2024 International Conference on Emerging Smart Computing and Informatics (ESCI)
AISSMS Institute of Information Technology, Pune, India. Mar 5-7, 2024

Minutes of Meeting Generation for Online Meetings Using NLP & ML Techniques

1st Renu Kachhoria
Associate Professor (AI&DS)
Vishwakarma Institute Of Information Technology, Pune - 411048, India
renu.kachhoria@viit.ac.in

2nd Netal Daga
Undergraduate (AI&DS)
Vishwakarma Institute of Information Technology,Pune - 411048, India
npsdaga@gmail.com

3rd Himanshu Ramteke
Undergraduate (AI&DS)
Vishwakarma Institute of Information Technology,Pune - 411048, India
hsramteke21@gmail.com

4th Yash Akotkar
Undergraduate (AI&DS)
Vishwakarma Institute of Information Technology, Pune - 411048, India
yashakotkar123@gmail.com

5th Samarth Ghule
Undergraduate (AI&DS)
Vishwakarma Institute of Information Technology,Pune - 411048, India
samarthghule27@gmail.com

2024 International Conference on Emerging Smart Computing and Informatics (ESCI) | 979-8-3503-0661-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/ESCI59607.2024.10497256

***Abstract*—Within the realm of remote work, meetings serve as unique spaces where individuals summon to exchange concepts and reach consequential conclusions. Currently, they are written down by hand, and while it can be challenging to keep up, occasionally they overlook crucial information. Therefore, automating the process of creating meeting minutes with machine learning and natural language processing techniques has emerged as a promising way to do rid of the tedious human processes involved in recording and summarizing meeting discussions. This study presents a novel method for automating the creation of meeting minutes through the use of machine learning (ML), natural language processing (NLP) algorithms, and several summarization models, including Google's PEGASUS, BART, LED, Meeting Summary, and bartconversation summary. In order to verify the model's functionality, audio clips from different websites are gathered, and the results are computed and compared using the ROUGE-1, ROUGE-2, ROUGE-P, ROUGE-R, and ROUGE-F scores. With the aid of the Whisper AI model and pyAnnote, the suggested solution makes extensive use of natural language processing (NLP) to evaluate meeting transcripts. It is specifically made to create transcripts for online meetings, starting with the addition of time stamps, speaker identification, and the assignment of each sentence to the appropriate speaker. After that, the text is divided up into distinct sentences for processing. This technique accurately identifies and summarizes important themes, crucial choices, action items, and other significant details discussed throughout the discussion.**

***Index Terms*—Meeting minutes, speaker diariazation, Whisper AI, pyAnnote, text, audio-to-transcript, summary generation.**

I. Introduction

Before this technology, meetings needed physical attendance. The person taking notes wrote down details like start time, date, and who spoke. Clear titles organized discussions and decisions. After the meeting, the note-taker edited and sent minutes via email. This traditional method has drawbacks, such as being time-consuming, less accurate, and unable to capture every detail.

Digital alternatives like shared platforms and video recordings address some issues. Online meetings are vital for remote work, but creating accurate meeting minutes (MoM) remains challenging. Current methods often miss crucial information, leading to inefficiencies and miscommunications.

State-of-the-art speech recognition and natural language processing (NLP) technologies offer automated meeting minutes creation. This minimizes errors, saves time, and provides quick access to vital information, improving record quality.

Automatic Text Summarization has two types: abstractive (creates new sentences) and extractive (selects significant sentences). The research focuses on key concerns: - Incomplete Content: Current methods often miss important conversations. - Time-Consuming Manual Labor: Creating MoMs manually is error-prone and time-consuming. - Lack of Automation: MoM generation processes need automation for scalability. - Documentation: Effective management of critical documents is essential for future reference. - Ineffective Collaboration: Imprecise MoMs can lead to misconceptions in meetings.

