Real Time Captioning and Notes Making of Online Classes

A. Vasantha Raman, V. Sanjay Thiruvengadam, J. Santhosh, Thenmozhi Durairaj

Real Time Captioning and Notes Making of Online classes

Vasantha Raman A, Sanjay Thiruvengadam V, Santhosh J, and Thenmozhi Durairaj

Sri Sivasubramaniya Nadar College Of Engineering, Chennai.
{vasantharaman003, stsanjay807, santhoshjd23}@gmail.com,
theni_d@ssn.edu.in

Abstract. Due to the COVID-19 pandemic, all activities have turned online. The people who are hard of hearing are facing high difficulty to continue their education. So, the presented system supports them in attending the online classes by providing the real time captions. Additionally, it provides summarized notes for all the students so that they can refer to them before the next class. Google Speech to Text API is used to convert the speech to text, for providing real time captions. Three text summarization models were explored, namely BART, Seq2Seq model and the TextRank algorithm. The BART and the Seq2Seq models require a labelled dataset for training, whereas the TextRank algorithm is an unsupervised learning algorithm. For BART, the dataset is built using semi supervised methods. We evaluated all these models with rouge score evaluation metrics, among these BART proves to be best for our dataset with the following scores of 0.47, 0.30, 0.48 for rouge-1, rouge-2 and rouge-l respectively.

Keywords: Bidirectional Auto Regressive Transformer (BART) · Sequence to Sequence Model (Seq2Seq Model) · Transformers · Summarization · Speech-to- Text

1 Introduction

In wake of the COVID-19 pandemic, all the activities have turned online. It came to be called the virus that changes the way we use the internet. With increasing concerns over social distancing, people are seeking ways to connect socially online. All the classes are being conducted online and going by the present trends, the online mode of study is here to stay. Global corporate outlook of the scenario is no different. Companies that once had whiteboard pitch meetings, have turned to screen sharing and presentations on online meeting platforms. College lectures is one such scenario, wherein both the students and teachers are involved in the process of imparting and enriching their knowledge throughout the course of lecture. Online meeting platforms are playing a key role in making sure that the activities stay afloat during the course

2 of pandemic. To further aid the same, we introduce a real time caption and lecture summarizer system. This research work helps reduce the stress on

students by doing the summarization on the contents of the class and helps in preparation of exams. This system can be considered as an extension of the idea brought by IBM Viascribe [9] module wherein the speech is converted to text real time. This idea used by Viascribe is expanded to new depths by adding a summarization feature to the module lecture endings. People who miss classes can refer to the summarized notes for reference in a quick glance. This proposed system can be used by Online Meeting Platforms such as Zoom, Google Meets and Microsoft Teams. This system can also be extended to offline classes to real time scenarios as well. People with hearing impairments feel very difficult to understand the online classes. The proposed system supports them in attending the classes by providing the real time captions along with summaries of the classes.

The main objective of the proposed system is to create an application that provides services of Real Time Captioning and Notes Making for a Lecture.

The focus of this research work is on speech to text conversion and on text summarization. To implement the functionality of speech to text, Google Speech to Text API [3] is used. For providing the utility of summarization, BART model [5] is used. The model was chosen, after consideration of other established models like Sequence to Sequence and Text Rank [7]. The chosen BART model was attuned to suit that of a lecture on Computer Science domain by training it over a course of NPTEL video lectures and its summaries, making up for a data set of a thousand eight hundred rows.

2 Related Work

IBM proposed a system named ViaScribe and Hosted Transcription Service [9] in 2013 which is used for real time captioning system using speech recognition.It is captioning software for digital audio and video that creates a written transcript and audio recording, as well as automatically captions live audio content as it occurs. It used ViaVoice engine in the process of doing so. It was observed to produce 80% accuracy. They were working on using cloud to extend the service. But, recently it was announced that IBM discontinued the Viascribe service

Microsoft proposed a system named 'Advances in online audio visual meeting transcription' [6] in 2019 which is used to identify the speakers while transcripting when people are talking simultaneously in a meeting.In this speech separation system camera signal is not used. They used continuous speech separation method.the names of expected meeting attendees are known beforehand, the transcription system should be able to provide each utterance with the true identity (e.g., "Alice" or "Bob") instead of a randomly generated label like "Speaker1". It is often required to show the transcription in near real time, which makes the task more challenging.

