	Pass
3	Correct credentials fed into the system for a new user account that does not previously exist.	A new account would be created and user would be automatically logged into the application.	A new account was created successfully and user was logged into the application.	Pass

Login

User credentials would be fed into the system to log the user into the application.

No.	Steps	Expected Results	Actual Results	Pass/Fail
1	User enters the URL in field.	The Login Page will be loaded.	The Login Page will be loaded.	Pass
2	Correct credentials fed into the system for the user login.	The user would be logged into the application.	The user was logged into the application.	Pass

Add Meeting and Upload Meeting Recording

User can add meeting details that includes time, date, venue, attendees and agenda. Then the User can upload the meeting audio recording in mp3 format so that it can be processed in future,

No.	Steps	Expected Results	Actual Results	Pass/Fail
1	Users clicks the New Meeting in Left Navigation Bar.	Meeting Creation Page will be loaded.	Meeting Creation Page will be loaded.	Pass
2	User adds tittle, venue, date, start and end time of meeting. Then click NEXT button on the bottom.	New tab will be Opened to add Agendas.	New tab will be Opened to add Agendas.	Pass

No.	Steps	Expected Results	Actual Results	Pass/Fail
3	User will add agenda by clicking the Plus Button on the Tab. Then click NEXT button on the bottom.	New tab will be Opened to select attendees.	New tab will be Opened to select attendees.	Pass
4	User will select the attendees. He can also add new attendee in the meeting. Then click NEXT button on the bottom.	New tab will be Opened to select audio file.	New tab will be Opened to select audio file.	Pass
5	User will select the audio file. Then click UPLOAD button on the bottom.	Pop over will be open which will inform the user about the progress of transcription.	Pop over will be open which will inform the user about the progress of transcription.	Pass

Minutes of Meeting Generation

After the transcription of audio is completed. User will follow to multiple iterative steps and at the end Minutes of the Meeting will be generated.

No.	Steps	Expected Results	Actual Results	Pass/Fail
1	Users clicks the Completed Meeting.	A screen will be shown that contains the list of all the meeting that are completed and available for Minutes Generation.	A screen will be shown that contains the list of all the meeting that are completed and available for Minutes Generation.	Pass
2	User will click on the relevant of meeting.	New tab will be Opened that contains the Meeting Transcription.	New tab will be Opened that contains the Meeting Transcription.	Pass
3	User can read the transcript. Then will click Next to Proceed.	New tab will be Opened that contains the Meeting Annotated Transcription.	New tab will be Opened that contains the Meeting Annotated Transcription.	Pass

No.	Steps	Expected Results	Actual Results	Pass/Fail
4	The process will continue until the Rough Draft of Meeting's Minutes are Generated.	New tab will be Opened that contains the Meeting Minutes.	New tab will be Opened that contains the Meeting Minutes.	Pass
5	User can Export the Meeting Minutes. By Clicking Export to PDF.	A PDF File will be generated in a new tab of Browser.	A PDF File will be generated in a new tab of Browser.	Pass

Chapter 8

Conclusion and Future Work

Every enterprise, either small or large, organize meetings daily. The minutes of the meeting method is the way to note down the key insights of the meeting, including the previous development and proposals for an agenda. The solution made minimize the exploitation of human energy in doing an excessive amount of work daily for creating minutes of the meeting. The solution computed the minutes by first grouping the discussion according to the agendas of the meeting, then each question is identified as Imperative, Declarative, Suggestive, or Interrogative. Lastly, all the important sentences were marked and presented as Minutes of the Meeting. Due to the lack of the data, the accuracy was not perfect so we involved human error correction after each step to validate the result.

A stream of future directions can be deduced from our work which mainly belongs to two areas; Improving the accuracy of our solution and Improving the useability and functionality of our solution.

To enhance the accuracy we first require to build a dataset that contains the audio recording of the meeting, transcription, and final minutes. Also, a dataset should be developed that contains spoken English language sentences with their types so that accuracy of Classification can significantly be increased. We can also change our focus from audio data to video data of meetings from which we can identify some additionals features that will improve the system. On the Parallel end, we can integrate our solution with Calendar to schedule the meetings. Also, we can add search functionality to track the progress on any issue. Additionally, sentences identified as a task assigned should make a task for a relevant and ask for feedback automatically after a certain time.

Nevertheless, the solution developed is fair enough to marked as a first step to generate Minutes of the Meeting with any human intervene.

Chapter 9

References

Bibliography

[1] H. Hung and S. O. Ba, "Speech/non-speech detection in meetings from automatically extracted low resolution visual features", Idiap, Tech. Rep., 2009.

[2] H.-J. Song, J. Go, S.-B. Park, and S.-Y. Park, "A just-in-time keyword extraction from meeting transcripts", in *Proceedings of the 2013 conference of the north American chapter of the association for computational linguistics: human language technologies*, 2013, pp. 888–896.