Benefits of the Code and Models: - Speech-to-Text Transcription: Converts spoken language into text, aiding content generation and accessibility. - Speaker Diarization: Identifies speakers in audio files, useful for meeting summaries. - Text Summarization: Automatically produces concise transcriptions, saving time. - Audio Format Compatibility: Processes various audio files, ensuring interoperability. - Mono Conversion: Converts stereo audio to mono, enhancing precision.

II. Literature Survey

In the study conducted by Megha Manuel1 et al. [1], automated generation of meeting minutes was done using deep learning. Two recordings, one with a speaker and another with a meeting, were analyzed. Feature vectors using MFCC models were matched with the DTW algorithm for speaker identification. Speech-to-text converted the meeting recording

Authorized licensed use limited to: Chaitanya Bharathi Institute of Tech - HYDERABAD. Downloaded on October 17,2024 at 09:04:59 UTC from IEEE Xplore. Restrictions apply.

into text. The combination of speaker verification and speech-to-text produced a transcript. A T5 transformer with LSTM structure was used for abstractive text summarization.

In a study led by Nitish Singh Rajpurohit and team [2], they used machine learning to automate meeting minutes with Whisper AI's speech recognition. The system, using natural language processing, RNNs, and transformers, extracts key meeting details. Compared to manual note-taking, it saves time and works efficiently, featuring speech-to-text, speaker ID, and summarization with BERT and GPT-2 for streamlined meeting documentation across industries.

In the study conducted by Rakhmat Arianto et al. [3], drafting of online meeting minutes is done based on video recording using topic modelling. The research paper proposes an automatic note-taking system using Latent Dirichlet Allocation (LDA) topic modeling. It converts audio from meetings into text, applies LDA to identify key topics, and evaluates them for coherence and similarity to manually prepared meeting minutes.

Hui Liu, Huan Liu, and team [4] developed a mobile system for creating and analyzing meeting minutes using voice data. The cloud-based system automates minute generation, employing distributed architecture for security. It uses speech processing, speaker ID, and emotion recognition to extract action items. Two introduced algorithms, ELT for advanced summaries and BLSTM for sentiment analysis, show efficiency in real-world testing on Qiniu Cloud, confirming the system's effectiveness.

In the study conducted by Jingqing Zhang et al. [5], proposes a new self-supervised pre-training objective for abstractive text summarization called gap-sentences generation. The authors pre-train large Transformer-based encoder-decoder models on massive text corpora using this objective and evaluate the performance of their best model, PEGASUS, on various downstream summarization tasks. They also explore the application of PEGASUS in low-resource summarization scenarios and conduct human evaluation studies, which confirms that the generated summaries by PEGASUS are of high quality.

In the study conducted by Mike Lewis et al. [6],introduces BART, a autoencoder used for pretraining sequence-to-sequence models for natural language tasks. BART combines bidirectional and autoregressive transformers and can generalize other pretraining schemes such as BERT and GPT. It is trained by corrupting text and learning to reconstruct the original text. BART achieves state-of-the-art results on various tasks and shows strong performance across different training objectives.

Urvashi Khanna and team [7] studied summarizing lengthy financial documents and suggested using Transformer-based models, particularly the Longformer-Encoder-Decoder (LED) model. Through experiments and participation in a financial summarization task, they found that multi-stage fine-tuning enhances the model's performance. The authors stress the significance of fine-tuning for specific domains and point out drawbacks of traditional summarization methods. Their system shows promising results, evaluated by ROUGE scores on the validation dataset.

III. METHODOLOGY

The architecture begins the process of generating meeting minutes, with processing the extracted audio from the meeting. The extracted audio is provided to the model, generating transcription, which serves as the foundation for the subsequent steps. The steps for overall process are:

A. Speaker Diarization

Speaker diarization is a crucial step in the minutes generation process. It involves distinguishing and labeling different speakers within the audio recording. This pre-trained models can effectively separate speakers and assign speaker identities, ensuring that the minutes accurately reflect who said what during the meeting.

