SoC Summer 2021 Final Documentation

Unscripted - MinutesWriter

Anirudh Mittal, Siddhesh Powar

Keywords: *Machine Learning, Natural Language Processing, Speech-To-Text, Abstractive Summarization, Minutes of a Meeting, Audio Input, Text Output*

Team Member Name	Roll Number	Email-Id
Neilabh Banzal	170010014	neilabhbanzal21@gmail.com
Ashish Gupta	203310006	ashishguptap99@gmail.com

Brief Description

The objective of the project was to explore Speech-To-Text Libraries and then implement them on an ideated project. We explored the *SpeechRecognition* and *speech_recognition* libraries, which gave good results on the test inputs.

The next part was choosing a project to implement this on. Of the shortlisted projects, we were assigned to write Minutes of the Meeting from the audio file. Apart from the Speech-To-Text, this involves abstractive summarisation, where we have used the transformers library.

However, when we used the audio from real-world audio meets, the results were not at all good. So, we shifted to using Meet Transcribe Chrome Extension for saving the STT captions generated during the Google Meet as our input, to give structured minutes of the meeting.

Progress

We started with learning - revision of standard libraries, Git, followed by traditional ML. (April)

Post endsems, we researched and learnt about text summarisation. In particular, we looked at example uses of HuggingFace and TorchAudio. We also looked at BERT for extractive summarisation. (May)

In June, we shifted towards Speech-To-Text. Once again, we started with some reading, and then followed it up with implementation. We implemented Speech-To-Text using the Google Web Speech API using the SpeechRecognition Library here. The results looked visually okay, but we needed a way to compare the performance of different libraries. Thus, we went into reading precision, recall, f1-score, N-grams, WER - performance metrics for generated text against the reference text. Then, we also looked at BLEU and ROUGE metrics. After looking at the various parameters input to ROUGE and BLEU, and finalising the parameters after discussion with everyone, we implemented the metrics here.

Next, we looked into projects where we could implement our learnings. Out of the 3 projects - Minutes of the Meeting, Audiobook / Movie audio summarisation, and Journalism Transcriber, we were assigned the Minutes of the Meeting project. We worked on the flow of the project and divided it into 2 parts - STT and Transcript Summariser.

In July, we looked further into text summarisation. There are basically 2 types of summarisation - extractive, where the most important sentences are extracted from the input, and abstractive, where, using the input text, completely new sentences are generated. We quickly realised that for our project, we need abstractive summarisation.

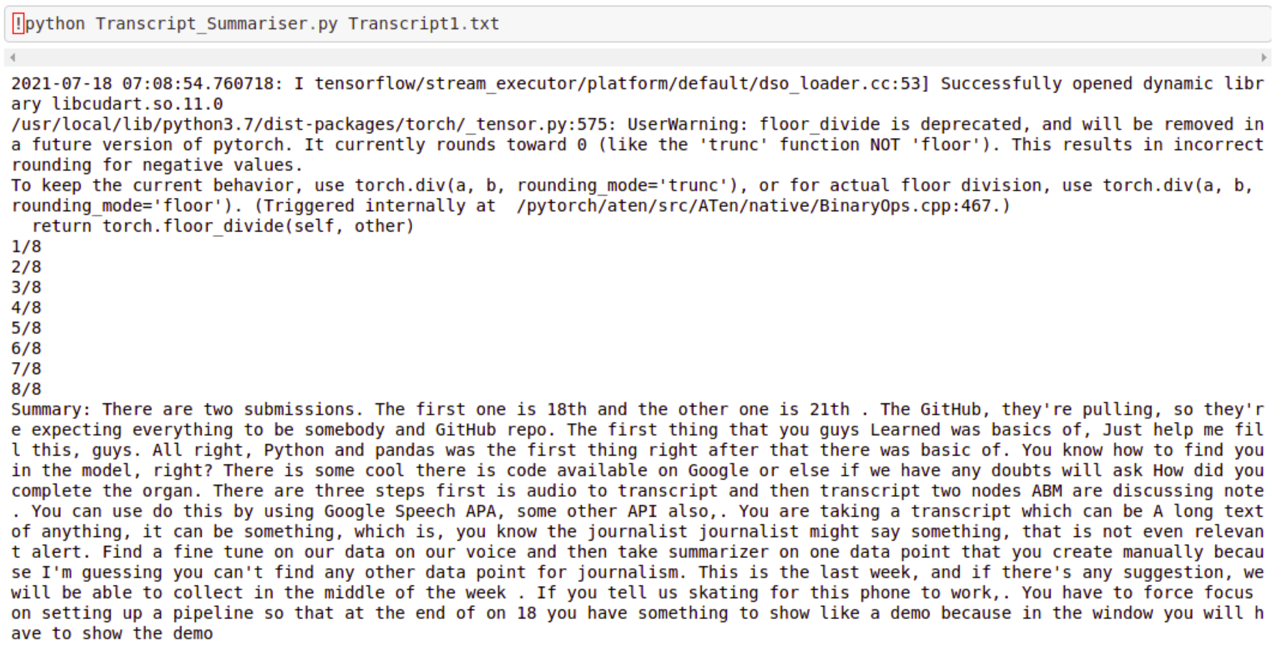
For generation of test input for the text summarisation, we used the Meet Transcribe Chrome Extension. Then, we used the *transformers* library's *summarizer* function to summarise the text under the default model, and then followed by a *t5-base* model. The base model was faster, but the text generated was slightly better visually under the *t5-base* model.