[3] J. Sheeba and K Vivekanandan, "Improved keyword and keyphrase extraction from meeting transcripts", *International Journal of Computer Applications*, vol. 52, no. 13, 2012.

[4] F. Liu, F. Liu, and Y. Liu, "A supervised framework for keyword extraction from meeting transcripts", *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 3, pp. 538–548, 2010.

[5] A. Balagopalan, L. L. Balasubramanian, V. Balasubramanian, N. Chandrasekharan, and A. Damodar, "Automatic keyphrase extraction and segmentation of video lectures", in *2012 IEEE International conference on technology enhanced education (ICTEE)*, IEEE, 2012, pp. 1–10.

[6] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features", in *European conference on machine learning*, Springer, 1998, pp. 137–142.

[7] Q. Wang, C. Downey, L. Wan, P. A. Mansfield, and I. L. Moreno, "Speaker diarization with lstm", in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2018, pp. 5239–5243.

[8] J Jagadeesh, P. Pingali, and V. Varma, "Sentence extraction based single document summarization", *International Institute of Information Technology, Hyderabad, India*, vol. 5, 2005.

[9] S. Negi, T. Daudert, and P. Buitelaar, "Semeval-2019 task 9: Suggestion mining from online reviews and forums", in *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019)*, 2019, pp. 783–883.

[10] R. Rameshkumar, P. Bailey, A. Jha, and C. Quirk, "Assigning people to tasks identified in email: The epa dataset for addressee tagging for detected task intent", in *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, 2018, pp. 28–32.

Services, Tools, and Libraries Used

- https://aws.amazon.com/transcribe/
- https://cloud.google.com/dialogflow/
- https://stanfordnlp.github.io/CoreNLP/
- https://spacy.io/
- https://www.nltk.org/
- https://scikit-learn.org/stable/
- https://tfhub.dev/google/universal-sentence-encoder/4
- https://flask.palletsprojects.com/en/1.1.x/
- https://nodejs.org/en/docs/
- https://angular.io/docs
- https://docs.mongodb.com/

Related Applications

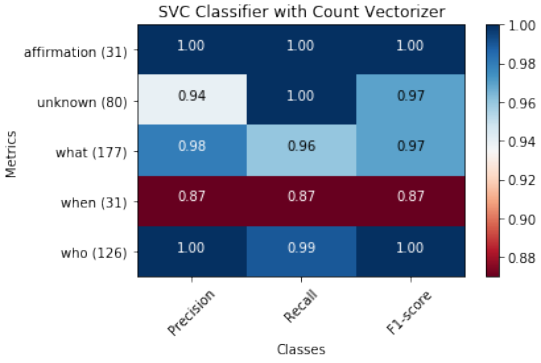
- https://reason8.ai/
- https://otter.ai/
- https://www.voicea.com/

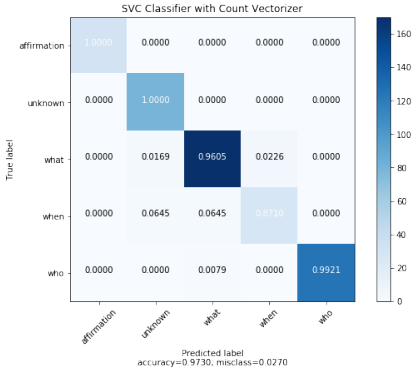
Appendix A

Sentence Classification

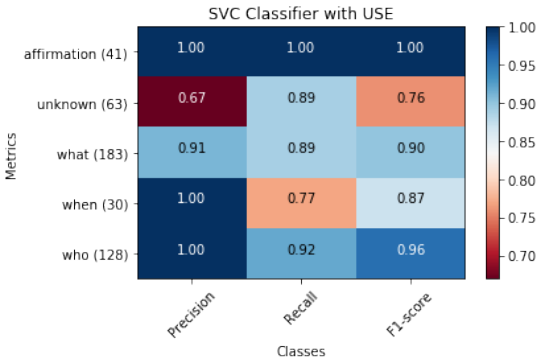
A.1 Interrogative Sentence Classification

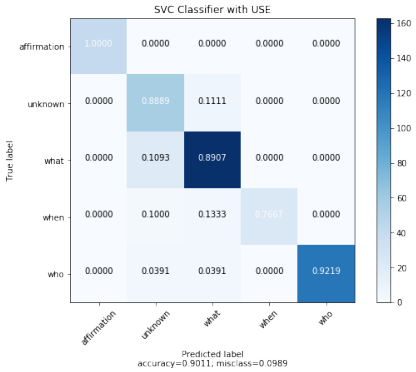
A.1.1 Support Vector Machine Classifier With Count Vectorization