Models Used:

a) Whisper OpenAI Model: Whisper AI [2] is an open-source encoder-decoder transformer designed for audio processing tasks, specifically tailored for tasks such as speech recognition and captioning. The model's operation can be broken down into the following steps:

1. Input Processing: The model takes audio input and segments it into 30-second chunks, allowing for manageable processing and analysis.

2. Spectrogram Conversion: Each 30-second audio chunk undergoes conversion into a log Mel Spectrogram. This transformation provides a representation of the audio signals in the frequency domain, capturing essential features for subsequent analysis.

3. Encoder Operation: The log Mel Spectrogram is fed into the encoder component of the transformer architecture. The encoder processes the input data hierarchically, extracting relevant features at different levels of abstraction.

4. Feature Extraction: The encoder performs feature extraction on the log Mel Spectrogram, highlighting key patterns and information present in the audio data.

5. Decoder Processing: The output from the encoder is then passed to the decoder component. The decoder is trained to generate corresponding captions or transcriptions for the given audio input.

Authorized licensed use limited to: Chaitanya Bharathi Institute of Tech - HYDERABAD. Downloaded on October 17,2024 at 09:04:59 UTC from IEEE Xplore. Restrictions apply.

b) PyAnnote: Pyannote is a powerful tool for speaker diarization, a crucial task in understanding who spoke when in an audio recording. The pipeline of the Pyannote model involves several intricate steps, each contributing to the accurate identification and clustering of speakers. Following are some functionalities:

1. Feature Extraction: Pyannote's main strength lies in its effective feature extraction ability. It uses methods like MFCCs (Mel-Frequency Cepstral Coefficients) or spectrograms, simplifying the process of training neural networks. This feature extraction supports the creation of an augmented dataset. This involves introducing variations by adding noise from databases like MUSAN. Pyannote ensures applying augmentation dynamically during training.

2. Sequence Labeling: The sequence labeling process encompasses various essential sub-steps:
- Voice Activity Detection (VAD): Pyannote excels in detecting speech regions within the given audio. The VAD step is crucial for isolating segments with meaningful speech, laying the foundation for subsequent analysis.
- Speaker Change Detection: Identifying points where speakers change in an audio recording is fundamental to speaker diarization. Pyannote's speaker change detection efficiently pinpoints transitions between speakers, aiding in the segmentation of the audio.
- Speaker Overlapping Detection: In scenarios where two speakers utter simultaneously, Pyannote's overlapping detection identifies these regions. This capability is essential for a comprehensive understanding of the dynamics within the audio.
- Resegmentation: It is an unsupervised task in which Pyannote refines speech turn boundaries and labels emerging from the diarization pipeline. The number of speakers is manually provided, and the resegmentation process adjusts accordingly.

3. Speaker Embedding and Clustering: The next phase involves converting the input into X-vector format, a prerequisite for the speaker diarization clustering process. Pyannote employs agglomerative clustering to group similar speaker embeddings, forming distinct clusters.

```
# Example speaker clustering function

def speaker_clustering(embeddings, num_speakers):
    clustering = AgglomerativeClustering(
        n_clusters=num_speakers, affinity='cosine', linkage='average').fit(embeddings)
    labels = clustering.labels
    return labels
```

Fig. 1: Speaker Clustering Function

4. Speaker Labelling: Finally, the clusters obtained through clustering are labeled. In the context of our use case, where two speakers were taken, the clusters are straightforwardly labeled as "Speaker 1" and "Speaker 2." The labeled output is then provided in a transcript.txt format, serving as the output from the diarization model. This transcribed text file becomes the input for our next task, which is summarization.