IBC proposed a system named 'Just in time prepared captioning for live transmission' [8] in 2017 which is an in-time transcription of audio in a live news broadcast. They done this using automatic speech recognition system. This work is done specific to news live broadcast. They are trying to increase accuracy.

Their focus was primarily on suiting to accents of english like UK,Australia and US. The percentage accuracy of pieces marked ranged from 40.91% to 100%, while the percentage of words missing ranged from 0% to 68.52%

Facebook proposed a system named 'Wav2vec 2.0' [1] in 2020, Framework for self-supervised learning of representations of raw audio data.This uses multi layer convolutional neural network.There is no extension to other regional languages.

Review of speech to text (STR) recognition technology for enhancing learning [10] studies and overviews new speech to text technologies in the period of 1999 to 2014 which was proposed on 2014. This is built on speech recognition technology. Their focus was to analyze the results of study and to understand how Speech-to-Text Recognition technology can enhance learning. The review revealed a number of distinct advantages of using STR for learning. That is, STR-generated texts enable students to understand learning content of a lecture better, to confirm missed or misheard parts of a speech, to take notes or complete homework, and to prepare for exams.

The invention of a new simple architecture, Transformer [2] was published on 2017, solely based on attention mechanisms. Self-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence. This aims on introducing transformers model to revolutionize Natural language Processing by overcoming the problems experienced by then most preferred Recurrent Neural Networks (RNN). RNNs generate a sequence of hidden states h_t, as a function of the previous hidden state h_{t1} and the input for position t. This becomes critical at longer sequence lengths, as memory constraints limit batching across examples. Extension of the Transformer to problems involving input and output modalities other than text.

Future n-gram prediction for CNN/DailyMail [12] was proposed in 2020 . It is a pretrained Seq2Seq model called ProphetNet. Instead of optimizing one-step-ahead prediction in the traditional sequence-to-sequence model, the ProphetNet is optimized by n-step ahead prediction that predicts the next n tokens simultaneously based on previous context tokens at each time step.

Summarize text with pre trained encoders [11] was proposed in 2019. In this they used Bidirectional Encoder Representations from Transformers. Quadratic dependency (mainly in terms of memory) on the sequence length due to their full attention mechanism.BERT's key technical innovation is applying the bidirectional training of Transformer, a popular attention model, to language modelling. This is in contrast to previous efforts which looked at a text sequence either from left to right or combined left-to-right and right-to-left training. The paper's results show that a language model which is bidirectionally trained can have a deeper sense of language context and flow than single-direction language models. In the paper, the researchers detail a novel technique named Masked LM (MLM) which allows bidirectional training in models in which it was previously impossible.

NLP tasks like question answering and generation [4] was proposed on 2020 with improved performance and Using sparse attention mechanisms to reduce dependency on sequence length memory. In this approximated learning is achieved

because of sparse attention mechanism. BigBird is a universal approximator of sequence functions and is Turing complete, thereby preserving these properties of the quadratic, full attention model. But, it Requires long context for the purposes of NLP operations including question generation, summarization etc.

A denoising autoencoder for pretraining sequence to sequence models [5] was proposed on 2019. Its trained by corrupting the text with an arbitrary noising function and learning a model to reconstruct it. This involves Exploring new methods to corrupt documents for pre-training and tailoring them to specific tasks including Question generation, Text generation, Summarization etc.. It uses a standard Transformer-based neural machine translation architecture which, despite its simplicity, can be seen as generalizing BERT (due to the bidirectional encoder), GPT (with the left-to-right decoder), and many other more recent pretraining schemes. It achieves new stateof-the-art results on a range of abstractive dialogue, question answering, and summarization tasks, with gains of up to 6 ROUGE.