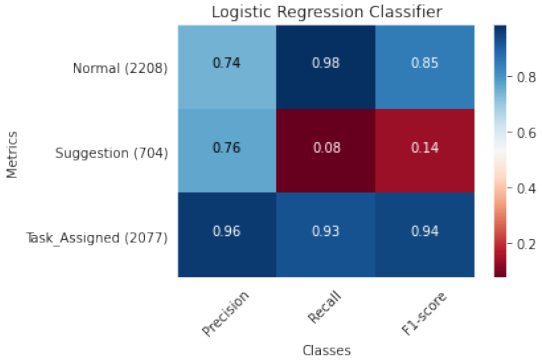
A.1.2 Support Vector Machine Classifier With Universal Sentence Encoding

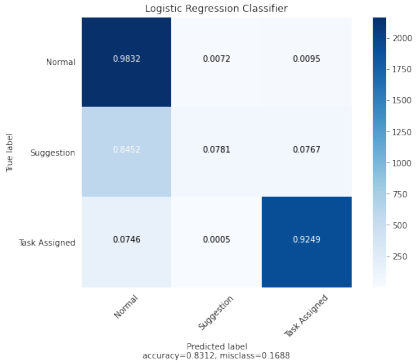




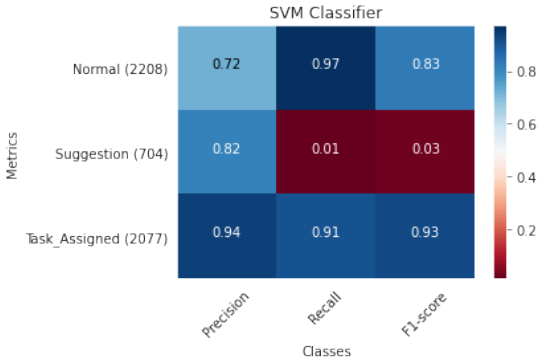
A.2 Non-interrogative Sentence Classification

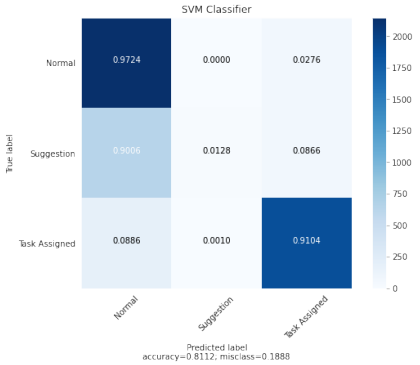
A.2.1 Logistic Regression



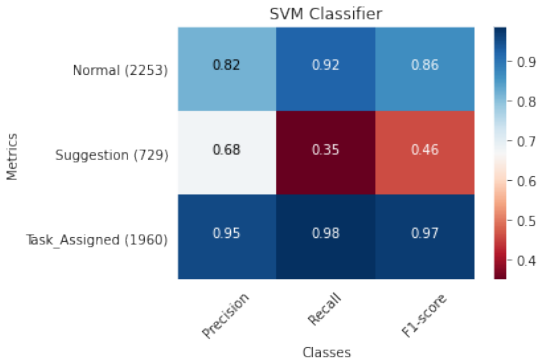


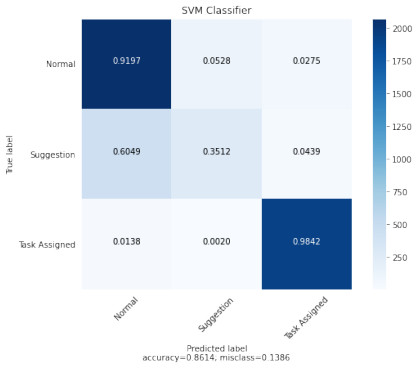
A.2.2 Support Vector Machine Classifier With Count Vectorization



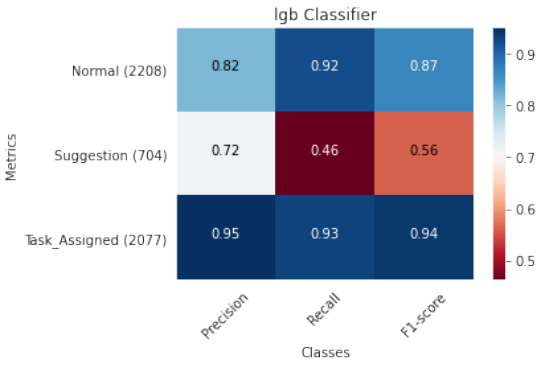


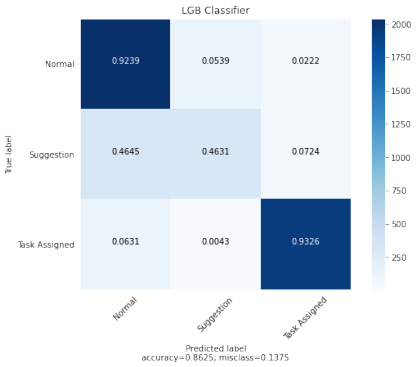
A.2.3 Support Vector Machine Classifier With Universal Sentence Encoding



