This document is part of the Research and Innovation Action "European Live Translator (ELITR)". This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 825460.

Deliverable D6.5

Demonstrator of Automatic Minuting

Muskaan Singh (CUNI), Rishu Kumar (CUNI), Tirthankar Ghosal (CUNI), Ondřej Bojar (CUNI), Chiara Canton (PV), Andrea Sosi (PV), Adelheid Glott (AV), Franz C. Krüger (AV)

Dissemination Level: Public

Final (Version 1.0), 28th February, 2022

Grant agreement no.	825460
Project acronym	ELITR
Project full title	European Live Translator
Type of action	Research and Innovation Action
Coordinator	Doc. RNDr. Ondřej Bojar, PhD. (CUNI)
Start date, duration	1st January, 2019, 36 months
Dissemination level	Public
Contractual date of delivery	Former: 31st January, 2022; Updated: 28th February, 2022
Actual date of delivery	28th February, 2022
Deliverable number	D6.5
Deliverable title	Demonstrator of Automatic Minuting
Type	Demonstrator
Status and version	Final (Version 1.0)
Number of pages	24
Contributing partners	AV, PV, CUNI, KIT
WP leader	PV
Author(s)	Muskaan Singh (CUNI), Rishu Kumar (CUNI), Tirthankar Ghosal (CUNI), Ondřej Bojar (CUNI), Chiara Canton (PV), Andrea Sosi (PV), Adelheid Glott (AV), Franz C. Krüger (AV)
EC project officer	Luis Eduardo Martinez Lafuente
The partners in ELITR are:	• Univerzita Karlova (CUNI), Czech Republic • University of Edinburgh (UEDIN), United Kingdom • Karlsruher Institut für Technologie (KIT), Germany • PerVoice SPA (PV), Italy • alfatraining Bildungszentrum GmbH (AV), Germany
Partially-participating party	• Nejvyšší kontrolní úřad (SAO), Czech Republic

For copies of reports, updates on project activities and other ELITR-related information, contact:

Doc. RNDr. Ondřej Bojar, PhD., ÚFAL MFF UK
Malostranské náměstí 25
118 00 Praha, Czech Republic
bojar@ufal.mff.cuni.cz
Phone: +420 951 554 276
Fax: +420 257 223 293

Copies of reports and other material can also be accessed via the project's homepage:

`http://www.elitr.eu/`

Contents

1 Executive Summary **4**

2 Minuting Demonstrator Design **5**

3 The Minuting Pipeline **5**
3.1 alfaview® platform 6
3.2 Minuting REST API 7
3.3 Minuting Model 8

4 Accessing Minutes from alfaview Platform **10**

5 Conclusion **15**

References **15**

Appendices **16**

Appendix A A System Description Paper from AutoMin 2021 **16**

Appendix B Sample Outputs from our Minuting Model **24**

1 Executive Summary

This deliverable reports on the presentation platform for a minuting demonstrator developed during the ELITR (European Live Translator) project.

As required by task T6.1, the alfaview® platform has been extended and integrated with the PerVoice Service Architecture to deliver live transcription and translation to remote meeting participants. For the purposes of automatic minuting, these transcriptions are now further processed to produce the minutes, as reported in this deliverable. The feature has already been tested by the ELITR consortium and by alfatraining, an educational provider who uses alfaview®. All participants in the meeting are transcribed in real time, the transcript is repeatedly automatically summarized, and the live summary is made available to the participants by sharing a URL with them within the alfaview® platform.

The actual quality of the summary critically depends on the underlying summarization model and in this deliverable, we demonstrate that so far, the practical performance is severely limited by speech recognition errors and other issues.

From the technical point of view, PerVoice has developed a REST API to deliver the transcription of the meeting stream from the Alfaview platform. CUNI has implemented a minuting model, where the transcribed text of the meeting is summarized with a BERT-based model. It uses a similar processing of live transcription as used in translation. Further scripts integrate the minuting model to the system and repeatedly apply it on the transcribed text from PerVoice. The entire demonstrator was tested at an internal meeting between all the project partners of ELITR (namely AV, PV, KIT, and CUNI).

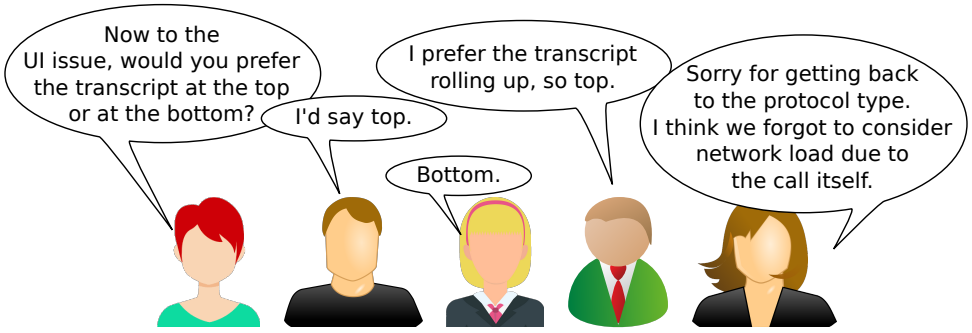
In Section 2, the main design for the demonstrator is described. Then in Sections 3 and 4, three different implementations are presented with their technical details.

2 Minuting Demonstrator Design

The automatic summarization of speech as explored in the ELITR project focuses on delivering minutes for a meeting in textual form.

Our envisaged minuting tool would consist of two components: (1) a user interface for writing meeting minutes, independent of automatic minuting, which may be used as such by human note-takers (2) automatic minuting software, ideally using the same user interface and helping the note-taker.

This ideal goal is illustrated in Figure 1: Participants' speech is recorded on the fly, with distinguishing participants in the meeting. Further, the transcript is manually corrected and fed into the model with an aligned hierarchical agenda ("empty agenda" in the following). The goal is to generate minute as summary of the transcript with the help of agenda (wherever possible). When specifying the difference between meeting minutes and text summarization, we explained that we prefer to keep all information, only deduplicate.



Original agenda as prepared by the organizer beforehand:
- Protocol type: push or pull?
- Layout of the user interface:
  - Transcript grows at the top or bottom of the document?
  - Or in a side pane?

Shared document, everyone allowed to edit.
Starts with the agenda and gets populated by Automatic Minuting (AM)
- Protocol type: push or pull?
  - (AM) > Pull easier to implement.
  - (AM) > Updates can get lost with push *in case the user*
  - (AM) > Consider network load.
- Layout of the user interface:
  - Transcript grows at the top or bottom of the document?
    (AM) > Top (AM) > Bottom (AM) > Top, transcript rolling up.
  - Or in a side pane?

Transcript, optionally editable to correct ASR errors:
11:03 Sorry for putting back to the protocol type. I think we forgot ...
11:02 I prefer the transcript rolling up, so top.
11:02 Bottom
...

Figure 1: Minuting Design

In practice, we fulfilled all the promised tasks but we did not get as far as this ideal suggested. For (1), we simply used standard shared documents such as Google Docs. While we considered to implement a Docs app that would live populate the docs with the transcript for manual revision, this was not promised in the project proposal and we put priorities to other tasks.

For (2), we wrapped, deployed and integrated our minuting models with alfaview, the conferencing used in ELITR, as described below.

3 The Minuting Pipeline

In the following sections, we technically describe the components, which are used to compose the minuting demonstrator. The full pipeline is sketched in Figure 2.

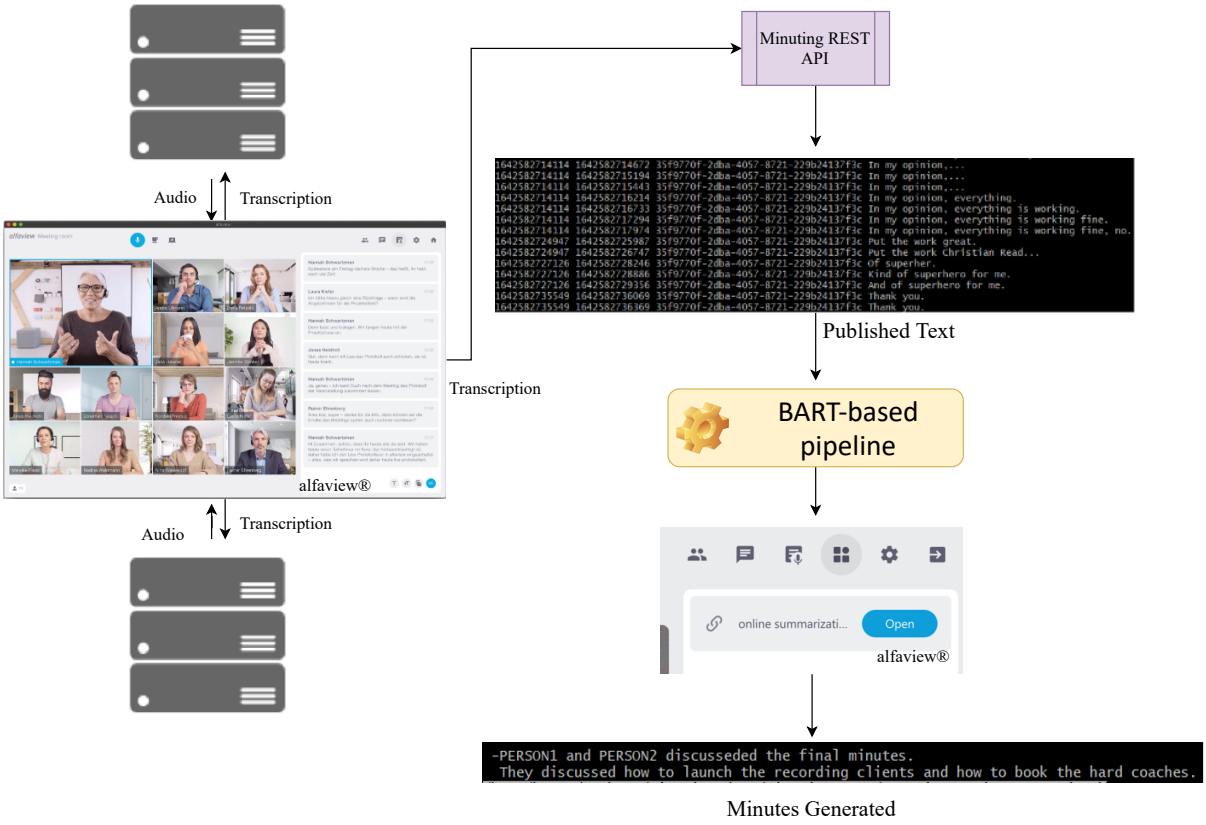


Figure 2: Minuting Pipeline

Initially, as the meeting takes place in the alfaview platform, the ASR worker and PerVoice Service Architecture (please refer to D6.1) provide the transcription of the audio (as described in Section 3.1). The transcription is further passed to the Minuting REST API (refer to Section 3.2). It is exposed to a REST endpoint used by the alfaview platform to send timestamps, speaker identification, and transcription data. These data are saved in text files on the server. This text file (published text) is passed to the summarization model (Section 3.3) to generate a summary of the meeting. The summary is generated on the server which offers it as a simple web page. Upon every reload, an updated summary is available.