TextRank [7] system was proposed in 2004. It is based on bringing Order into Texts. Graph based ranking algorithm for text processing and Exploring new ways to link sentences / keywords. Graph-based ranking algorithms like HITS algorithim, Google's PageRank have been successfully used in citation analysis, social networks, and the analysis of the link-structure of the World Wide Web. In short, a graph-based ranking algorithm is a way of deciding on the importance of a vertex within a graph, by taking into account global information recursively computed from the entire graph, rather than relying only on local vertex-specific information. Applying a similar line of thinking to lexical or semantic graphs extracted from natural language documents, results in a graph-based ranking model that can be applied to a variety of natural language processing applications, where knowledge drawn from an entire text is used in making local ranking/selection decisions

Google's Automatic Speech to Text Recognition Systems provide state of the art results [3]. They are built using attention based encoder-decoder architectures. They present results with a unidirectional LSTM encoder for streaming recognition.

The above related work is summarized in the TABLE with the purpose, methodology and the open issues.

3 Proposed System

The proposed system consists of the below mentioned modules.

1. Speech to Text Recognition
2. Data Collection
3. Data Annotation
4. Data Preprocessing
5. Choosing the best model
6. Fine Tuning
7. Summary Generation

The application gets the speech as input from the microphone and the Google Speech to Text API converts the speech into text. This is provided as real time captions to the clients. The captions are appended into a transcript. Once the meeting gets over, the transcript is sent to the text summarizer for the summarization task which is given as notes to the clients.

3.1 Application Workflow

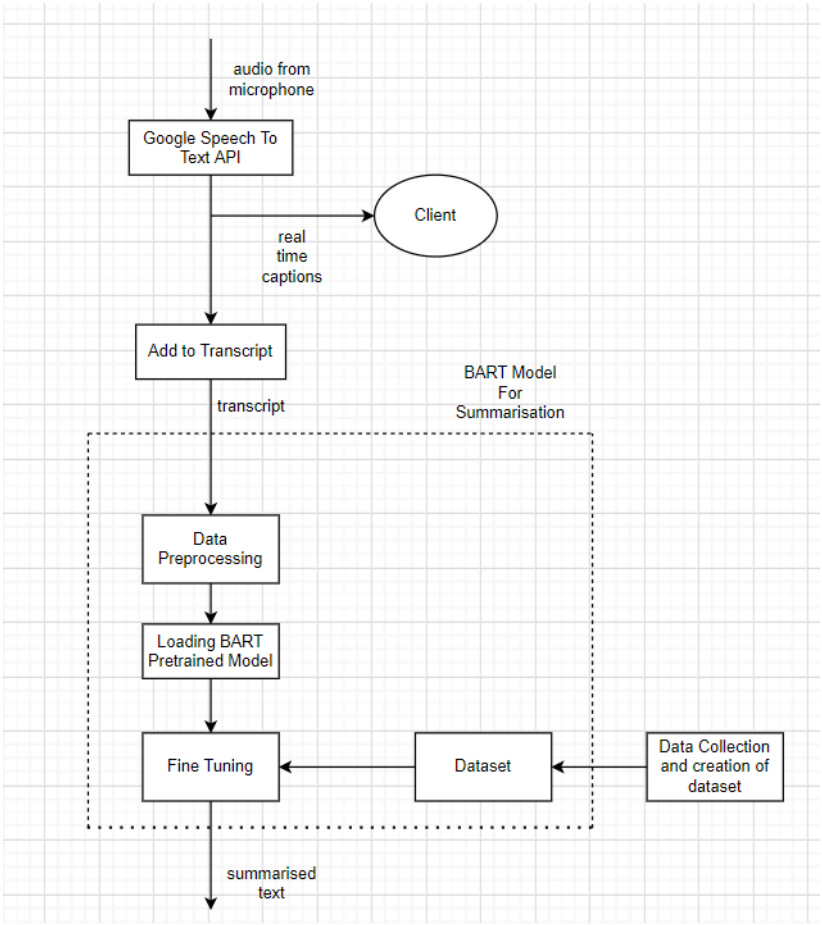


Fig. 1: Application workflow

Once the application starts, a new session can be started and the teacher can start talking. The speech to text transcription happens using the Google Speech to Text API. This real time caption is presented to the client in the application.

The captions are appended to a transcript which is used for creating notes at the end of the lecture. Once the meeting ends, the transcript is sent to the summarizer model which generates the notes. This flow is depicted in FIGURE 1.