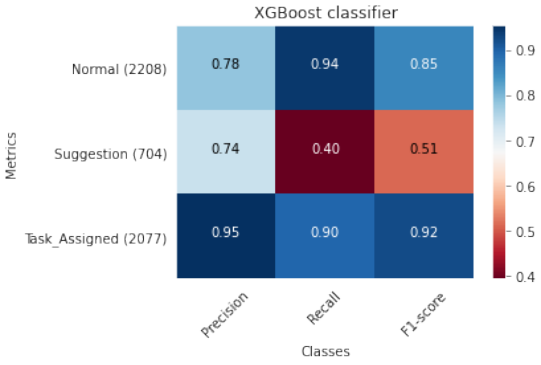


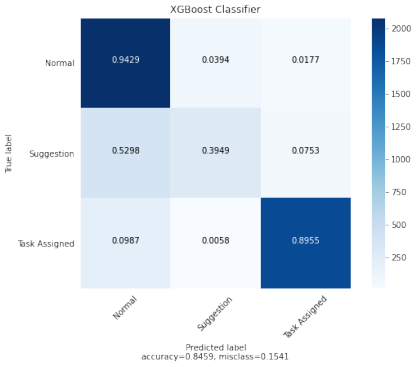
A.2.4 Light Gradient Boosting Decision Tree Ensemble Model



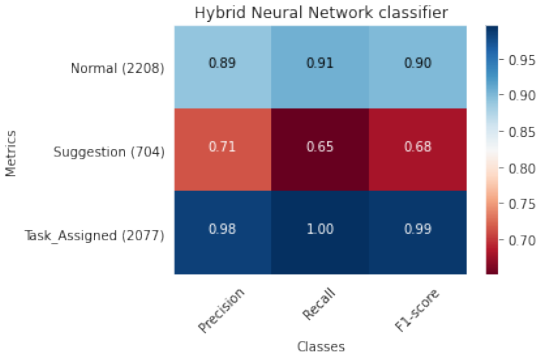


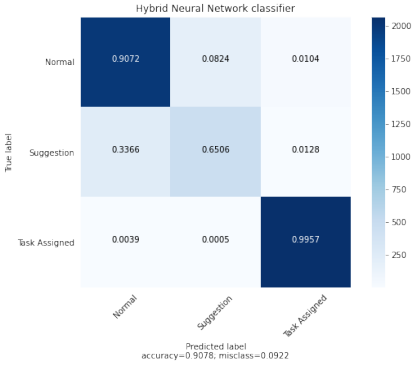
A.2.5 eXtreme Gradient Boosting Decision Tree Ensemble Model