To simplify users' access to the generated summary, the link to the live web page can be added to the running AV meeting using the toolkit option at the AV platform as discussed below in Section 4.

3.1 alfaview® platform

alfaview® is a GDPR compliant video conferencing software. With alfaview®, 200 or more people can stably communicate with audio and video simultaneously in every room, in high video quality, worldwide and, in real time. For larger meetings and events, even more people can participate in the spectator mode.

The alfaview® client sends the audio streams to the alfaview® service architecture. Dedicated microservices re-stream the audio to all connected participants and forward it to the PerVoice service architecture via the PerVoice client library for further processing. The PerVoice Service Architecture provides a central coordination point, called the mediator. alfaview® integrates the implementation of software modules called workers. In this case, two worker modules are required:

- ASR: Process and transform audio into a textual transcript,
- Text Recording: Provide the ASR result as a file stream for the summarization.

In addition to this, the alfaview® client links the final minuting result via the toolbox section in the sidebar. The link points to the minuting output hosted on CUNI servers.

We do not describe the ASR service here, it has been described in D6.1.

The "text recording" service is achieved using our novel minuting REST API described Section 3.2 below.

3.2 Minuting REST API

The Minuting REST API exposes a REST endpoint used by alfaview® platform for sending timestamps, speaker identification, and transcription data. This data is stored in text files on the server. The API exposes a POST endpoint **http(s)://<server-address>:<server-port>/saveSession**. The endpoint configuration depends on how the APIs are configured, it accepts JSON data in the payload of the request. An example of the payload is here:

Listing 1: POST request JSON payload example

```
{
    "sessionId": "00000000-0000-0000-0000-000000000004",
    "speakerId": "00000000-0000-0000-0000-000000000003",
    "language":"en-UK",
    "text": "Text content",
    "start": "09/06/21-17:56:45.761",
    "end": "09/06/21-17:56:48.521",
    "accessToken": "bd968709-2a47-478b-bf81-111111111111"
}
```

The Minuting RESP API is shipped into a Docker container and can easily be installed on a server running Docker. For the installation and configuration a Docker compose file is used.

Listing 2: Minuting API docker-compose.yml

```
version: "3.3"

services:
  minuting-api:
    image: pvdockerregistryprod.azurecr.io/pv/elitr/minuting-api/
        stable:latest-SNAPSHOT
    hostname: minuting-api
    container_name: minuting-api
    restart: always
    ports:
      - "8085:8081"
      # Configuration port on SLT server (CUNI premises), the port
          8443 is exposed over HTTPS by NGINX
      # - "127.0.0.1:8443:8443"
    volumes:
      # Folder where to save the output data
      - /opt/minuting-api/minuting-data:/opt/minuting-data
      # Configuration file for the Minuting API
      - /opt/minuting-api/application.yml:/opt/application.yml
    networks:
      - minuting-api-net

networks:
  minuting-api-net:
```

The outputs generated by the Minuting API are saved into **/opt/minuting-api/minuting-data** folder. Each generated text file includes the session name it its filename, which means that it is possible to manage multiple sessions simultaneously.

Each line of the generated text file has the following format:

```
<start_time> <end_time> <speaker_id> <text>
```

An example of the output is following:

```
1641309498939 1641309502899 00000000-0000-0000-0000-000000000003
  Text content
```

The timestamps saved in the text files are the **start** and the **end** times received, expressed in milliseconds from the Unix epoch time. The Unix epoch (or Unix time or POSIX time or Unix timestamp) is the number of seconds that have elapsed since January 1, 1970 (midnight UTC/GMT), not counting leap seconds (in ISO 8601: 1970-01-01T00:00:00Z).

For building the project, Java 8 and Maven needs to be installed on the PC. Docker is also required to manage and run the Docker image. It is possible to download the source code repository from GitHub[1] and save it into a local project folder. Once everything is set up correctly, we can build the project using Maven with the following command:

```
mvn clean package
```

After the build is finished, build the Docker image by using the following command (make sure to use a proper tag name):

```
docker build -f docker/Dockerfile -t tagname .
```

If there is a Docker registry, it can quickly push the image.

```
docker push tagname
```

In case there is no Docker registry to push the generated Docker image, it is possible to save the image into a compressed file (make sure to install **gzip** before proceeding) by using the following command:

```
docker save tagname | gzip > tagname.tar.gz
```

The created **.tar.gz** can be moved to a server, where it can be loaded in the local registry using the following command:

```
docker load -input tagname.tar.gz
```

3.3 Minuting Model

The ASR outputs from the minuting REST API are received on a separate machine. In our particular instance, we use the machine called SLT running at CUNI premises.

As described in Section 5.3.1 of D6.1, we already have tools that convert the stream of updating transcript messages to a full transcript, namely the "online-text-flow events" script. A master script keeps running in the background, checking for changes in the ever-growing ASR output file of a particular session every 60 seconds. If new lines are added to the file, it processes them with online-text-flow events to get an updated transcript and runs the minuting model in the background. As further information keeps coming to the file, whenever the output is generated from the minuting script, it is locally version controlled with git to have a full log of the changing state and avoid data redundancy.

We briefly describe our minuting model here. For further details on the methodology please refer to the Appendix A.

[1] https://github.com/ELITR/Minuting-API

The first phase of our pipeline architecture comprises preprocessing and topical segmentation of meeting transcripts. It involves utterance-level separation and filtering. We filter specific words and several entities (speaker names, locations, vocal sounds, and rest) from every utterance via a customized "stopwords" set. The filtered text is cleaned and then tokenized with the help of the NLTK Library (Loper and Bird, 2002).

At this point, the residual utterances from the data either have a very lean word count or are just meaningless sequences of words that contribute to the context of some topic from the meeting. They are further passed through a multi-layered threshold to create a partition between such utterances. The border values of this threshold are determined based on (i) word count in the utterance and (ii) the presence of certain uncommon words that may contribute to a topic's context. The generated transcript concatenates the "roles" and corresponding "cleaned" forms of utterances that qualify the threshold. It preserves the fluency of data and eliminates potential redundancies from the transcript.

Further, generated transcripts are segmented on a token length restraint (1024 to be precise). The token limit is different for every transcript and is lower than the maximum input length of the summarization model. Generally, this value depends on the type of dataset used. A longer transcript means that a greater count of topics discussed, that the meeting involved long topic discussions, or both. Hence, a longer token length will intake a relatively longer chunk of the transcript and will best suit the summary in such cases.

In the second phase, we consider several SOTA text summarization models including BART (Lewis et al., 2019), Pegasus (Zhang et al., 2020), T5 (Raffel et al., 2019), Roberta2Roberta (Rothe et al., 2020) and a few more. We use the pre-trained models and fine-tune them on summarization datasets. We eventually selected BART as the best-performing summarization model for our pipeline approach. It is a denoising autoencoder for pretraining sequence-to-sequence model trained in two stages: (i) by corrupting text using an arbitrary noising function, which induces noise in a text, (ii) by teaching it to reconstruct the original text. It generalizes the use of bidirectional encoders from BERT (Devlin et al., 2018) and autoregressive decoders from GPT-2 (Radford et al., 2019). The text passed into the model is first processed by an encoder that reads the sequence, and the decoder generates the corresponding output in an autoregressive manner. These layers are further connected using a cross-attention mechanism. The decoder layer learns to focus on features from encoder outputs.

BART's ability to use bi-directionality when operating on sequence generation tasks is a crucial feature that further bolsters up BART to be used for text summarization. While BERT cannot adopt a bidirectional mechanism for sequence generation, BART exploits the GPT-2 architecture for predicting the following words with the help of words encountered previously in the current sequence. Hence, these combined embeddings are of great significance in BART's architecture. BART's architecture follows the conventional encoder-decoder approach. Basically, the encoder maps the input sequence $\mathbf{X_{1:n}}$ to a encoded sequence $\overline{\mathbf{X}}_{\mathbf{1:n}}$. The decoder then maps this encoded sequence $\overline{\mathbf{X}}_{1:n}$ and a target sequence $\mathbf{Y}_{0:m-1}$ to the logit vectors $\mathbf{L}_{1:m}$. The logit vector is further used to define the distribution of the target sequence $\mathbf{Y}_{1:m}$ conditioned on the input sequence $\mathbf{X}_{1:n}$ by applying a softmax. According to Bayes' Rule, for each new word, $\mathbf{y_i}$ is represented, see Figure 3.