Module Description

1. Google Speech to Text API
   The speech from the speaker's microphone is detected and sent to the Google Speech to Text API. The output is the real time captions which is sent to the clients. Along with this, the captions are added to the transcript.
2. Data Collection
   The audio from YouTube using YouTube Data API are collected. The audio files of 3 playlists were taken, namely Design and Analysis of Algorithms, Software Testing and Programming in C. A total of 182 audio files were collected. These audio files were given as input to the Google Speech to Text API and the corresponding transcripts were collected.
3. Data Annotation
   The transcripts collected from Google Speech to Text API is summarized manually. For each transcript, the corresponding summary is written and a csv file is made which constitutes (transcript, summary) pairs. This is the dataset which we use for choosing the summarization model. This dataset consists of 600 rows. Among these 600 rows, 300 rows are training set and the remaining 300 rows are testing set.
4. Data Preprocessing
   The dataset is loaded into a dataframe using pandas. The various steps done for preprocessing are
   - Cleaning The null values, escape characters, and symbols are removed. All the alphabets are converted into their lowercase letters.
   - Tokenization All the words are tokenized using a tokenizer and special tokens are added at the beginning and the end of each sentence. _START_is the start token and _END_token.
   - Stop words removal The words which don't add much value to the meaning are removed. Words like the, a, are, etc. are removed.
   - Stemming Its the process of converting the words into its root form. For example, eaten word is stemmed as eat.
5. Choosing the best summarization model The 3 models we are considering are BART, Sequence to Sequence model, and the TextRank algorithm. These 3 models are trained with the above annotated dataset of 300 rows. The 3 trained models are evaluated with the testing set of 300 rows using ROUGE score. The best performing model is chosen as our summarizer.

6. Fine Tuning and Building Dataset

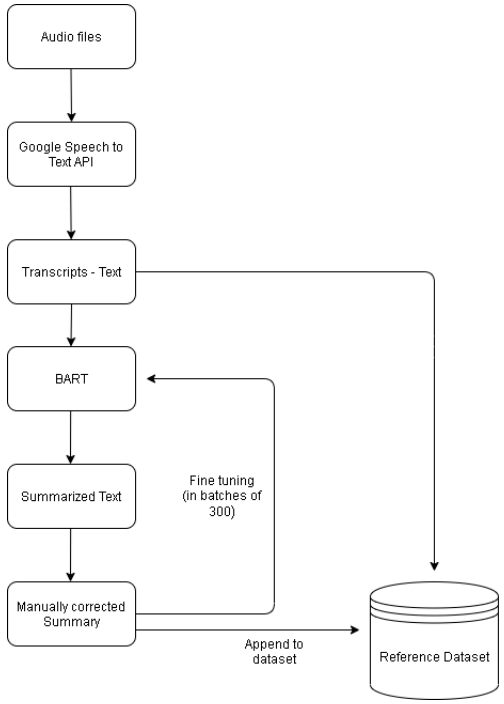


Fig. 2: Data Processing Flow

- Getting Audio Files: Audio files of the lectures/ online classes are obtained from NPTEL HRD Swayam Youtube page. The playlists included Design and Analysis of Algorithms, Introduction to Programming with C, Software testing.
- Google Speech To Text API: The input audio are files are given to Google Speech to Text which is pretrained model for speech to text, gives the transcripts for the input audio files.
- Transcripts: The Google Speech to Text API gives the transcripts in form of text files.
- BART Model: The BART model is initially downloaded from the huggingface transformers collection. It is pretrained on large Corpus of 16GB of data. The transcript is fed to this version of BART. And the output is obtained as summarized text.
- Manual Correction: This summary is manually edited to add on in the required words and remove the unwanted ones. This way, the BART model is fine tuned on the corrected version of the data and learns them over a period of time and over many iterations in batches of 300.

- Reference Dataset: This is built over many iterations in a semi supervised manner. This collected dataset can be used for future purposes. The same flow is depicted in FIGURE 2

7. Summary Generation: After fine tuning the BART model, the model is exported as pickle file and its used in the application.

3.2 Architecture Diagram

For the task of summarization, as shown in FIGURE 3, 3 models were considered, namely BART, Sequence to Sequence model, and TextRank Algorithm. The transcript from the lecture is sent to these summarizers and their outputs are evaluated. According to the results obtained, the best model is chosen.