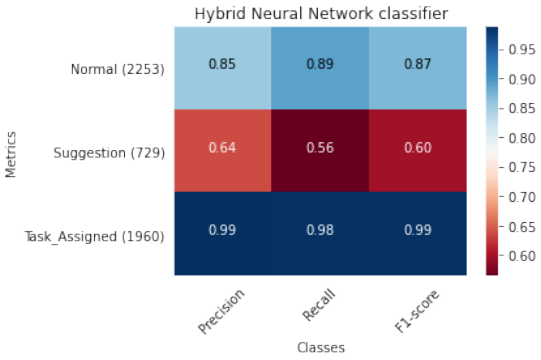


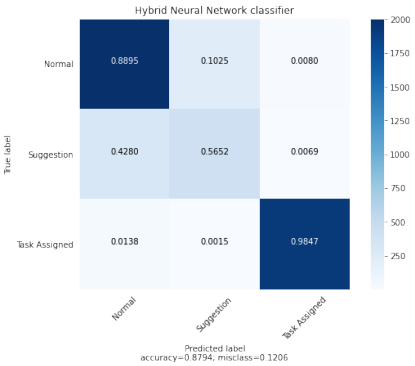
A.2.6 Hybrid Neural Network





A.2.7 Hybrid Neural Network with Universal Sentence Encoder

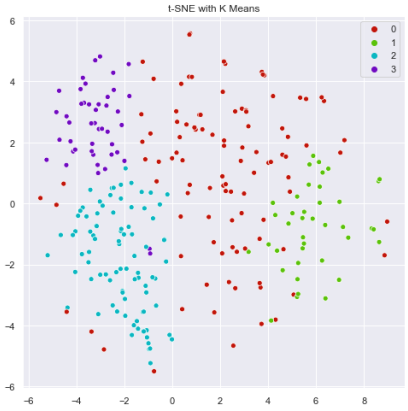


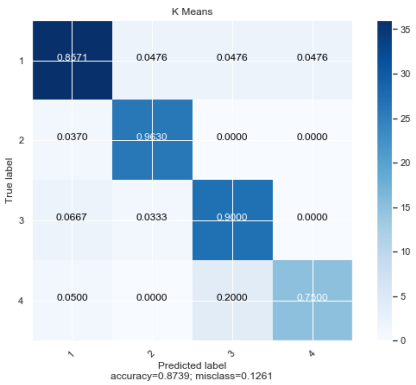


Appendix B

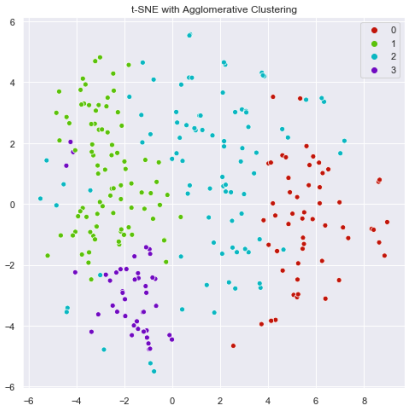
Agenda Wise Sentence Clustering