Evaluation Metric for diarization: Diarization Error Rate [8] is a metric commonly used in speaker diarization tasks to evaluate the accuracy of the diarization system. It quantifies the disparity between the reference and hypothesis speaker diarizations, providing valuable insights into the performance of the system. The DER is computed using the following formula:
- False alarm: represents the total hypothesis speaker time not attributed to a reference speaker.
- Missed detection: represents the total reference speaker time not attributed to a hypothesis speaker.
- Confusion: represents the total reference speaker time attributed to the wrong speaker.
- Total: It is the total reference speaker time, i.e., the sum of the duration of all reference speaker segments.

$$\mathrm{DER} = \frac{\text{false alarm} + \text{missed detection} + \text{confusion}}{\text{total}}$$

Fig. 2: DER Formula

B. Summary Generation

This step follows the minutes generation process. It turns online meeting transcripts into brief, well-organized summaries, improving efficiency and accuracy. This streamlining of record-keeping boosts productivity and facilitates better communication in remote work settings. While the generated transcript offers a comprehensive meeting record, it can be lengthy and overly detailed.

Models Used:

a) Google Pegasus: Google's Pegasus is a pre-trained model for abstractive text summarization. It is used to generate a resonable and short summary of the meeting's content, preserving key points and details. [5]

b) BART: BART, or Bidirectional and Auto-Regressive Transformers, is a pre-trained model specializing in text summarization. It efficiently condenses meeting minutes while retaining essential details, functioning as an autoencoder for various tasks [6].

c) led: The LED (Longformer Encoder-Decoder) model is great at summarizing long content like narratives, articles, and papers. It can handle up to 16,384 tokens at once, using a transformer architecture with extended attention. This simplifies processing lengthy documents for tasks like question answering and document classification [7].

Authorized licensed use limited to: Chaitanya Bharathi Institute of Tech - HYDERABAD. Downloaded on October 17,2024 at 09:04:59 UTC from IEEE Xplore. Restrictions apply.

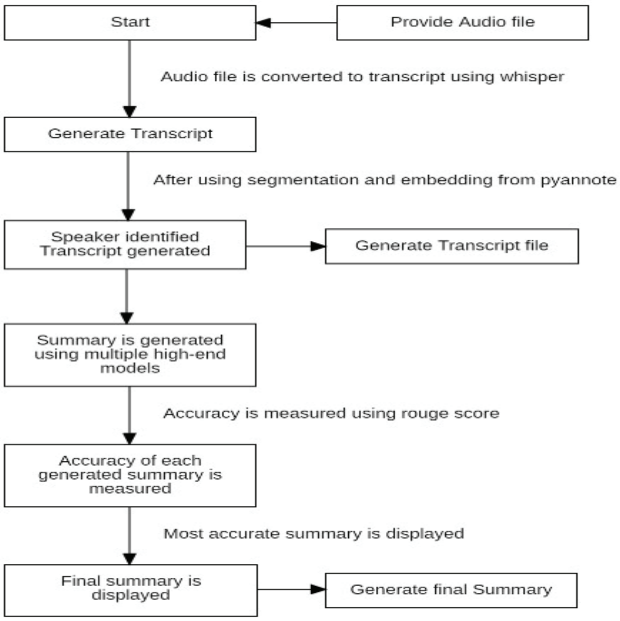

Fig. 3: Flowchart

d) Meeting Summary: This model is crafted for generating short meeting summaries, capturing key points and important details. It's handy for businesses, media, education, legal, and other fields where summarizing conversations is valuable.

e) bart-conversation-summarizer: A conversation summarizer is a model that generates concise summaries of conversations, like chat logs or dialogues. It condenses lengthy discussions into shorter summaries, capturing key points or the overall context from a series of turns [6].

Evaluation Metric: ROUGE metrics assess the quality of machine-generated summaries. We use:
- ROUGE-N (ROUGE-1, ROUGE-2) to evaluate how well the summary captures key phrases or sentences from the reference minutes.
- ROUGE-P (Precision) to check if the summary avoids incorrect information or irrelevant content.
- ROUGE-R (Recall) to measure how well the summary covers content from the reference, emphasizing recall.
- ROUGE-F (F-Measure) to balance the trade-off between recall and precision.

IV. Work Flow

The user starts by submitting an audio file, setting off a detailed analysis. The audio is turned into written text through transcription, creating a comprehensive transcript of the conversation. The system then uses advanced methods to recognize different speakers in the audio.