For the final code, we needed *sys.argv* to take in arguments and *regex* to remove unnecessary fillers in the text.

The final part - generating names of the members of the meet, generating the title of the meeting is something that is going to be done in the next 2-3 days to get the complete minutes structure.

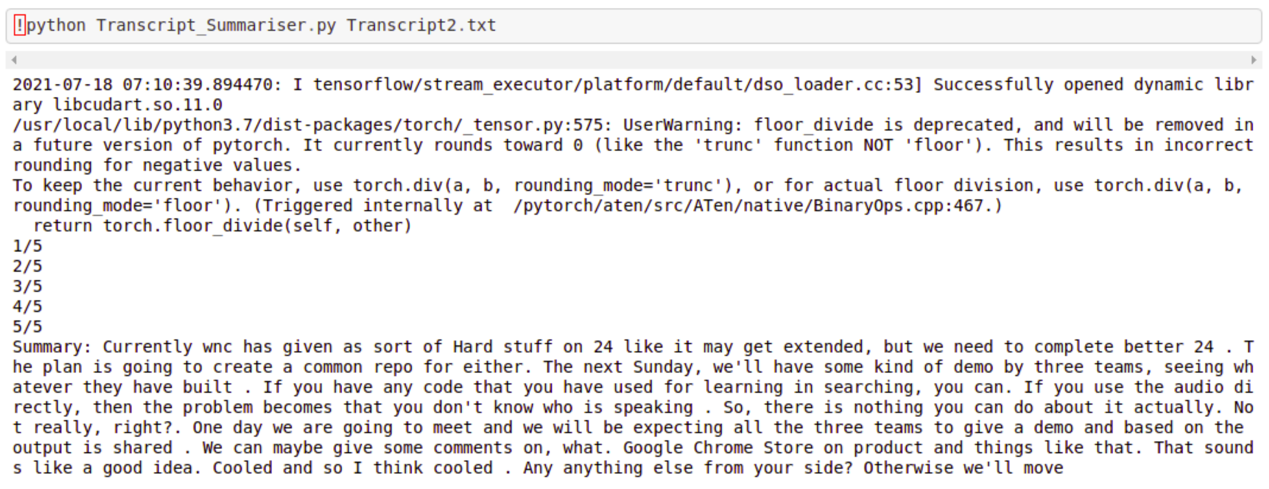
Results

The GitHub Repo is linked here, and this folder has some runs on actual meetings.
The Documentation and Reports for SoC are available here.
The weekly progress reports are linked here- Ashish, Neilabh
The testing Google Colab notebooks used for playing around and testing are saved here.



```
!python Transcript_Summariser.py Transcript1.txt
```

```
2021-07-18 07:08:54.760718: I tensorflow/stream_executor/platform/default/dso_loader.cc:53] Successfully opened dynamic libr
ary libcudart.so.11.0
/usr/local/lib/python3.7/dist-packages/torch/_tensor.py:575: UserWarning: floor_divide is deprecated, and will be removed in
a future version of pytorch. It currently rounds toward 0 (like the 'trunc' function NOT 'floor'). This results in incorrect
rounding for negative values.
To keep the current behavior, use torch.div(a, b, rounding_mode='trunc'), or for actual floor division, use torch.div(a, b,
rounding_mode='floor'). (Triggered internally at  /pytorch/aten/src/ATen/native/BinaryOps.cpp:467.)
  return torch.floor_divide(self, other)
1/8
2/8
3/8
4/8
5/8
6/8
7/8
8/8
Summary: There are two submissions. The first one is 18th and the other one is 21th . The GitHub, they're pulling, so they'r
e expecting everything to be somebody and GitHub repo. The first thing that you guys Learned was basics of, Just help me fil
l this, guys. All right, Python and pandas was the first thing right after that there was basic of. You know how to find you
in the model, right? There is some cool there is code available on Google or else if we have any doubts will ask How did you
complete the organ. There are three steps first is audio to transcript and then transcript two nodes ABM are discussing note
. You can use do this by using Google Speech APA, some other API also,. You are taking a transcript which can be A long text
of anything, it can be something, which is, you know the journalist journalist might say something, that is not even relevan
t alert. Find a fine tune on our data on our voice and then take summarizer on one data point that you create manually becau
se I'm guessing you can't find any other data point for journalism. This is the last week, and if there's any suggestion, we
will be able to collect in the middle of the week . If you tell us skating for this phone to work,. You have to force focus
on setting up a pipeline so that at the end of on 18 you have something to show like a demo because in the window you will h
ave to show the demo
```



```
!python Transcript_Summariser.py Transcript2.txt