Next, we utilize the fine-tuned summarization model for generating segment-wise summaries of the transcript. The segmented dialogue blocks are passed through the summarization pipeline, and segment summaries are obtained. These are again passed through a filter that tests the contextual relevance of each sentence, relative to the typical topics discussed during a meeting. The segment summaries are finally concatenated, pronouns are inserted wherever possible, their shorter versions replace specific phrases, and a uniform tense is enforced. Consecutive mentions of a speaker or set of speakers in the summaries generally indicate that this part of the summary corresponds to a particular part of the meeting. These speakers had an elaborate dialogue, usually involving one specific topic. With this assumption, supported by analyses of target minutes, the obtained summaries are then bulleted based on the number of times a set of speaker name(s) appear in consecutive lines from the summary.

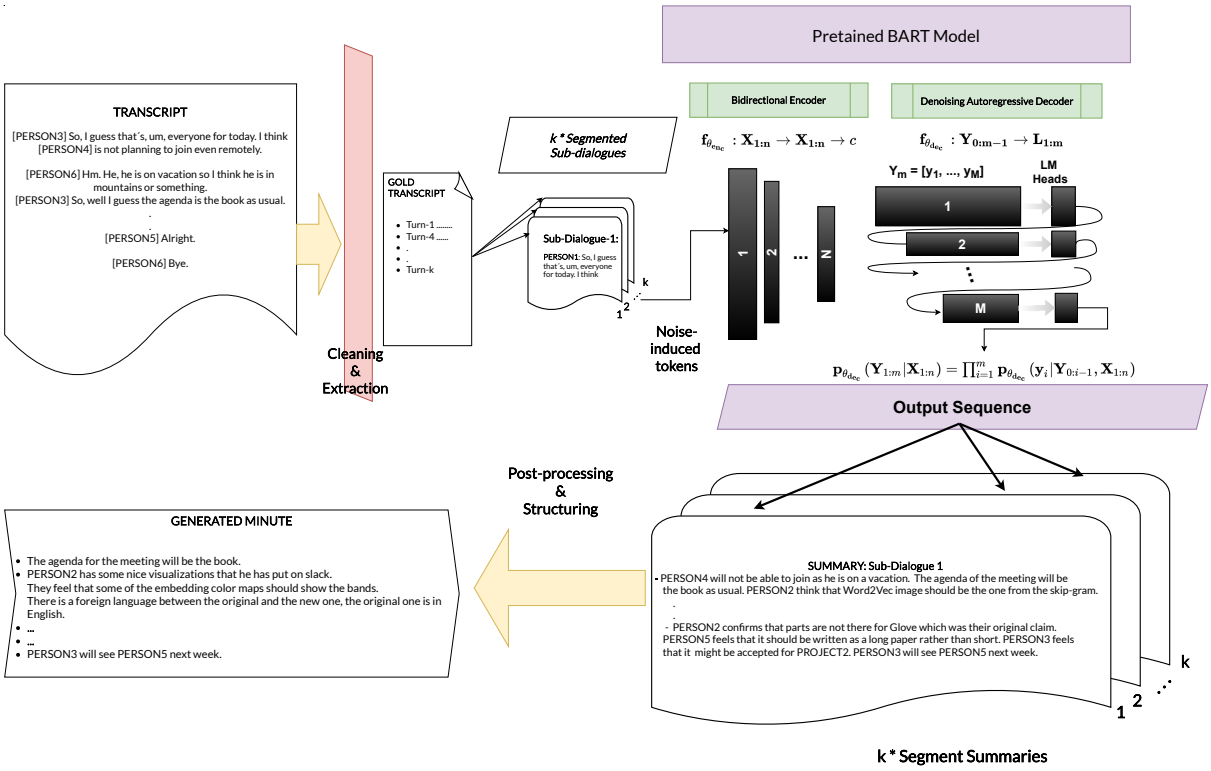


Figure 3: Overall architecture of the BART-based extractive-abstractive pipeline.

We leverage the findings from one of the top-performing systems in the AutoMin[2] shared task that we organized at Interspeech 2021. Please refer to Appendix A for our system paper that inspired the current minuting architecture.

Appendix B shows the output of the above the pipeline on our AutoMin dataset meetings. Please note that the outputs are based on manually revised transcripts. For all the above datasets, a good amount of manual processing has been done on the raw ASR transcripts to make them error-free. In real time, the ASR-generated raw transcripts may have certain additional signals, system variables, which can result in erroneous text generation. We plan to work on this shortcoming as our next step.

4 Accessing Minutes from alfaview Platform

The automatic minuting demonstrator provides live minuting for participants of online meetings based on automatic transcription.

The alfaview platform with live transcripts is illustrated in Figure 4. The left panel presents all the participants of the meeting with the speaker as highlighted in blue color. The center of the platform shows the shared screen of the participant in the meeting. The rightmost panel of the platform presents the generated ASR. It also contains speakers, messages, generated ASR, toolbox, setting, and leave the room options.

The endpoint used by alfaview® platform are exposed by minuting API from PerVoice service architecture. The API sends timestamps, speaker identification and transcription data. These data are saved in text files on the server as illustrated in Figure 5.

The output presented in Figure 5 is further fed into the CUNI server to pass the input to the minuting model. The details of the minuting model are described in Section 3.3.

Further, we have to solve a small technical issue: the minuting model is deployed on a machine which does not offer public web service. To make the minutes available to the users,

[2] `https://elitr.github.io/automatic-minuting/index.html`

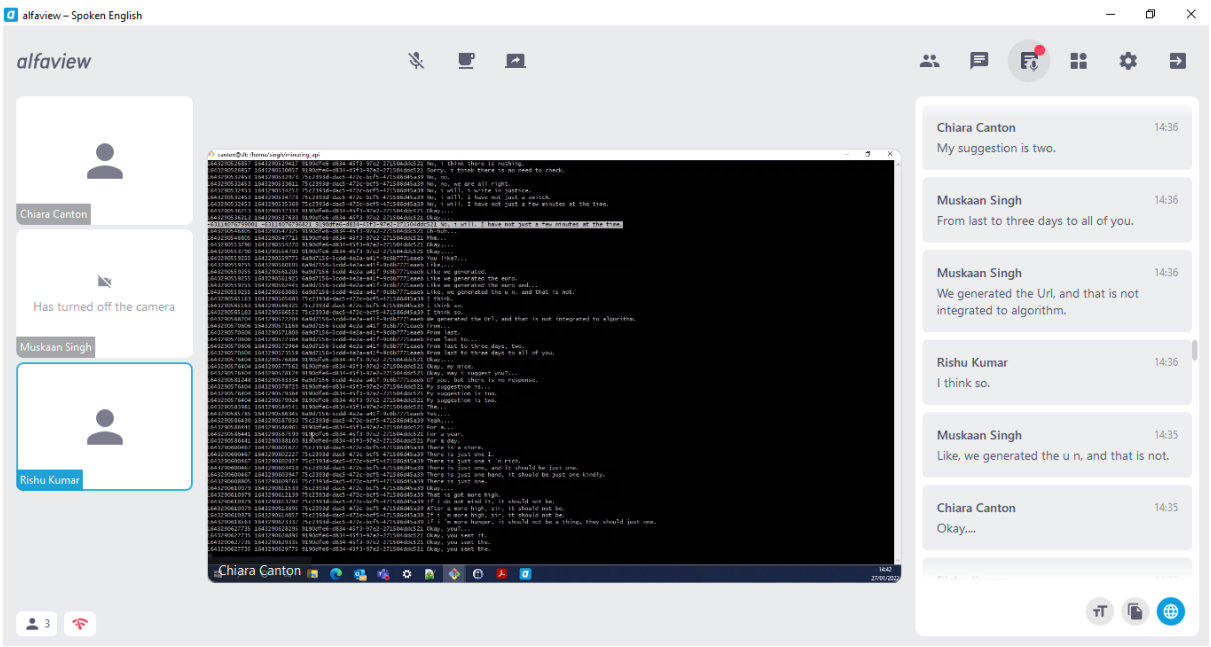


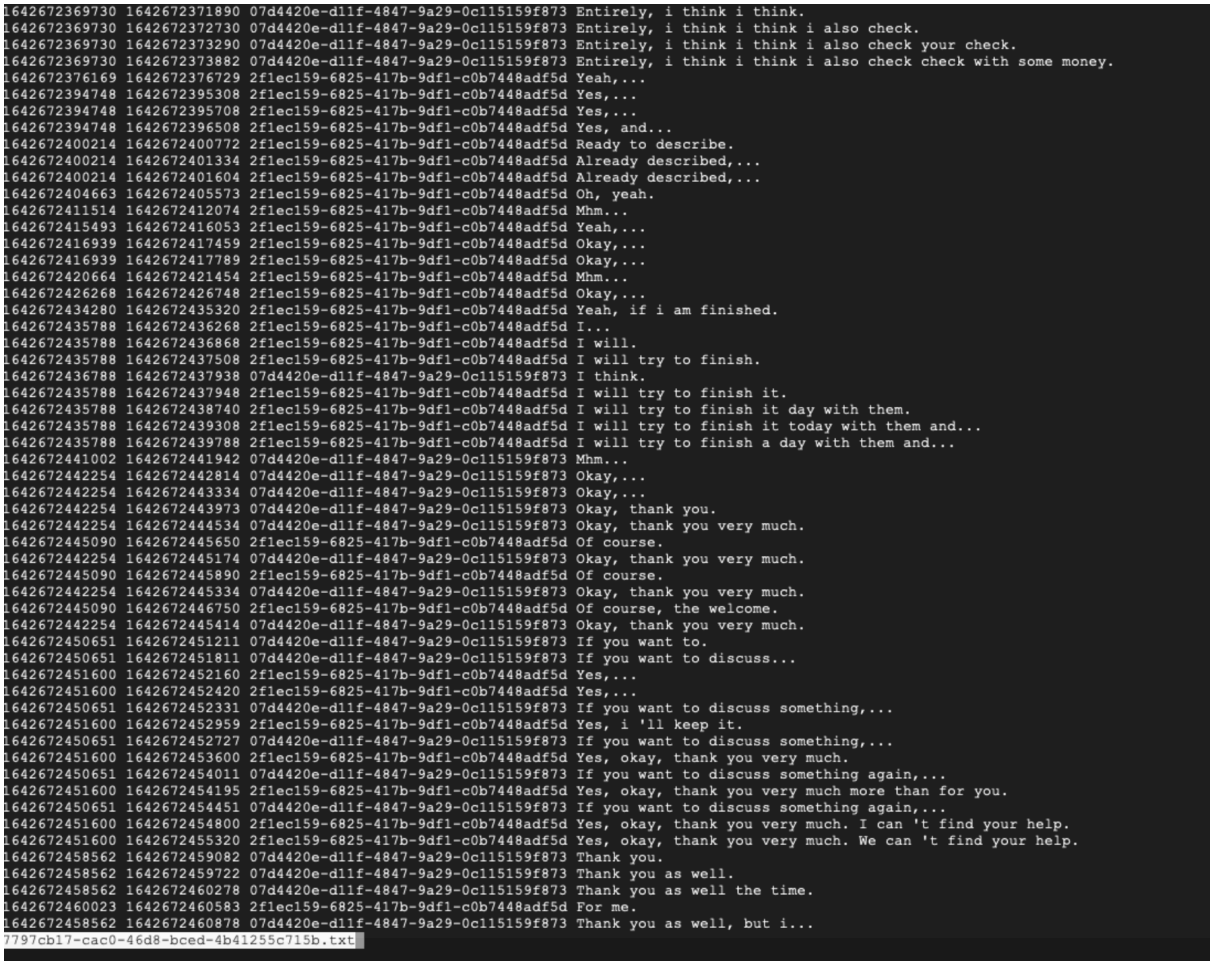
Figure 4: Alfaview meeting platform with live transcript displayed on the side.