With the transcript and speaker details in place, the system employs specialized models for summarization. These models review the transcript, extracting important details to produce multiple summaries that offer different perspectives. Each summary provides a condensed understanding of the content, contributing to a more comprehensive view when considered together.

The generated summaries undergo a thorough quality assessment to ensure accuracy. This involves comparing them to predetermined benchmarks or references to evaluate how well they capture the key points in the audio.

The final step involves selecting the most accurate summary from the collection produced by the best models. This chosen summary is presented as the final product, offering a polished and streamlined version of the audio information. This approach ensures that the summarization process effectively conveys the main points accurately, facilitating quick understanding for the user.

V. Results and Discussion

a) ***Speaker Diarization:*** The model took "Chat with new friends Learn English.mp3" as input, generating a manual transcription named "chatWithNewFriendsLearnEnglish-GroundTruth.txt." The model outputted "transcript.txt," including time stamps, speaker IDs, and corresponding sentences. Figure 4 illustrates this process.

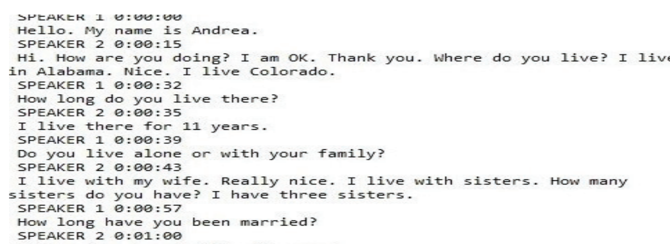
SPEAKER 1 0:00:00
Hello. My name is Andrea.
SPEAKER 2 0:00:15
Hi. How are you doing? I am OK. Thank you. Where do you live? I live in Alabama. Nice. I live Colorado.
SPEAKER 1 0:00:32
How long do you live there?
SPEAKER 2 0:00:35
I live there for 11 years.
SPEAKER 1 0:00:39
Do you live alone or with your family?
SPEAKER 2 0:00:43
I live with my wife. Really nice. I live with sisters. How many sisters do you have? I have three sisters.
SPEAKER 1 0:00:57
How long have you been married?
SPEAKER 2 0:01:00

Fig. 4: Transcript file generation of Audio File - "Chat with new friends Learn English.mp3"

b) ***Speaker Diarization metric:*** The reference diarization is shown in Figure 5. The correct sentences of speaker 1 are indicated by the color red, and the correct sentences of speaker 2 are indicated by the color green. The reference diarization is used as the benchmark for assessing how well the diarization system performs:

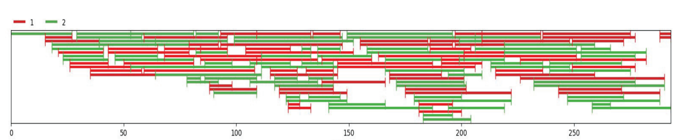

Fig. 5: Reference Diarization

The hypothesis diarization is depicted in Figure 6, which shows the actual values that the diarization system predicted.

Authorized licensed use limited to: Chaitanya Bharathi Institute of Tech - HYDERABAD. Downloaded on October 17,2024 at 09:04:59 UTC from IEEE Xplore. Restrictions apply.

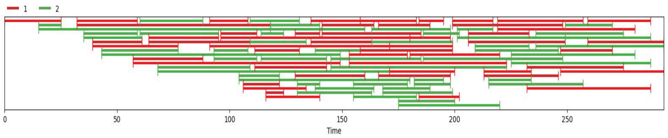

Fig. 6: Hypothesis Diarization

Diarization Error Rate Calculation: By comparing the reference and hypothesis diarizations, we apply the DER formula to quantify the diarization error.