B.1 K-Means

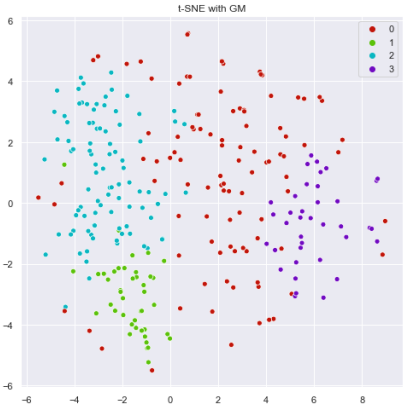


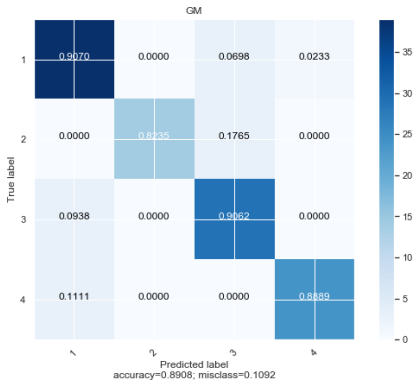


B.2 Agglomerative Clustering

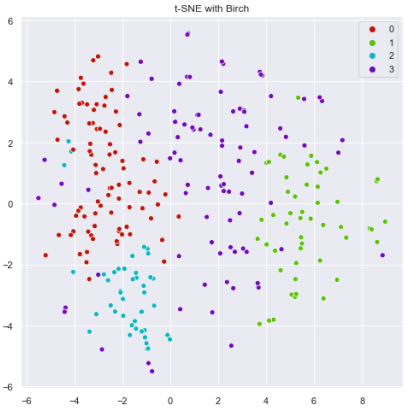


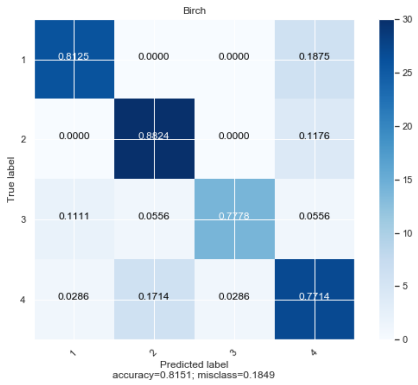
B.3 Gaussian Mixture Model





B.4 Balanced Iterative Reducing and Clustering using Hierarchies





Appendix C