we use *cron* to repeatedly copy the outputs to a publicly exposed directory, which can be accessed via a browser.

The minuting result is also accessible via the alfaview® toolbox. The toolbox can be managed by the moderators and administrators of an alfaview® room. Every alfaview® room has its own toolbox. The following steps are necessary to activate and manage the toolbox in alfaview®:

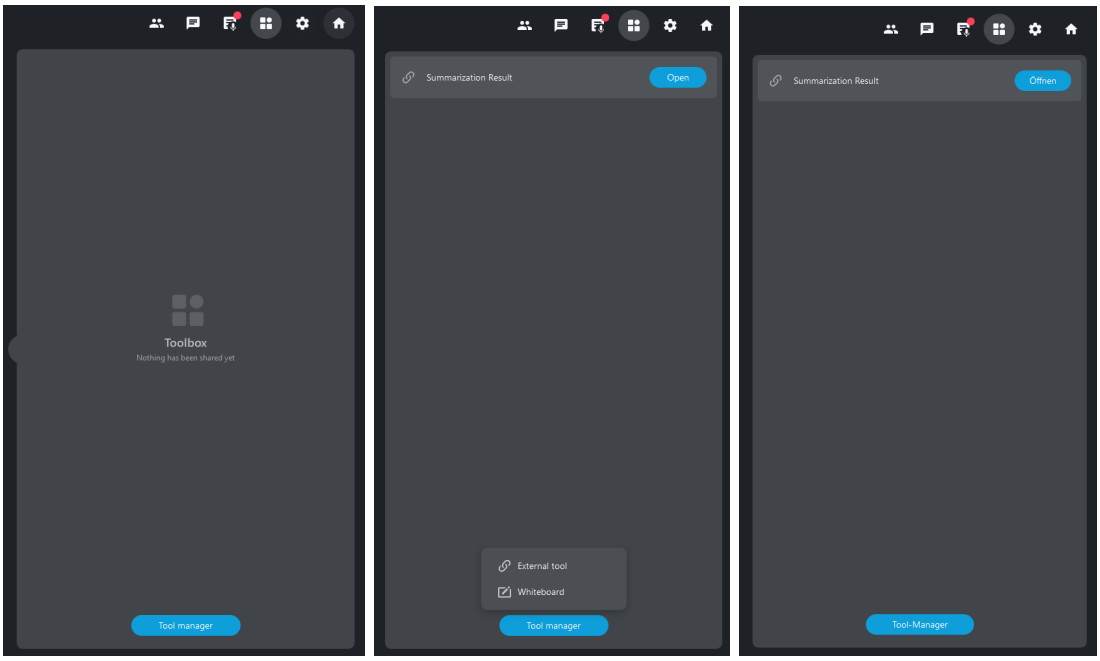
1. Click on the button in the sidebar to open the toolbox (Figure 6a):
2. As a moderator or administrator of the room, you will see the "Tool Manager" button in the lower section. When you click on this button following selection option appears (Figure 6b)
3. Click on the button "External tool". Now the Tool Manager opens in a standard internet browser (Figure 6c):
4. To integrate a new shared text document, click on the "Create" button in the upper right corner. Now enter the required information such as link title, details (optional), and address (URL link to the Google Docs sheet) and click on create. The link is generated in the alfaview® administration interface and can be used by all participants in the conference room. The link remains in the toolbox until removed or until the room itself is deleted (Figure 7).
5. By clicking on create, the link is created in the alfaview® administration interface and can be used by all participants in the conference room (Figure 8).

An example of the minutes generated from unedited transcript recorded using the described pipeline is provided in Figure 9. It is worth visually comparing the output quality with summaries created from manually revised transcripts (of different meetings) in Appendix B.



```
1642672369730 1642672371890 07d4420e-d11f-4847-9a29-0c115159f873 Entirely, i think i think.
1642672369730 1642672372730 07d4420e-d11f-4847-9a29-0c115159f873 Entirely, i think i think i also check.
1642672369730 1642672373290 07d4420e-d11f-4847-9a29-0c115159f873 Entirely, i think i think i also check your check.
1642672369730 1642672373882 07d4420e-d11f-4847-9a29-0c115159f873 Entirely, i think i think i also check check with some money.
1642672376169 1642672376729 2f1ec159-6825-417b-9df1-c0b7448adf5d Yeah,...
1642672394748 1642672395308 2f1ec159-6825-417b-9df1-c0b7448adf5d Yes,...
1642672394748 1642672395708 2f1ec159-6825-417b-9df1-c0b7448adf5d Yes,...
1642672394748 1642672396508 2f1ec159-6825-417b-9df1-c0b7448adf5d Yes, and...
1642672400214 1642672400772 2f1ec159-6825-417b-9df1-c0b7448adf5d Ready to describe.
1642672400214 1642672401334 2f1ec159-6825-417b-9df1-c0b7448adf5d Already described,...
1642672400214 1642672401604 2f1ec159-6825-417b-9df1-c0b7448adf5d Already described,...
1642672404663 1642672405573 2f1ec159-6825-417b-9df1-c0b7448adf5d Oh, yeah.
1642672411514 1642672412074 2f1ec159-6825-417b-9df1-c0b7448adf5d Mhm...
1642672415493 1642672416053 2f1ec159-6825-417b-9df1-c0b7448adf5d Yeah,...
1642672416939 1642672417459 2f1ec159-6825-417b-9df1-c0b7448adf5d Okay,...
1642672416939 1642672417789 2f1ec159-6825-417b-9df1-c0b7448adf5d Okay,...
1642672420664 1642672421454 2f1ec159-6825-417b-9df1-c0b7448adf5d Mhm...
1642672426268 1642672426748 2f1ec159-6825-417b-9df1-c0b7448adf5d Okay,...
1642672434280 1642672435320 2f1ec159-6825-417b-9df1-c0b7448adf5d Yeah, if i am finished.
1642672435788 1642672436268 2f1ec159-6825-417b-9df1-c0b7448adf5d I...
1642672435788 1642672436868 2f1ec159-6825-417b-9df1-c0b7448adf5d I will.
1642672435788 1642672437508 2f1ec159-6825-417b-9df1-c0b7448adf5d I will try to finish.
1642672436788 1642672437938 07d4420e-d11f-4847-9a29-0c115159f873 I think.
1642672435788 1642672437948 2f1ec159-6825-417b-9df1-c0b7448adf5d I will try to finish it.
1642672435788 1642672438740 2f1ec159-6825-417b-9df1-c0b7448adf5d I will try to finish it day with them.
1642672435788 1642672439308 2f1ec159-6825-417b-9df1-c0b7448adf5d I will try to finish it today with them and...
1642672435788 1642672439788 2f1ec159-6825-417b-9df1-c0b7448adf5d I will try to finish a day with them and...
1642672441002 1642672441942 07d4420e-d11f-4847-9a29-0c115159f873 Mhm...
1642672442254 1642672442814 07d4420e-d11f-4847-9a29-0c115159f873 Okay,...
1642672442254 1642672443334 07d4420e-d11f-4847-9a29-0c115159f873 Okay,...
1642672442254 1642672443973 07d4420e-d11f-4847-9a29-0c115159f873 Okay, thank you.
1642672442254 1642672444534 07d4420e-d11f-4847-9a29-0c115159f873 Okay, thank you very much.
1642672445090 1642672445650 2f1ec159-6825-417b-9df1-c0b7448adf5d Of course.
1642672442254 1642672445174 07d4420e-d11f-4847-9a29-0c115159f873 Okay, thank you very much.
1642672445090 1642672445890 2f1ec159-6825-417b-9df1-c0b7448adf5d Of course.
1642672442254 1642672445334 07d4420e-d11f-4847-9a29-0c115159f873 Okay, thank you very much.
1642672445090 1642672446750 2f1ec159-6825-417b-9df1-c0b7448adf5d Of course, the welcome.
1642672442254 1642672445414 07d4420e-d11f-4847-9a29-0c115159f873 Okay, thank you very much.
1642672450651 1642672451211 07d4420e-d11f-4847-9a29-0c115159f873 If you want to.
1642672450651 1642672451811 07d4420e-d11f-4847-9a29-0c115159f873 If you want to discuss...
1642672451600 1642672452160 2f1ec159-6825-417b-9df1-c0b7448adf5d Yes,...
1642672451600 1642672452420 2f1ec159-6825-417b-9df1-c0b7448adf5d Yes,...
1642672450651 1642672452331 07d4420e-d11f-4847-9a29-0c115159f873 If you want to discuss something,...
1642672451600 1642672452959 2f1ec159-6825-417b-9df1-c0b7448adf5d Yes, i 'll keep it.
1642672450651 1642672452727 07d4420e-d11f-4847-9a29-0c115159f873 If you want to discuss something,...
1642672451600 1642672453600 2f1ec159-6825-417b-9df1-c0b7448adf5d Yes, okay, thank you very much.
1642672450651 1642672454011 07d4420e-d11f-4847-9a29-0c115159f873 If you want to discuss something again,...
1642672451600 1642672454195 2f1ec159-6825-417b-9df1-c0b7448adf5d Yes, okay, thank you very much more than for you.
1642672450651 1642672454451 07d4420e-d11f-4847-9a29-0c115159f873 If you want to discuss something again,...
1642672451600 1642672454800 2f1ec159-6825-417b-9df1-c0b7448adf5d Yes, okay, thank you very much. I can 't find your help.
1642672451600 1642672455320 2f1ec159-6825-417b-9df1-c0b7448adf5d Yes, okay, thank you very much. We can 't find your help.
1642672458562 1642672459082 07d4420e-d11f-4847-9a29-0c115159f873 Thank you.
1642672458562 1642672459722 07d4420e-d11f-4847-9a29-0c115159f873 Thank you as well.
1642672458562 1642672460278 07d4420e-d11f-4847-9a29-0c115159f873 Thank you as well the time.
1642672460023 1642672460583 2f1ec159-6825-417b-9df1-c0b7448adf5d For me.
1642672458562 1642672460878 07d4420e-d11f-4847-9a29-0c115159f873 Thank you as well, but i...
7797cb17-cac0-46d8-bced-4b41255c715b.txt
```