Figure 7's function uses the Diarization Error Rate metric without skipping overlaps and with a collar of 0.0, meaning that speaker segmentation mismatches are not tolerated.

```
# Function to calculate DER between reference and hypothesis
def calculate_der(reference, hypothesis):
    der_metric = DiarizationErrorRate(collar=0.0, skip_overlap=False)
    der = der_metric(reference, hypothesis)
    return der

# Calculate DER
der_result = calculate_der(reference, hypothesis)
```

Fig. 7: Function to calculate DER

```
1   # Calculate DER
2   der = calculate_der(reference, hypothesis)
3   print(f"Diarization Error Rate (DER): {der:.2%}")

Diarization Error Rate (DER): 25.99%
```

Fig. 8: DER of audio

We have implemented the same for multiple audio samples and got DER between 20% - 35%, which is considerably good.

*c) **Summary Generation**:* Five models each generate a separate summary by using the "transcript.txt" file, output file produced by our speaker diarization model, as the input. The models that produce the summaries are as follows:

```
[{'summary_text': 'SPEAKER 1 0:00:32 How long do you live there?
SPEAKER 1 0:00:35 I live there for 11 years. SPEAKER 1 0:00:43 I live
with my wife. SPEAKER 1 0:00:57 How long have you been married? SPEAKER
1 0:01:04 It was nice talking to you, but I have to go. SPEAKER 2
0:01:58 It was nice to meet you too. SPEAKER 2 0:02:21 Michael, do you
like working in the factory? SPEAKER 1 0:02:25 No, I do not like
working in the factory, but I need money.'}]

Summary text has been saved to /content/summaryGoogle.txt
```

Fig. 9: Google's PEGASUS Model Summary

```
[{'summary_text': "Hi Andrea, how are you doing? I am OK. Where do you
live? I live in Alabama. I live with my wife. Where are you from?  I am
from Hawaii. I would be very happy if you come to Hawaii. Hi William.
Hi Amelia. Hi Michael. Hi Emily. How are you? I'm fine."}]

Summary text has been saved to /content/summaryBart.txt
```

Fig. 10: BART Model Summary

```
[{'summary_text': 'Amelia introduces William to the boy, asking him
questions about his background. He answers that he is from Colorado and
has been married for five years. She asks if he likes working in the
factory, and he responds that he does not like working in factories.
Amelia suggests they meet for coffee, but before they can make up, he
tells her that he wants to go to college.'}]

Summary text has been saved to /content/summaryLed.txt
```

Fig. 11: LED Model Summary

```
[{'summary_text': "Andrea lives in Alabama, she's been there for 11
years. She's married to her husband for 5 years. Emily lives in
Colorado, she has three sisters. William is from Switzerland, he's
working in a factory. Emily works in the hospital, she loves it.
Michael is a doctor, he loves his job."}]

Summary text has been saved to /content/summaryBartConvo.txt
```

Fig. 12: bart-conversation-summarizer Model Summary

```
[{'summary_text': "Andrea lives in Alabama, she's been there for 11
years, she has three sisters, William lives in Switzerland, he's been
married for 5 years, Michael works in the factory, Emily studies at
university, and William wants to come to Hawaii. "}]

Summary text has been saved to /content/summaryMeetingSum.txt
```

Fig. 13: MEETING SUMMARY Model Summary

*d) **Evaluation Metric for summary models**:* The comparison graph displays the performance of five different summarizing models (Google Pegasus, BART, led-base-book-summary, bart-conversation-summary, and Meeting Summary) using three different Rouge metrics (Rouge-1, Rouge-2, and Rouge-L). In the graph below, the Rouge measurements are displayed on the x-axis, and the Rouge scores vary from 0.00 to 0.35 on the y-axis.