The train dataset consists of 1800 instances of (transcript , summary). The BART and the Sequence to Sequence models are fine tuned using this dataset. The test dataset comprises of 300 instances of (transcript,summary). The Rouge scores of each model is found by comparing the predicted summary of each model with the target summary.

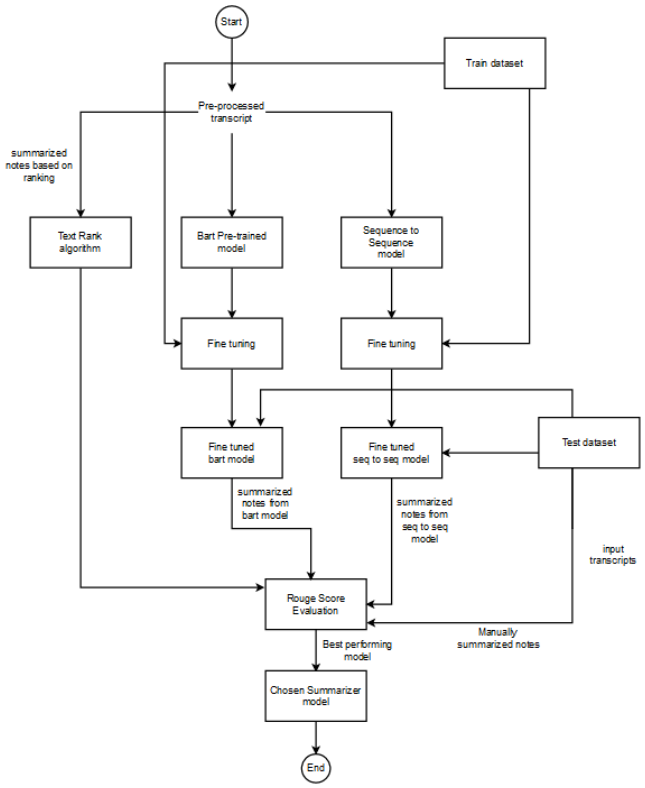


Fig. 3: Architecture Diagram

4 Implementation

4.1 Dataset

Dataset consists of output text files given by Google Speech to text API, when read through Lecture content of 82 videos. These videos were got from YouTube. The playlists that we used are Design and Analysis of Algorithms, Programming in C, Software Testing.

The input is fed into pretrained BART model from which intermediate output is obtained. This undergoes manual supervision, and the revisioned form is used as target summary for fine tuning the data in the next iteration.

4.2 Process of collecting dataset

Since the ground truth for the summaries of the video transcripts were not available, the dataset was created using semi-supervised learning method. The transcripts were downloaded and the summaries were created manually for 300 rows. Then the model was fine-tuned using this as the dataset. Then, another 300 rows were given to the model to predict the summaries. Then, the predicted summaries were manually validated and again the model was trained with the updated dataset. Likewise, this process was repeated for 2100 rows.

4.3 Experimental Setup

The environment and the various libraries and API used for the project are explained. Primarily Google Speech to Text API is used for speech to text conversion and Huggingface transformers are used for summarization by BART model.

Python v3.7 Python 3.7, the latest version of the language aimed at making complex tasks simple and is a minimum version requirement for BART finetuning.

Fastai v2 fastai is a deep learning library which provides practitioners with high-level components that can quickly and easily provide state-of-the-art results in standard deep learning domains, and provides researchers with low-level components that can be mixed and matched to build new approaches. It aims to do both things without substantial compromises in ease of use, flexibility, or performance. This is possible thanks to a carefully layered architecture, which expresses common underlying patterns of many deep learning and data processing techniques in terms of decoupled abstractions. These abstractions can be expressed concisely and clearly by leveraging the dynamism of the underlying Python language and the flexibility of the PyTorch library. Thse are internally used by the trasnsformers while finetuning

TorchText This library is part of the PyTorch project. PyTorch is an open source machine learning framework. The torchtext package consists of data processing utilities and popular datasets for natural language. This is internally used by transformer BART during the process of finetuning.