Figure 5: Minuting API output as recorded from a meeting taking place in the alfaview® platform



(a) alfaview® Toolbox (b) alfaview® Tool Manager (c) Tool manager with link to summarization view

Figure 6: Adding the link to the live summary in alfaview® platform.

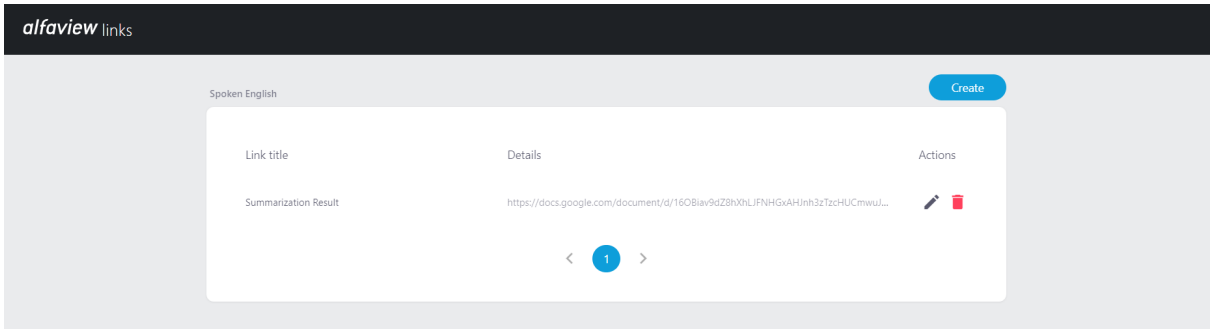


Figure 7: alfaview® toolbox links

Figure 8: Create link for the room

```
 -PERSON2, PERSON4 and PERSON1 will start with transcription by their systems.
 -PERSON4, PERSON4, PERSON4, PERSON1, PERSON7 and PERSON1 are talking about a business.
```

(a) 130 lines

```
 -PERSON7, PERSON6 and PERSON1 will start with transcription by their systems.
 -PERSON1, PERSON6 and PERSON7 are in a company.
 -PERSON1 has a few jobs coming up in the upcoming weeks where she can present her work
package and show it online.
 -PERSON4 was on one run on one random date.
  Company is slowly opening PERSON7, PERSON6 and PERSON1 left the system wrong.
 -PERSON6, who is for the follow-up, thinks it's too early and expensive.
  It's also possible that some products might not be available in the future.
```

(b) 250 lines

```
 -PERSON3, PERSON3 and PERSON1 will start with transcription by their systems.
 -PERSON1, PERSON1 and PERSON8 are in a company.
 -PERSON1 has a few jobs coming up in the upcoming weeks where she can present her work
package and show it online.
 -PERSON8 was on one run on one random date.
  Company is slowly opening PERSON8, PERSON1 and PERSON8 have problems with the system.
  There are too many gaps in the recording of the throne and it's too noisy to listen
to the audio files.
  There would be a few more days to finish PERSON3, PERSON A, PERSON B, PERSON C,
PERSON 1, PERSON 6, PERSON 7 and PERSON 8 discusseded the final report on summer
created.
```

(c) 340 lines

```
 -PERSON53, PERSON50 and PERSON54 will start with transcription by their systems.
 -PERSON54, PERSON5 and PERSON5 are in a company.
  They have a few jobs coming up in the upcoming weeks where she can present her work
package and show it online.
 -PERSON8 was on one run on one random date.
  Company is slowly opening PERSON54, PERSON54 and PERSON52 have problems with the
system.
  There are too many gaps in the recording of the throne and it's too noisy to listen
to the audio files.
  There would be a few more days to finish PERSON50, PERSON54, PERSON5, PERSON54 and
PERSON54 will review the final report on summer created.
 -PERSON5, Andrea, PERSON54, and PERSON540 are talking about the work package.
 -PERSON5 is having problems with the transcription on his computer.
 -PERSON540, PERSON5 and PERSON544 discussed how to improve the quality of their
messages.
  They0 explains how to process the Kodam sun containers.
```

(d) Complete transcript

Figure 9: Sample minutes generated automatically from unedited transcript. We show stages of the minutes from the first 130, 250 and 340 lines of the transcript.

5 Conclusion

This deliverable described the minuting demonstrator for the ELITR project. The alfaview® platform provides a meeting set up for all the participants to interact. The Minuting REST API by PerVoice exposes a REST endpoint used by the alfaview® platform to send timestamped transcription data to the CUNI server. This data is saved in text files on the server. Further, it is fed into the BART-based model to generate a summary for the meeting. This meeting summary, referred to as minutes, is offered to the users in the AV platform in the form of a link to a regularly updated web page.

The quality of the summary output is so far insufficient for practical purposes but it allows us to experiment further and assess the usefulness of minuting in different application settings.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

Edward Loper and Steven Bird. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*, 2002.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280, 2020.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020.

A A System Description Paper from AutoMin 2021

Team ABC System Run for AutoMin Shared Task @ INTERSPEECH 2021

Kartik Shinde[1], *Nidhir Bhavsar*[2], *Aakash Bhatnagar*[2], *Tirthankar Ghosal*[3]

[1]Indian Institute of Technology, Patna, India
[2]Navrachana University, Vadodara, India
[3]Institute of Formal and Applied Linguistics, Faculty of Mathematics and Physics, Charles University, Czech Republic

kartik_1901ce16@iitp.ac.in, 18103488@nuv.ac.in, 1824526@nuv.ac.in, ghosal@ufal.mff.cuni.cz

Abstract

This paper introduces the approach of team ABC for the first shared task on Automatic Minuting (AutoMin) @ Interspeech 2021. This shared task aims to create minutes from multiparty meetings. Automatic Minuting is a challenging task in the domain of natural language processing and sequence-to-sequence transformation. In Task A, we generated rational minutes from a given meeting transcript by developing a system that utilizes the knowledge of pre-trained language models to generate dialogue summaries. Our model splits the given transcript into multiple pairs of utterances and roles (speakers). These pairs are then summarized independently by a pretrained BART model. Our proposed system generates the briefest and detailed summary of meeting transcripts concerning coverage, adequacy, and readability. The result for this task was amongst the best in Human Evaluated Scores: Adequacy: 4.46/5.00, Grammatical: 4.45/5.00, Fluency: 4.18/5.00, and Automatic Evaluated Scores: ROUGE-1: 0.28, ROUGE-2: 0.07, ROUGE-L:0.16. Furthermore, We used six similarity matrices to determine whether or not a derived minute is from the given transcripts or the given minute (Tasks B and C). The accuracy scores of tasks B and C were 95.00% and 91.00%, respectively.

Index Terms: automatic minuting, summarization, topic segmentation

1. Introduction

Since the pandemic, most of our interactions were virtual, and hence the automatic support was needed for the smooth running of online events and meetings. Frequent remote conferences and day-to-day video calls demand an efficient and effective functioning system to document these interactions. Hence, the requirements rose for the systems to automatically produce a concise and coherent summary of a given meeting transcript. Meeting Summarization is considered one of the most complex and compelling research topics in natural language understanding and machine learning. The survey shows that summarizing meetings into structured minutes of speech saves up to 80% of an annotator's time. Although 'Meeting Summarization' and 'Automatic Minuting' appear to be the same task, automatic minuting has several hidden challenges along with its primary objective to generate meaningful minutes. The primary obstacle in automatic minuting lies in the variability of the structure and fabrication of transcripts in their generated minutes. There is no universal framework for creating minutes, and it varies across different types of meetings, subjects, and objectives. Hence, one annotator might create minutes different from the second one. This variation can be found in terms of - format, length, use of novel words, ignorance towards trivial details, and summary briefness for some discussed topics. Besides these differences, a minute of a meeting is also judged based on its adequacy, readability, semantic meaningfulness, clarity, coverage, and grammaticality. Table 1 shows the task-wise goals for this shared task. We build a system that generates, analyzes, and compares meeting minutes

Table 1: *Task-wise challenges of the shared task*

Subtask A: **(Generation)**	Transcript → Minute
Subtask B: **(Verification)**	Transcript + Minute → True/False (true corresponding to a pair of matching transcript and minute, and vice versa)
Subtask C: **(Comparison)**	Minute + Minute → True/False (true corresponding to a pair minutes that belong to the same transcript)

AutoMin @ Interspeech 2021 has provided a platform for all these innovations in meeting summarization using natural language processing. This shared task consisted of one central task and two supporting tasks for different meeting scenarios (technical meetings and parliamentary proceedings). Moreover, the technical meetings data included transcripts of two languages, English and Czech. The first shared task on Automatic Minuting aims to create minutes from multiparty meetings. The objective was to strengthen the community's interest in attempting this exciting problem and unfold the challenges towards creating a unified framework for automatic minuting.