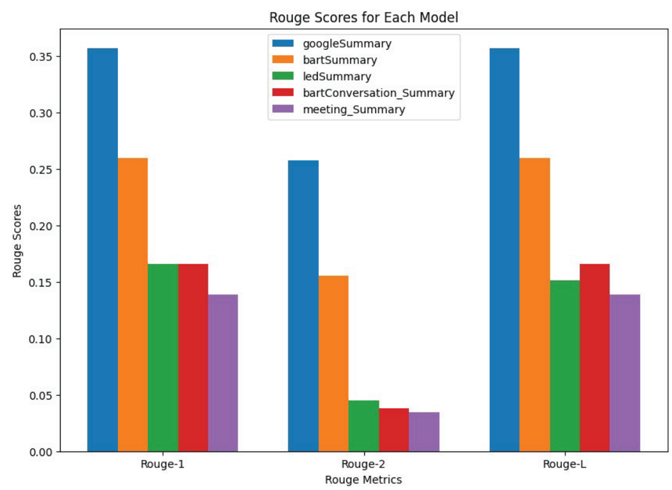

Fig. 14: ROUGE scores of 5 summary models

- Rouge-1 Score: Google Pegasus topped the list with a Rouge-1 score of 0.35, showcasing strong performance in unigram overlap. BART follows closely with a score of 0.26, indicating robust performance. Led-base-book-summary and Bart-conversation-summary tie for third place with Rouge-1 ratings of 0.16. Meeting Summary lags behind with a Rouge-1 score of 0.14.
- Rouge-2 Score: Google Pegasus excels with a Rouge-2 score of 0.25, demonstrating strong bigram overlap recognition. BART follows with a score of 0.15, showing good but smaller bigram overlap. Led-base-book-summary and Bart-conversation-summary score lower at 0.05 and 0.045, respectively, for two-word sequence recognition. Meeting Summary closely follows with a Rouge-2 score of 0.04.
- Rouge-L Score:Google Pegasus leads with the highest Rouge-L score of 0.35, indicating its effectiveness in

Authorized licensed use limited to: Chaitanya Bharathi Institute of Tech - HYDERABAD. Downloaded on October 17,2024 at 09:04:59 UTC from IEEE Xplore. Restrictions apply.

finding the longest common subsequence. BART follows closely with a Rouge-L score of 0.26, showcasing excellent performance. Led-base-book-summary and Bart-conversation-summary tie for third place with Rouge-L ratings of 0.15 and 0.16, respectively. Meeting Summary trails slightly with a Rouge-L score of 0.14.

In summary, Google Pegasus outperforms the other models in Rouge metrics when it comes to generating summaries that closely resemble reference summaries. BART also operates quite well; results from the led-base-book-summary and bart-conversation-summary are comparable. Meeting Summary provides decent ratings, but it performs a little worse than the top-performing models in this.

VI. Conclusion

This research method automates minute creation for virtual meetings, showing promise across various businesses and institutions. By integrating speaker diarization, audio transcription, and advanced summary tools, it produces concise and informative meeting minutes. The system is compatible with various deep learning models, addressing challenges with diverse audio formats. It's reliable, user-friendly, with error management and mono conversion.

The program uses big deep learning models for tasks like recognizing speakers and transcribing speech. This makes downloading slow and requires a lot of storage space. The accuracy of recognizing speech and speakers depends on factors like audio quality and pre-trained models.

VII. Future Scope

Upcoming research can concentrate on code optimization for increased efficiency, such as quicker speaker diarization and transcription. This might entail investigating deeper learning models with more efficiency and formulas.

Acknowledgment

We would like to express our heartfelt gratitude to the AI&DS department at Vishwakarma Institute Of Information Technology, Pune for their support and guidance in the development of the project on "Minutes Of Meeting Generation For Online Meetings Using NLP & ML Techniques."

We extend our sincere appreciation to the faculty members and mentors who provided valuable insights and feedback throughout the project duration. Their expertise and knowledge greatly contributed to the success of this endeavor.

References

[1] Megha Manuel1, Amritha S Menon1, Anna Kallivayalil1, Suzana Isaac1 and Lakshmi K.S, "Automated Generation of Meeting Minutes Using Deep Learning", in 2022 International Journal of Computing and Digital Systems, Int. J. Com. Dig. Sys. 12, No.1 (Jul-2022), ISSN pp. 2210-142X.

[2] Nitish Singh Rajpurohit, Srujan SP, Tejas Panagar TK, Mrs. K Padma Priya, "Automated Generation Of Minutes Of Meeting Using Machine Learning" in 2023 International Journal of Advance Research and Innovative Ideas in Education (IJARIIE), Vol-9 Issue-3 2023, pp. IJARIIE-ISSN(O)-2395-4396.