Website User Interface

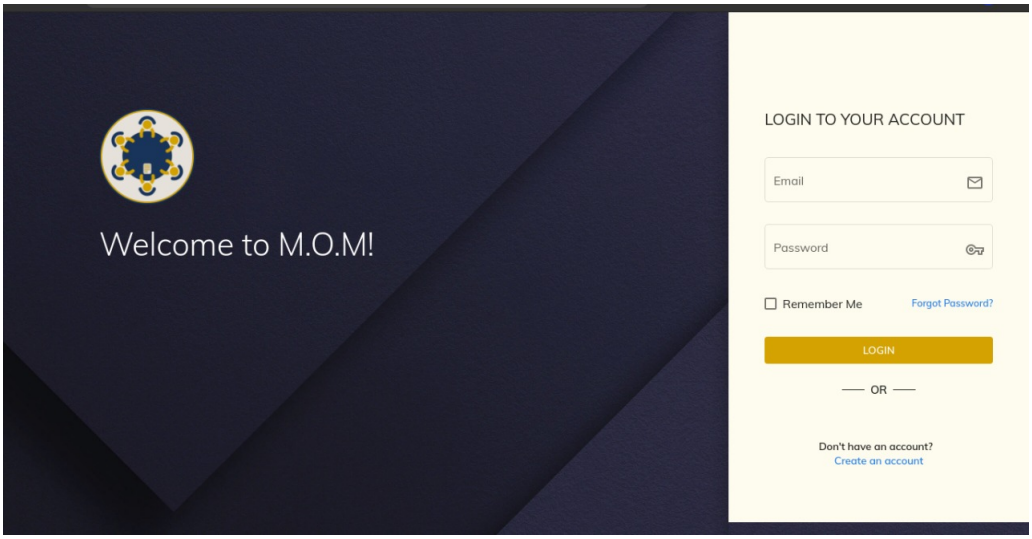


FIGURE C.1: Login Screen - UI

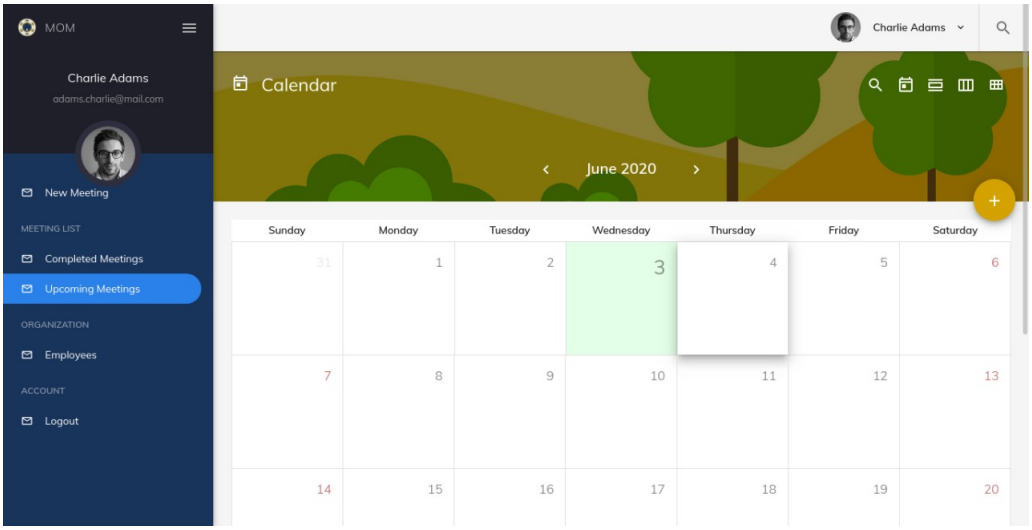


FIGURE C.2: Calendar Screen - UI

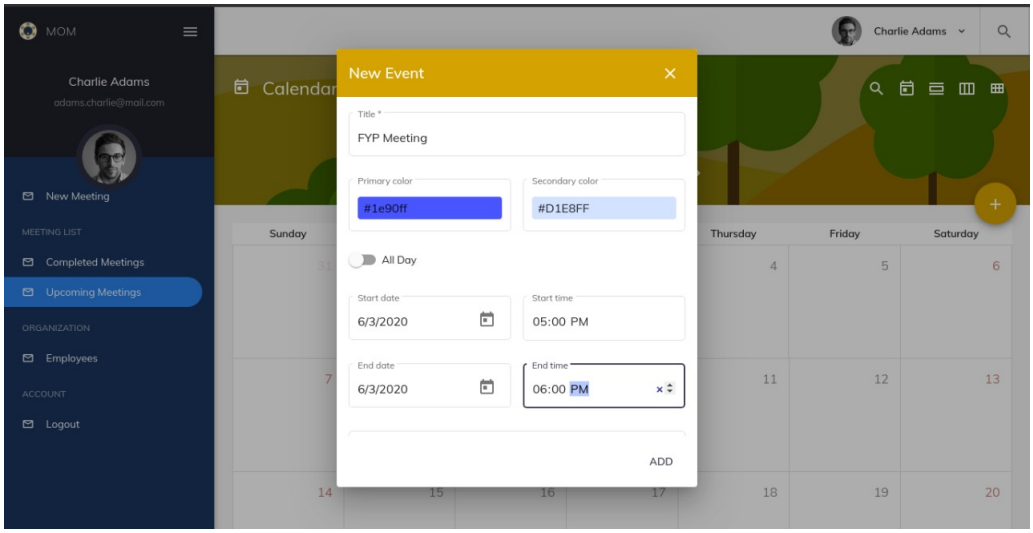


FIGURE C.3: Schedule Meeting PopOver - UI

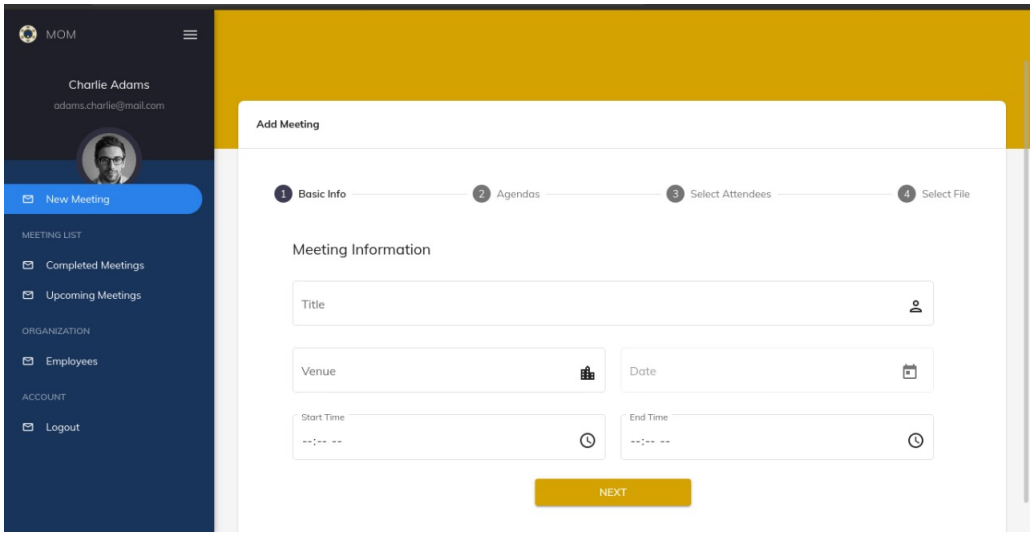


FIGURE C.4: Add Meeting - UI