2. Related Work

Automatic Minuting is very closely related and often linked with the task of Meeting Summarization. In recent years, there has been immense research on summarization and text-to-text generation models. This section focuses on some of the best approaches in the field over the past few years. The following is an overview of the current state-of-the-art strategies deployed by various models.

2.1. Pegasus

Pegasus[1] is a pre-trained large transformer-based encoder-decoder model on massive text corpora with a new self-supervised objective. In PEGASUS, important sentences are removed/masked from an input document and are generated together as one output sequence from the remaining sentences, similar to an extractive summary. It has achieved state-of-the-art performance on all 12 downstream datasets measured by ROUGE[2] scores, and also an excellent performance on low-resource summarization, surpassing previous state-of-the-art results on 6 different datasets with only 1000 examples. The base architecture of PEGASUS is a standard Transformer encoder decoder. The (Gap Sentence Generation) GSG and (Mask Language Model) MLM components are executed simultaneously in the model's framework.

2.2. T5

T5[3] is an overly vast model which, again, has achieved the state-of-the-art in various NLP tasks. It generalized the conventional text-to-text framework to suit a variety of challenges in the domain of natural language understanding T5 also explained the advantage of scaling up the model size (to 11B parameters) and pre-training corpus, by introducing C4 (a massive text corpus which is derived from Common Crawl). T5 was pretrained on randomly corrupted text spans using different combinations of mask ratios and sizes of span.

2.3. HMNet

HMNet[4] is an end-to-end deep learning framework. Hierarchical Meeting summarization Network (HMNet) leverages the encoder-decoder transformer architecture (Vaswani et al., 2017)[5] to produce abstractive summaries based on meeting transcripts. As we discussed earlier, meeting transcripts are usually lengthy, a direct application of the transformer structure is not feasible. In addition, the accommodation of multispeaker scenarios casts a complex challenge for the model. Hence, to adapt the structure to meeting summarization, HMNet exploits the hierarchical structure and carries out both, token-level and turn-level understanding across the entire transcript. Simultaneously, it also makes use of a role vector for each meeting participant to represent the speaker's information during encoding. This role vector is appended to the turn-level representation for later decoding.

2.4. BART

BART[6] is a denoising autoencoder for pretraining sequence-to-sequence models.BART uses a standard Transformer-based neural machine translation architecture. BART has proved to be a versatile, yet simple, breakthrough in transformers and is extremely effective when fine-tuned for text generation. It matches the performance of RoBERTa with comparable training resources and achieves new state-of-the-art results on a range of abstractive dialogue, question answering, and summarization tasks. BART has outperformed the last state-of-the-art models on many tasks. Hence, we have utilized the versatility of BART to tackle the main challenge in this Shared Task.

Architechture of BART -

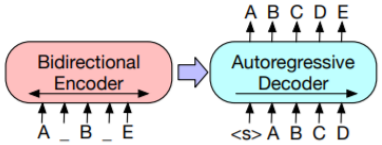


Figure 1: *A representation of BART from (Lewis et al., 2019), here the input need not to be aligned with the decoder output, allowing arbitrary noise transformation.*

BART uses the standard sequence-to-sequence transformer architecture from (Vaswani et al., 2017), except the ReLU activation is replaced with GeLU (Hendrycks & Gimpel, 2016)[7]. The base model uses 6 layers in the encoder and decoder, And, the architecture is closely related to that used in BERT, except, a) each layer of the decoder additionally performs cross-attention over the final hidden layer of the encoder, b) BART does not use an additional feed-forward network before word prediction (see Figure 1).

3. Task and Dataset Description

Task A defines the primary objectives for this shared task. It requires us to create minutes from multiparty meeting transcripts automatically. The evaluation of this task will consist of both manual and automatic assessments. The most critical aspects of the automatically generated minutes would be adequacy, relevance, coverage, readability, and grammaticality. At the same time, the model may not pay attention to textual coherency since meeting minutes are not always supposed to have a coherent textual form.

For the first subtask, the dataset provided on the Official GitHub Repository of AutoMin 2021 has 85, 10, 28 instances in its TRAIN, DEV, and TEST directory, respectively. Each instance consists of : (i) a meeting transcript; (ii) one or more annotated meeting minutes corresponding to the meeting transcript. We have used several methods to separate unique entities(e.g. PERSON, ORGANIZATIONS, PROJECTS) and dia-

logues from the speaker's context.

A single transcript consists of turnwise separated dialogues. Every utterance is mentioned inside curved brackets/parentheses "()", and is located at the beginning of that particular dialogue. Any further mention of unique entities in an utterance can be distinguished using the help of square brackets/parentheses "[]". The transcripts also consist of pointed brackets/parentheses "⟨⟩" to show pauses, sounds, or any unintelligible information

While there is no informed format for a generated minute, almost every minute contains these vital components - Date of the Meeting, attendees, Agenda, Bulleted Minutes, annotator of the corresponding minute. Minutes for one particular meeting transcript, annotated by different annotators, may not have coherence or congruence. At the same time, the format of minutes may be very distinct.

4. System Description

Our system efficiently utilizes the knowledge from pretrained language and summarization models via transfer learning. In the below sections, we present various strategies to solve tasks A, B, and C. We discuss the various methodologies which stand out and explain the details of our model for each task. Our systems vary for each subtask, and they can be combined in the desired manner to create an end-to-end pipeline.

4.1. Subtask-A

For the first subtask, we build a system that efficiently summarizes any given multi-party conversation (here, meeting transcripts). From the above-identified pre-trained frameworks, we chose to leverage the versatility of BART, combined with transfer learning to mend the weights to suit our methodology.

Our method employs a pretrained BART (Lewis et al., 2019) transformer model, with a denoising autoencoder architecture, since we have opted for a sequence-to-sequence approach for the task. The model that we use is a BART architecture - "facebook/bart-large-xsum", provided by HuggingFace Transformers[8] available here[1][2], for the fine-tuning task,while the dataset chosen for evaluation is the SAMSum Corpus, released in Nov, 2019.

SAMSum Corpus - The SAMSum dataset[9] contains about 16k messenger-like conversations with summaries. These conversations were written by fluent English linguists. These dialogues are similar to the conversations one experiences daily, and reflect the proportion in the topics of such real-life scenarios. The style used is very diversified, as the dataset includes all - casual, semi-formal, and formal conversation threads. Some of them also show the use of slang, common typos, and emoticons. Every conversation is annotated with its summary.

Other datasets that can be considered for the task are - DialogSum[10], MediaSum[11], XSum[12] and Spotify Podcasts Dataset[13]. The SAMSum corpus provides multiline summaries for short-length conversations and includes modern slang, acronyms, and abbreviations alongside. Therefore, turn out to be the most suitable dataset for this task. DialogSum has similar specifications, the conversations are longer, and summaries take into account the sentimental contexts of the speakers as well (e.g. "Mark is angry about the new reforms", while the word 'angry' was never a part of the original conversation), which is not an important information for the generated minutes. On the other hand, XSum provides one-sentence summaries to answer the question - "What is the article about?". MediaSum is a large-scale media interview dataseet consisting of 463.6K transcripts with abstractive summaries.

Compared to other corpora, MediaSum is significantly larger and contains 'complex, multi-party conversations'. It includes conversations from diverse domains and covers a range of topics and long spans. MediaSum stands as the second most suitable alternative for the task between SAMSum and MediaSum. For further details refer to our models and code.[3].

Segmentation - - To tackle the main challenge of Automatic Minuting, one of the conventional approaches is employing topical segmentation. A transcript is segmented into topics, which is further reduced to queries and corresponding summaries are acquired. Normally, a transcript encompasses several topics, distributed sparsely, combined with casual dialogues, miscellaneous chats, and superfluity. Furthermore, these topics might be interrelated and can occur multiple times across the document. One might be able to observe that these topics, generally get interlinked and consequently get overlapped during discussion. Thus a definite conclusion would be that meeting transcript, in real-world scenarios, does not adhere to any type of conventional structure. Instead, transcripts are discrete and deficient, due to human indulgence and could cause difficulties in modeling relevance and salience. Our proposed model comprises segmentation based on token length. With experimentations, we concluded that a threshold must be calculated to limit the number of tokens in a segmented block of conversation, which in turn would help in minimizing the effect of contextual interdependency across distinct blocks of conversation. The above-mentioned redundancies are supervised using the summarization model and pre-defined filtering rules applied to post summary generation. Semi-supervised methods are adopted to tackle the problem of minute structure and the issue of constituting topics. Additionally, rules are formulated by observing generated summaries for further increase in performance.