Huggingface Transformers The model was downloaded from huggingface transformers collection. The Hugging Face transformers package is an immensely popular Python library providing pretrained models that are extraordinarily useful for a variety of natural language processing (NLP) tasks. Features of huggingface transformers community includes:

- state-of-the-art NLP for everyone
- Deep learning researchers
- Hands-on practitioners
- AI/ML/NLP teachers and educators
- Lower compute costs, smaller carbon footprint

Google Colab Colaboratory, or "Colab" for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs. Google colab was used to fine tune the BART model over our dataset consisting of 2 x 2234 rows. The content consists of 120 videos of data related to computer science field, including software testing, Design And Analysis of Algorithms, Introduction to Programming with C.

Google Speech to Text An application(Videoder) used to download the audios in any given format from youtube servers for the required playlist. This was then given to the Google Speech to Text API available in Google Cloud Platforms, from which the transcripts for the whole audios were returned. This transcripts were in turn used to fine tune the BART model for the purpose of summarization.

4.4 Model Building

The explanation about the implementation is mentioned below. The link to access the notebook is also given at the last section of implementation.

Importing the packages The first step involved importing the ohmeow-blurr, datasets, bert-score packages.

Loading the dataset The dataset is stored into a dataframe from a csv file.

Data preprocessing The null values are removed using dropna() function in pandas. The tokenizer object of BART is then created which takes care of the tokenization process.

The dataset is broken into batches called datablocks for processing. The default argument values for the batch transform were given using the function default_text_gen_kwargs(). The arguments involve max_length, min_length, number of return sequences, padding token id, beginning of sentence token, end of sentence token, cachable and many more. To view full list, refer to the implementation link. Once all the arguments are assigned values, the BatchTransform constructor is called with the architecture object, configuration object, tokenizer as parameters. The data is split into batches called as datablocks.

Loading BART Pretrained model The bart pretrained model, architecture, configuration objects are loaded using the ohmeow-blurr library from huggingface.

Fine Tuning The BART Model is prepared for training by wrapping it in blurr's HF_BaseModelWrapper object and using the callback, HF_BaseModelCallback. A new HF_Seq2SeqMetricsCallback object can be used for specifying the Seq2Seq metrics, i.e Rouge.

A Learner object is created which is used for fine tuning. The constructor takes the datablocks, model loss function, callback object and splitter as parameters. Here, CrossEntropyLoss is used as loss function.

Then, we freeze our model so that only the last layer group's parameter is trainable. Then the fine tuning is done using a function named fit_one_cycle() with the number of epochs, learning rate and the callback object as the parameters.

Obtaining results The results are obtained using the show_results() function.

Exporting the model The model was be exported as a pickle model using load_learner() function.

Summarize using the model The blurr_generate(text) function can be used for summarizing the text.

5 Results and Performance Analysis

5.1 Rouge (metric)

ROUGE (Recall Oriented Understudy for Gisting Evaluation) is a set of metrics and a software package used for evaluating automatic summarization and machine translation software in natural language processing. The metrics compare an automatically produced summary or translation against a reference or a set of references summary or translation.

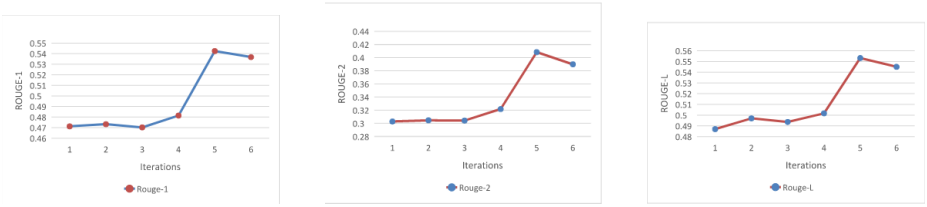
5.2 BART Model

The BART model is trained using semi-supervised learning. The train dataset consists of 1800 rows with transcripts, but not summary. So, the train dataset was divided into 6 blocks, each with 300 rows each. At each iteration, the predicted summary of the previous iteration model is manually corrected. This (transcript, summary) is used for fine tuning the next iteration BART model. The ROUGE score of the model got after each iteration is mentioned below. The results of the BART model at each iteration are shown below in TABLE 1. There is a good improvement in the Rouge scores after each iteration.