```

```
2021-07-18 07:10:39.894470: I tensorflow/stream_executor/platform/default/dso_loader.cc:53] Successfully opened dynamic libr
ary libcudart.so.11.0
/usr/local/lib/python3.7/dist-packages/torch/_tensor.py:575: UserWarning: floor_divide is deprecated, and will be removed in
a future version of pytorch. It currently rounds toward 0 (like the 'trunc' function NOT 'floor'). This results in incorrect
rounding for negative values.
To keep the current behavior, use torch.div(a, b, rounding_mode='trunc'), or for actual floor division, use torch.div(a, b,
rounding_mode='floor'). (Triggered internally at  /pytorch/aten/src/ATen/native/BinaryOps.cpp:467.)
  return torch.floor_divide(self, other)
1/5
2/5
3/5
4/5
5/5
Summary: Currently wnc has given as sort of Hard stuff on 24 like it may get extended, but we need to complete better 24 . T
he plan is going to create a common repo for either. The next Sunday, we'll have some kind of demo by three teams, seeing wh
atever they have built . If you have any code that you have used for learning in searching, you can. If you use the audio di
rectly, then the problem becomes that you don't know who is speaking . So, there is nothing you can do about it actually. No
t really, right?. One day we are going to meet and we will be expecting all the three teams to give a demo and based on the
output is shared . We can maybe give some comments on, what. Google Chrome Store on product and things like that. That sound
s like a good idea. Cooled and so I think cooled . Any anything else from your side? Otherwise we'll move
```

Learning Value

Learned basics and implementation of NLP as a part of this exploratory project. We tried out multiple libraries and finally implemented them into a pipeline for solving a real-world problem - automatic generation of minutes of the meeting.

Software used

Python
GitHub, Git
Google Colab
Meet Transcribe Chrome Extension

Python Libraries -

- SpeechRecognition
- speech_recognition
- transformers
- re (regex)
- math, sys, os

Suggestions for others

- Implementation and Learning in parallel
- Try out different libraries and look up how they work

References and Citations

[1] P. Khandare, S. Gaikwad, A. Kukade, R. Panicker, and S. Thamke. Audio data summarization system using Natural Language Processing. *International Research Journal of Engineering and Technology (IRJET),* Vol- 06, 2019.

[2] M. Allahyari *et al.* Text Summarization Techniques: A Brief Survey, 2017.

[3] https://www.machinelearningplus.com/nlp/text-summarization-approaches-nlp-example/

[4] https://github.com/theamrzaki/text_summurization_abstractive_methods

Disclaimer

Licenses

Python - FOSS
GitHub, Git - Free, FOSS
Google Colab - Free
Meet Transcribe Chrome Extension - Free

Python Libraries -

- SpeechRecognition - Free
- Speech_recognition - Free
- Transformers - Free
- re (regex) - Free
- math, sys, os - Free