4.2. Subtask-B

The challenge in Task B is, given a pair of meeting transcripts and a minute, we have to identify whether or not the minute belongs to the corresponding transcript. The task helps in deciding the similarity in the context of both documents. This task may be useful in early predictions to check whether the meeting

[1] facebook/bart-large-xsum
[2] lidiya/bart-large-xsum-samsum

[3] github.com/cruxieu17/automin-2021-submission

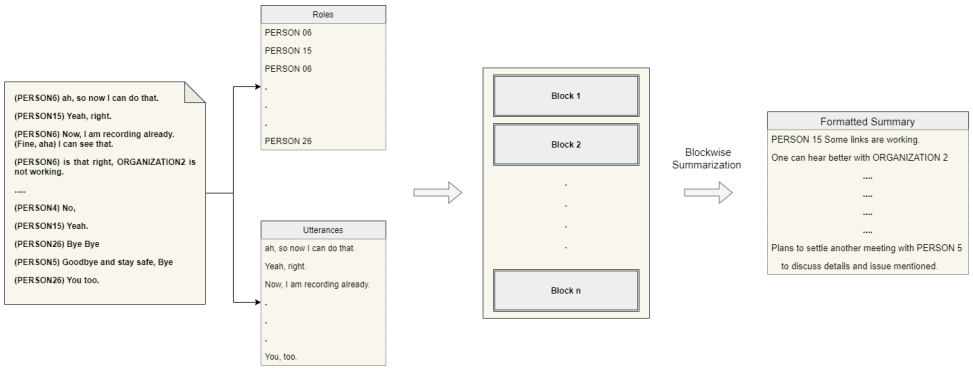


Figure 2: *The outer architecture of our proposed system for Subtask A*

minutes belong to given transcripts or not.

The dataset provided for this subtask consists of 846 instances including both TRAIN and DEV (each accounting for 566 and 280, respectively). After cleaning of faulty/empty instances, it leaves us with 843 entries in total. see Table 2

The next step involves the filtration of the textual data. We clean the transcript and the minute and remove the irrelevant details captured during transcription. The motive is to extract unique words and details that are generally 'exclusive' for a transcript. For this task, we use the NLTK library[14] and used POS tagging to extract such parts from the text. We define 6 scores as mentioned in Section 5.2 to create the data that we can use for further experimentation.

Dataset	True tag	false tag	**Total**
Task B	115	731	**846**
Task C	74	660	**734**

Table 2: *Class-wise distribution of train and dev data.*

4.3. Subtask-C

The challenge in this subtask is to identify whether the provided pair of minutes is relevant for the same meeting transcript or not. When a given transcript is minuted separately by multiple annotators, there can be a quite significant difference among the annotated minutes. The minutes may vary in multiple aspects and one would probably get confused whether they answer to the same transcript or not. This subtask may sound similar to the previous subtask, but it is not the same challenge. There is a slight difference when 'comparing' minutes, instead of verifying them for particular transcripts. The data allotted for this subtask, after cleaning and deletion leaves a total of 993 instances of labeled data for experimentation.

In the next step, data was further refined by removing all undesired characters and slang. POS tagging was executed, and the text was filtered to obtain the information needed to generate a usable dataset.

The dataset had a distinctly visible imbalance with "TRUE" labels being 74, and "FALSE" labels 660 in the count. Due to this imbalance, the recall and precision values of classifiers were significantly low. To mitigate this problem, we chose to perform oversampling on the minority class from the dataset. After an extensive examination, SVM SMOTE from Scikit-learn[15] proved to be the most effective technique for this dataset. see Table 2

We perform scoring of the extracted pair of texts based on different methods, and get a 1 x 6 dimensional vector corresponding to each instance. This generates an equal number 'TRUE' and 'FALSE' samples, i.e., 562; ready to be fed to a suitable classifier model.

5. Experiment and Evaluation

5.1. Subtask-A

In our approach, we carried out fine-tuning process with parameters: 'max_input_length' to 512, 'min_target_length' to 128. The training data for this method is SAMSum corpus, with batch size equal to 4. Subsequently, we initiated the 'Seq2SeqTrainer'[16] class with an 'LR' of 2e-5. We then calculate ROUGE scores of the output summaries for evaluation.

We have conditioned that the transcripts are easily distinguishable for experimentation on distinct fine-tuned models and should have a proper arrangement or format. For this purpose, we perform turn-level separation and cleaning of the dialogues in the transcripts. Then, we extracted the mentions of all the speakers, organizations, and projects across the transcript. We stored this data in a dictionary format and iterated at turn-level again before feeding the conversation threads to the model.

Influence of datasets, over the performance of the summarization model - For Experimentation, we have used two types of datasets - (i) Dialogue Summarization Dataset, and (ii) Text summarization datasets with a low target to input length ratio.

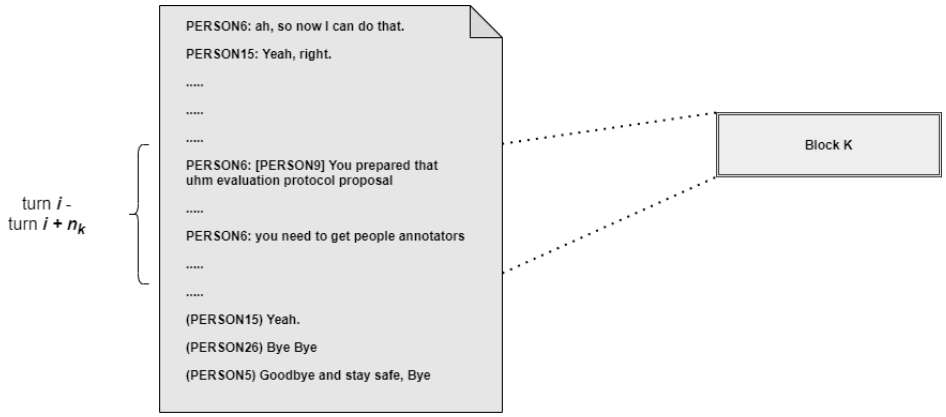


Figure 3: *Illustrates the segmentation procedure applied to transcripts, as shown above, Block K represents the conversational thread which comprises of n_k conversational turns starting from the i^{th} position in a transcript.*

The contemporary summarization models are not trained for interpreting and efficiently summarizing dialogues. Thus while evaluating a fine-tuned model, the generated summaries might show a lack of fluency and adequacy to a significant extent. Datasets similar to XSum, CNN/DM, have a low "target to input length" ratio and serves the purpose of adjusting the weights and hence sufficing the proportion required to generate adequate minutes. Lower generated summary length implies a powerful discriminating model, thus fine-tuning on these datasets shortens the summary length by manipulating the model to consider important information. Moreover, our experiment has precisely interpreted the difference in summaries generated by the model, which were fine-tuned on selected datasets, prior to the main fine-tuning stage.

Methodology - The motive here is to maintain the adequacy of the generated minutes while covering every necessary detail across the transcripts. To achieve this, we split a transcript into several blocks of short conversation threads. Simultaneously, we filter the final chain of conversation threads by excluding the unnecessary words, filler dialogues and utterances with negligible informational context. The model uses these generated blocks as the input to return the minutes corresponding to each block (see Figure 3); we further concatenate these outputs to obtain the full-length summary of the entire interaction. These summaries present an adequate amount of information but lack adequacy. For instance, the model might capture a casual conversation from some part of the meeting that mentions a particular location, meeting, or some other miscellaneous term. To tackle this issue, we chose to use the TextRank[17] Algorithm. This method is based on PageRank, which ranks the sentences according to their significance. Sentences from the summary with a good amount of contextual information get high scores. On the other hand, the sentences with less information or semantic duplicates of some other sentence in the transcript are ranked lower accordingly. On average, the model captures 15% of trivial and irrelevant information from the full-length summary. The obtained "gold-span" of the summary is then sorted and formatted to our convenience. Furthermore, we add the appropriate pronouns and eliminate grammatical mistakes. After the successful execution of the above steps, we obtain a final compressed minute (See Figure 2).

Evaluation - For experimenting and evaluating across a variety of approaches and models, BART(facebook/bart-large-cnn), Distil-BART(pre-trained on CNN corpus and fine-tuned on SAMSum), and T5(large, fine-tuned on SAMSum corpus) are considered as top contenders. A few minor details and anomalies were also observed during experimentation. While the performances of the base BART model, T5, and Pegasus were comparable, the fine-tuned versions of BART performed much better than those of the latter two. The extracted coverage and topic threads varied for each model. Although the readability of fine-tuned models was excellent, the fine-tuned Pegasus model lacked this quality to some extent. For evaluation purposes, we used the SummEval Evaluation Toolkit[18], which provides a variety of evaluation metrics, which assessed the minutes on different aspects of their quality.

Model/ Score	BART*	DistilBART	T5	**Proposed Model**
ROUGE-1	0.297	0.375	0.406	**0.45**
ROUGE-WE	0.162	0.205	0.229	**0.298**
BLEU	2.907	6.535	6.278	**7.068**
BERT-F1	0.563	0.620	0.615	**0.673**
TF-idf	0.19	0.25	0.31	**0.38**

Table 3: *shows the performance of different baseline models, employed during experimentation.*

5.2. Subtask-B

After the extraction, we apply a total of 6 similarity scores; which are cosine similarity, Rouge-1, Rouge-L, Sequence Matching, and some defined methods for cross-verification of important mentions (such as ORGANIZATION, LOCATION, PROJECT, etc) in both - the minute and transcript, as well as checking the presence of some rarely used set of words, in both of them. A combination of these scores is used in classification to determine the best possible outcome.

For experimentation and evaluation purposes, we use the available classifier models provided in the Scikit-learn library. During the final experimentation, the Support Vector Machine (SVM) and the Random Forest Classifier were found to outperform the rest of the models on our data.

Classifier	Accuracy	Precision	Recall	F1
Random Forest	**0.91**	**0.71**	**0.62**	**0.66**
SVM	0.88	0.65	0.40	0.49

Table 4: *describes the results achieved on Task B*

5.3. Subtask-C

To perform the classification, 6 similarity scores were evaluated, and a combination of which was used in classification to determine the best possible result. These scores included 'Cosine-Similarity' (measured similarity between two feature vectors, by capturing the orientation of the document and not the magnitude, unlike the Euclidean distance), 'Rouge-1' (took into account the number of matching uni-grams), and 'Rouge-L' (quantification of similairty based on the longest matching subsequence). 'Jaccard-Similarity' (measures the ratio of shared and distinct words between sentences). 'SequenceMatcher' (finds the longest contiguous matching subsequence that contains no junk element). Whereas, the 'RES-score' is computed with the help of the ratio of most common words to the total number of unique words.