Iteration	Rouge-1	Rouge-2	Rouge-L
1	0.471191912847482	0.302830036989221	0.486972871206460
2	0.473210500215044	0.304601467425532	0.496998216586448
3	0.470217640597870	0.304199531679189	0.493619067435048
4	0.481398041623864	0.321774241800320	0.501677609785666
5	0.542424041559528	0.408314456403223	0.553228532702422
6	0.536735590282537	0.389777546282537	0.545027953284034

Table 1: Rouge Scores of BART at each iteration

The graph of the TABLE 1 is presented below for each type of the Rouge scores. The graph for Rouge-1, Rouge-2 and Rouge-L scores are shown in FIGURES 4a, 4b and 4c respectively.



(a) Rouge-1 (b) Rouge-2 (c) Rouge-L

Fig. 4: Rouge Score

Models	Rouge-1	Rouge-2	Rouge-L
BART	0.471191	0.302830	0.486972
Seq2Seq	0.191351	0.014464	0.203535
TextRank	0.463548	0.31577	0.521487

Table 2: Rouge Score Comparison of Models

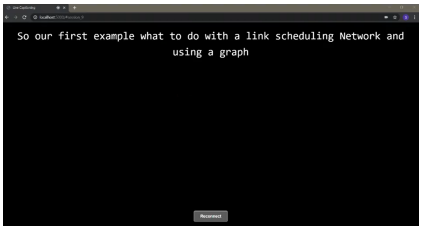
5.3 Inference

From the above results, for each iteration of the summarization, there is a gradual increase in the rouge score. The dataset is built progressively using semi-supervised learning. This shows that for a particular domain, our model performs well when there is more dataset. The ROUGE scores are calculated with the TextRank Algorithm, the Sequence to Sequence Model and with the BART model. The results are shown in TABLE 2. From the above results, the BART model outperforms and proves to be the best. Thus it can be shown that the semi supervised learning can be applied to domains that lack a good quantity of dataset to begin with, gradually building over time whilst also improving the ML model.

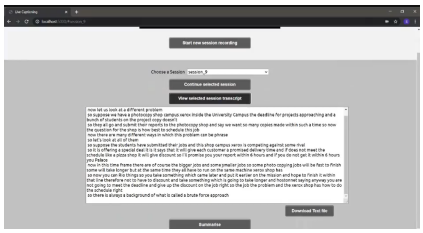
5.4 Screenshots of GUI

The Application Home Page is shown in Figure 5a. The new session can be started by clicking the "Start new session recording" button in the home page. Then a new screen is displayed where the real time captions are printed as shown in FIGURE 5b.

(a) Application Home Page



(b) After clicking start new session recording



(c) Displaying the transcript after the session ends

(d) Summarized notes of the transcript

Once the meeting ends, the session can be stopped by clicking at the text. Then the home page is returned. The transcript can be viewed by pressing the button "View selected session transcript" as shown in FIGURE 5c The transcript downloaded by clicking the button "Download text file". Then the transcript can be summarized by clicking "Summarize" button as shown in FIGURE 5d

6 Conclusion and Future Work

The design of the proposed system is discussed in the project. The application will perform real time captioning of lectures and provides summarized notes at the end. Google Speech to Text API is one of the tools available to perform speech to text conversion which is availed and used in the project. Similarly, the BART model for text summarization is used to provide the notes of the lecture at the end. The BART model for text summarization is used because it performs well for the dataset when compared to sequence-to-sequence model and text rank algorithm using rougescore evaluation metrics. Bart model performs with scores of 0.47, 0.30, 0.48 for rouge-1, rouge-2 and rouge-l whereas Sequence to Sequence performs with scores of 0.19, 0.01, 0.20 for rouge-1, rouge-2 and rouge-l respectively. Many different adaptations, tests, and experiments have been left for the future due to lack of time (i.e., the experiments with real data are usually very time consuming, requiring even days to finish a single run). Future work concerns deeper analysis of particular mechanisms, new proposals to try different methods, or simply curiosity. As a part of the future work, this project could be extended by integrating this application with the existing online meeting platforms such as Zoom, Google Meet and Microsoft Teams. This

involves exposing the application's service as an API across the web and performing seamless integration with online meeting platforms as depicted in Figure 6.