[3] Rakhmat Arianto, Alwy Abdullah, Usman Nurhasan, Rokhimatul Wakhidah, "Drafting of Online Meeting Minutes Based on Video Recording Using Topic Modelling", in 2023 Journal of Information Systems (e-Journal), Vol. 10 — No. 1 — Th. 2023, pp. ISSN 2442-7888.

[4] Hui Liu, Huan Liu, Xin Wang, Wei Shao, Xiao Wang, Junzhao Du, Jonathan Liono, Flora D. Salim, "SmartMeeting: An Novel Mobile Voice Meeting Minutes Generation and Analysis System", in 2023 Journal of Information Systems (e-Journal), Vol. 10 — No. 1 — Th. 2023, pp. ISSN 2442-7888.

[5] Jingqing Zhang, Yao Zhao, Mohammad Saleh, Peter J. Liu, "PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization", in 2020 37th International Conference on Machine Learning, Online, PMLR 119, 2020

[6] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, Luke Zettlemoyer, "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension", arXiv:1910.13461v1 [cs.CL] 29 Oct 2019.

[7] Urvashi Khanna, Samira Ghodratnama, Diego Molla, Amin Beheshti, "Transformer-based Models for Long Document Summarisation in Financial Domain", in 2022 the 4th Financial Narrative Processing Workshop @ LREC 2022, Marseille, 24 June 2022.

[8] Bredin, H. (2017) pyannote.metrics: A Toolkit for Reproducible Evaluation, Diagnostic, and Error Analysis of Speaker Diarization Systems. Proc. Interspeech 2017, 3587-3591, doi: 10.21437/Interspeech.2017-411

[9] Rajat Verma, Sparsh Gupta, Shubh Sharma, Tanishq Aggarwal, Mahesha A.M, "Automated Meeting Minutes Generator", in 2022 Journal of Emerging Technologies and Innovative Research (JETIR), January 2022, Volume 9, Issue 1, (ISSN-2349-5162).

[10] Tirthankar Ghosal, Ondrej Bojar, Muskaan Singh, Anja Nedoluzhko, "Overview of the First Shared Task on Automatic Minuting (AutoMin)", in 2021 at Interspeech, 10.21437/AutoMin.2021-1.

[11] Kanhgo Heo, Jinwoo Yang, Donghyun Kim, Kyoungsoo Bok, Jaesoo Yoo, "Design and Implementation of Minutes Summary System Based on Word Frequency and Similarity Analysis", in 2019 The Journal of the Korea Contents Association, Volume 19 Issue 10, 2019, 1598-4877(pISSN), 2508-6723(eISSN).

[12] Anuj Pandya, Prof. Namrata Gawande, "Automatic Generation of Minutes of Meetings", in 2022 International Journal of Scientific Research in Science, Engineering and Technology, Volume 9 Issue 2, ISSN : 2394-4099.

[13] Vinnarasu A., Deepa V. Jose, "Speech to text conversion and summarization for effective understanding and documentation", in 2019 International Journal of Electrical and Computer Engineering (IJECE), Vol. 9, No. 5, October 2019, pp. 3642-3648

[14] Wolf, Thomas, Debut, Lysandre, Sanh, Victor,Chaumond, "Transformers: State-of-the-Art Natural Language Processing", in oct 2020 Association for Computational Linguistics, Version 5

[15] Jia Jin Koay, Alex Roustai, Xiaojin Dai, Fei Liu, "A Sliding-Window Approach to Automatic Creation of Meeting Minutes", Proceedings of NAACL-HLT 2021: Student Research Workshop, June 6–11, 2021. ©2021 Association for Computational Linguistics

Authorized licensed use limited to: Chaitanya Bharathi Institute of Tech - HYDERABAD. Downloaded on October 17,2024 at 09:04:59 UTC from IEEE Xplore. Restrictions apply.