In our run, again, the SVM and the Random Forest were the two classifiers that outstood the list.

Classifier	Accuracy	Precision	Recall	F1
Random Forest	**0.85**	**0.42**	**0.61**	**0.5**
SVM	0.77	0.26	0.53	0.35

Table 5: *describes the results achieved on Task C*

6. Conclusion and Future Work

In this work, we have described our system for AutoMin @ Interspeech 2021 for automatic minuting and analysis comparison of meeting minutes. The proposed system leverages the knowledge captured by large-scale transformer-based language and summarization models. We have also discussed various approaches that one can use to tackle the challenge. Our official submission obtained an accuracy of 95% in subtask B and 91% in subtask C. Our proposed system also takes care of the coverage, adequacy, and readability of meeting minutes that are to be generated. In the future, we would like to implement a Topical Segmentation strategy so that the generated minutes become more sound and convenient for a reader. We also plan to train a joint model by combining systems incorporating all the subtasks, and even newer challenges, which would surely prove to be time-saving, and would help in managing the documented transcript-minute pairs. This, combined with the main model can potentially prove to be a smooth and efficient utility that would inculcate time-saving and simplicity in the day-to-day schedules of different working groups. Due to the limitation of language models in capturing external knowledge and their training being restricted by the dataset (especially, the scarcity of meeting transcripts data), automatic minuting has posed a strong challenge to the researchers in the field of NLP. However, the knowledge grasped by various extensively pre-trained language models can be effectively leveraged to generate summaries and structure meeting minutes from them.

7. Acknowledgement

Tirthankar Ghosal has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 825460 for the project European Live Translator (ELITR)[4].

8. References

[1] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization," 2020.

[2] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 74–81. [Online]. Available: https://aclanthology.org/W04-1013

[3] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," 2020.

[4] C. Zhu, R. Xu, M. Zeng, and X. Huang, "A hierarchical network for abstractive meeting summarization with cross-domain pretraining," *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, November 2020. [Online]. Available: https://www.microsoft.com/en-us/research/publication/end-to-end-abstractive-summarization-for-meetings/

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[6] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," 2019.

[7] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," 2020.

[8] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Huggingface's transformers: State-of-the-art natural language processing," 2020.

[4] https://elitr.eu

[9] B. Gliwa, I. Mochol, M. Biesek, and A. Wawer, "Samsum corpus: A human-annotated dialogue dataset for abstractive summarization," *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, 2019. [Online]. Available: http://dx.doi.org/10.18653/v1/D19-5409

[10] Y. Chen, Y. Liu, L. Chen, and Y. Zhang, "Dialogsum: A real-life scenario dialogue summarization dataset," 2021.

[11] C. Zhu, Y. Liu, J. Mei, and M. Zeng, "Mediasum: A large-scale media interview dataset for dialogue summarization," 2021.

[12] S. Narayan, S. B. Cohen, and M. Lapata, "Don't give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, 2018.

[13] A. Clifton, A. Pappu, S. Reddy, Y. Yu, J. Karlgren, B. Carterette, and R. Jones, "The spotify podcast dataset," 2020.

[14] E. Loper and S. Bird, "Nltk: the natural language toolkit," *CoRR*, vol. cs.CL/0205028, 07 2002.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, A. Müller, J. Nothman, G. Louppe, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and Édouard Duchesnay, "Scikit-learn: Machine learning in python," 2018.

[16] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," 2014.

[17] R. Mihalcea and P. Tarau, "TextRank: Bringing order into text," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 404–411. [Online]. Available: https://aclanthology.org/W04-3252

[18] A. R. Fabbri, W. Kryściński, B. McCann, C. Xiong, R. Socher, and D. Radev, "Summeval: Re-evaluating summarization evaluation," 2021.

9. Generated Samples

Given below is an example of minutes generated by our model sampled from data-set provided in Task A:

DATE : 2021-07-21
ATTENDEES : PERSON4, PERSON5, PERSON8, PERSON10, PERSON13

SUMMARY-

- The deadline for the project is next Monday, June 15th.
  Someone from the project needs to be registered there.
  PERSON8 will try to register today.
- PERSON13 is going with PERSON4 to LOCATION5.
  They have a meeting before lunch on Monday.
  They have one more paper, she wants to submit it to Archive and PROJECT8 so that someone can read it.
- PERSON10 is on holiday for next two days.
  They have written one and half paragraph of the book yesterday, and will work on the book from now on.
- PERSON4 will write half of the chapters.
- PERSON8 will organize the chapters.
  They added some information from papers.
  They will write a preface to the book.
  He needs to generate, to get the similar metrics from the PROJECT3 and the rest.
- PERSON5 is going to write his survey.
  They will work with PERSON8.
- ALL are working on the papers.
  The deadline for feedback is at the end of June.
  The reviewers for PROJECT5 need to be at least a professor, but don't have to be from the university.
  The grant will be 5000 for it.
  The deadline for PROJECT7 should be in November.
  The conference will be virtualised and take place in 2021.
- PERSON8, PERSON13, PERSON5 and PERSON10 discussed the details of the conference.
  The abstract submission is on Monday, June 15th.
  PERSON5 and PERSON8 are going to write a survey for the project. They want to introduce new people to it.
- ALL discussed about the amount of money they are getting from the university.
  The money for this year cannot be used for bonuses.
  PERSON7 bought the computer that he is now using for some grant.
- PERSON8 got a mail from PR person saying that they can come to the official event.

Minuted by: Team ABC

And, on the following page there is a true positive instance predicted by our model, for TASK-C :

Minute:A)

PROJECT3 31. 08. 2020
Attendees: PERSON1, PERSON9, PERSON2
Purpose of meeting: Preparing for the demo, choosing the right people and language combination
Summary

- PERSON9 sent email to PERSON11
- PERSON1 checked PROJECT5 emails
- Discussed about the attendees during the demo
- Discussed input language
- Discussed language translation combination
- PERSON9 offered help with finding Romanian speaker
- Discussed person involved in the testing
- Discussed about date of the demo
- Discussed about a ORGANIZATION8 ASR
- Discussed about risk of Italian source
- Discussed a Session closing day date

Milestones

- PERSON8 will be person from ORGANIZATION2
- PERSON8 will be person from ORGANIZATION5
- German will be OK as input language
- PERSON1 does not have access to Romanian speaker
- PERSON1 will fill the Doodle

Minute:B)

Organizational stuff

- Monthly call will be on Thursday, 5 PM LOCATION1 time
  - At least PERSON14 and PERSON10 should take part
  - PERSON14 will care about including PERSON6 into the mailing list
- PERSON6's coming to LOCATION1
  - It is very desirable that PERSON6 comes to LOCATION1 in person
  - Visa issues due to Covid situations

PROJECT2

- PERSON10 is trying to contact ORGANIZATION5 colleagues, the communication is not completely perfect
- PERSON4 is preparing the leaflets, LOCATION1 is waiting

Progress on PROJECT6

- PERSON10 is trying the back-translation
  - It's low priority, is running on server, but may be stopped if needed.
  - No interesting results to discuss yet. Should be discussed with PERSON15 first, what to do next
  - PERSON10 may try the translations on CPUs

PROJECT4

- No special updates for now
- a related paper on BLEU that might be useful for evaluation
- Discussing metrics, using semantic metrics, different kinds of metrics
- Why do we need special metrics for MT

B Sample Outputs from our Minuting Model

AutoMin:dev_009

-PERSON2 is trying to record the call to see if they are able to summarize it automatically.
-PERSON13 will give a presentation on PROJECT1 on the Monday seminar on the 17th of February.
-The students' firm fair is taking place two weeks from now.
-It is an important event for data collection because students are presenting their companies, we record them, they transcribe it, they compete in how well their work their voice was recognized.
-There will be nonnative speakers PERSON2 and PERSON1 will train empty systems on corpora which are refind to contain higher frequency words.
-PERSON2, PERSON15, PERSON8 and PERSON8 worked on the paper for exceptement.
-PERSON2 has only one recordings translated into English.
-They have a limited file in LOCATION4, OTHER1.
-There is only one other recording in English.
-PERSON1 has found the appropriate command flacs so that the audio is compress to mp3 then shipped as mp3 to The recordings were before in mp3 format, then they were actually converted into flac format, and now they are in WAV format.
-Some words get cut in the middle.
-There are some problems with speaker diarization on fly.
-PERSON14 has already worked on the project last June.
-PERSON10 is waiting for a virtual machine for the translation server.
-They and PERSON2 will meet on Friday instead of Thursday next week to do a doodle.

AutoMin:dev_003

-PERSON3 and PERSON5 worked on the ORGANIZATION1 data set.
- The deadline for the data is November 30th, so they need to prepare it by the end of November.
- Today they will discussed the annotations, the alignment tool by PERSON4 and the PERSON4 is a tool for summarization.
-PERSON5, PERSON3, PERSON6 and PERSON1 want to make it an integral part of the shared task.
-PERSON3 proposes to make the alignment tool gated accepted by the community.
- They and PERSON5 will prepare a shared task proposal before the end of November.
- They need to prepare the data in the form like the PROJECT1 meetings.
-PERSON5 needs someone to take charge of the project until November 30th.
- They are waiting for someone who can help him with the task.
- The task is written in Python and there are technical issues with the tool.
- The tool is doing alignment, but it is not clear how to use it.
-PERSON3, PERSON5 and PERSON2 agree that they should start the alignment thing with their annotators soon, but they disagree on whether it should be a shared task.
-PERSON5 wants to start the annotation work on the ORGANIZATION1 corpus creation as soon as possible, because it's going to take a lot of time.
- They worked on an annotators tool created by himself and his girlfriend.
- They need to connect to the annotators by the end of this week.
- They also need to put all their data to GitHub, but they haven't finished it yet.
- They don't have a concrete PERSON5 should have it by the end of this week.