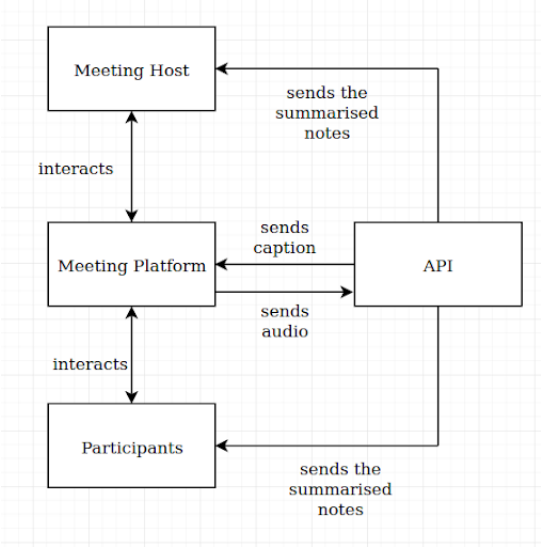


Fig. 6: Summarized notes of the transcript

References

[1] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed and Michael Auli (2020), 'wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations', Advances in Neural Information Processing Systems 33 ,pp 2541-2551.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Lilon Jones, Aiden. N. Gomez, Lukasz Kaiser and Illia Polosukhin (2017), 'Attention Is All You Need', Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 3862-3872.

[3] Chung-Cheng Chiu, Tara N. Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J. Weiss, Kanishka Rao, Ekaterina Gonina, Navdeep Jaitly, Bo Li, Jan Chorowski, and Michiel Bacchiani, 'STATE-OF-THE-ART SPEECH RECOGNITION WITH SEQUENCE-TO-SEQUENCE MODELS', IEEE Transactions and Learning Technologies, Vol 5(3), pp. 1206-1214.

[4] Manzil Zaheer, Guru Guruganesh , Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang and Amr Ahmed (2020) 'Big Bird: Transformers for Longer Sequences',Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 505-516.

[5] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov and Luke Zettlemoyer (2019), Facebook AI

'BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension', Association for Computational Linguistics (ACL), pp. 7871–7880.

[6] Takuya Yoshioka, Igor Abramovski, Cem Aksoylar, Zhuo Chen, Moshe David, Dimitrios Dimitriadis,Yifan Gong, Ilya Gurvich, Xuedong Huang, Yan Huang, Aviv Hurvitz, Li Jiang, Sharon Koubi,Eyal Krupka, Ido Leichter, Changliang Liu, Partha Parthasarathy, Alon Vinnikov, Lingfeng Wu,Xiong Xiao, Wayne Xiong, Huaming Wang, Zhenghao Wang, Jun Zhang, Yong Zhao and Tianyan Zhou-Microsoft (2019), 'ADVANCES IN ONLINE AUDIO-VISUAL MEETING TRANSCRIPTION', IEEE Transactions and Learning Technologies, Vol 4(2), pp. 1181-1192.

[7] Rada Mihalcea, Paul Tarau, 'TextRank: Bringing Order into Texts' (2004), Association for Computational Linguistics, pp. 404-411.

[8] Renals, S. & Simpson, M.N. & Bell, P.J. & Barrett, J.. (2016). Just-in-time prepared captioning for live transmissions, IET Publication pp 27 - 35.

[9] Rohit Ranchal, Teresa Taber-Doughty, Yiren Guo, Keith Bain, Heather Martin, J. Paul Robinson, and Bradley S. Duerstock. (2013), 'Using Speech Recognition for Real-Time Captioning and Lecture Transcription in the Classroom' IEEE Transactions and Learning Technologies, pp. 299-311.

[10] Shadiev, Rustam & Hwang, Wu-Yuin & Chen, Nian-Shing & Huang, Yueh-Min. (2014). Review of Speech-to-Text Recognition Technology for Enhancing Learning. Educational Technology & Society. 17. pp 65–84.

[11] Yang Liu and Mirella Lapata (2019), 'Text Summarization with Pretrained Encoders', Association for Computational Linguistics (ACL), pp, 3730-3740.

[12] Yu Yan, Weizhen Qi, Yeyun Gong, Dayiheng Liu,Nan Duan, Jiusheng Chen,Ruofei Zhang and Ming Zhou (2020), 'ProphetNet: Predicting Future N-gram for Sequence-to-Sequence Pre-training', Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 2401-2410.