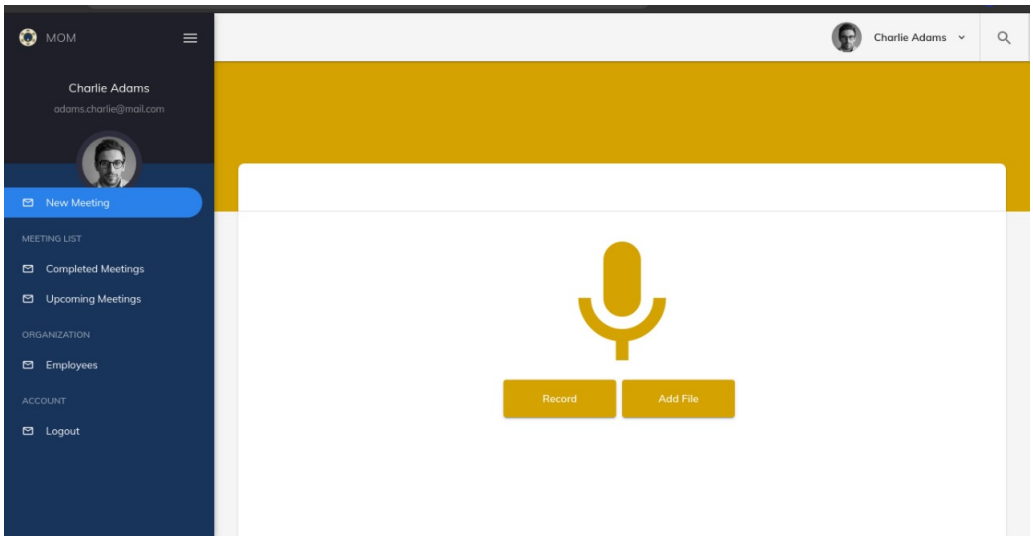


FIGURE C.5: Upload or Record Meeting Screen - UI

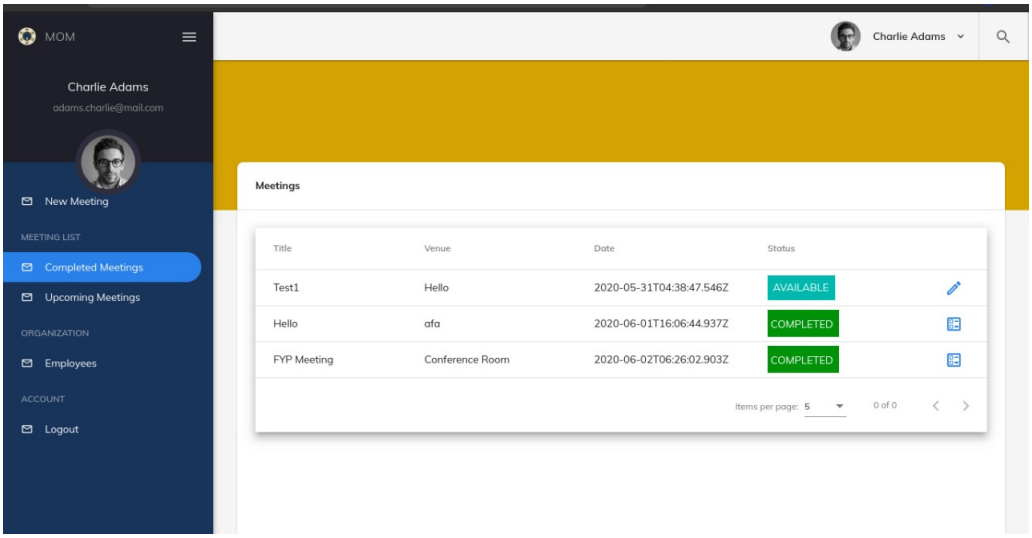


FIGURE C.6: Meetings Recording Ready PopOver - UI

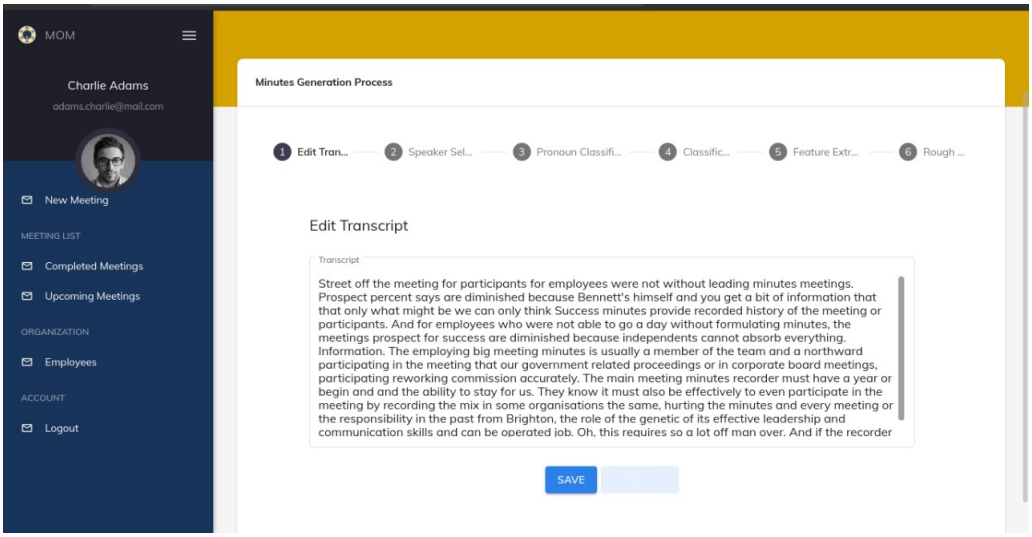


FIGURE C.7: Transcript Screen - UI

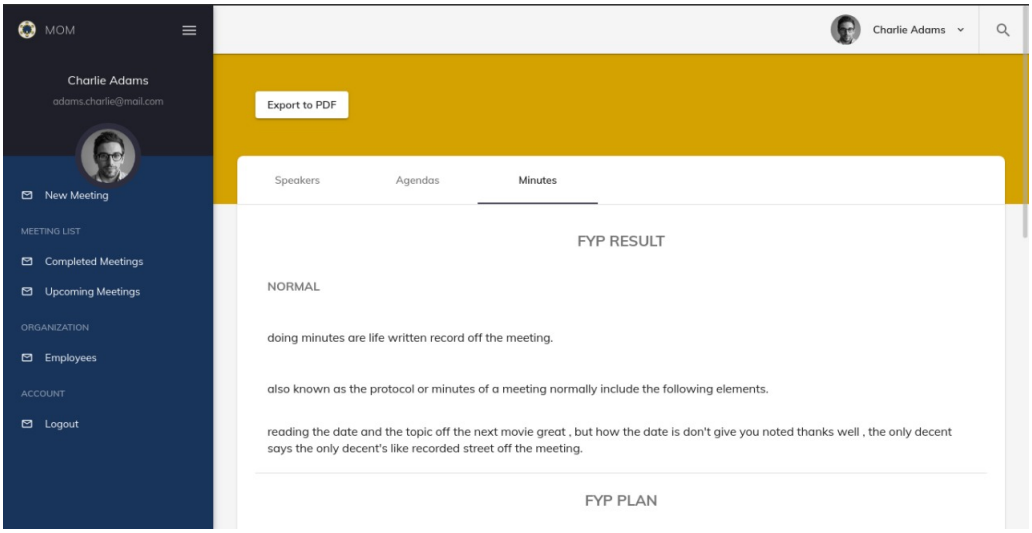


FIGURE C.8: Minutes of Meeting Screen - UI