CPEN 291 - Final Report

A Brief Description

This project is a way to summarize your lectures into short notes to go over without needing to re-watch the video. If you upload a lecture's video file to the application, it will output summarized notes of the whole lecture in a text file.

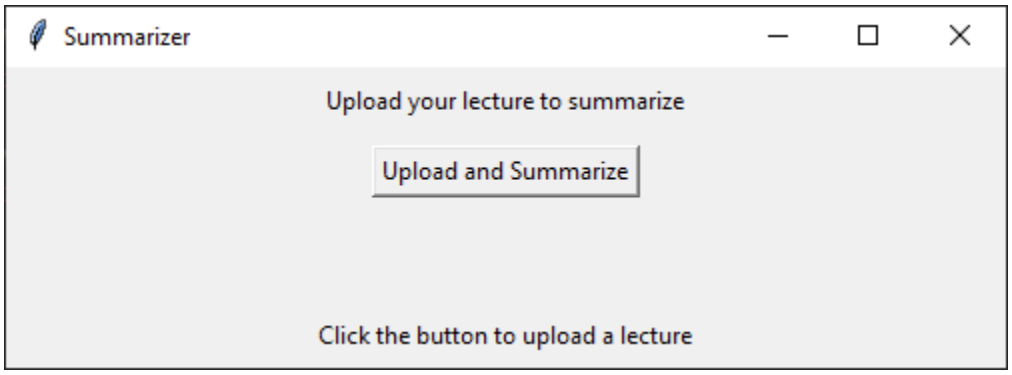
The video lecture is converted to an audio file with the ffmpeg tool using the ffmpeg-python package. The temporary audio file is then transcribed using Google's speech-to-text API. Then the transcript is summarized using a sequence-to-sequence machine learning model, BART. This model would have been trained on lectures related to the field of computer science by maximizing the ROUGE accuracy scores of the summaries generated in the training dataset using fastai2.

Instructions

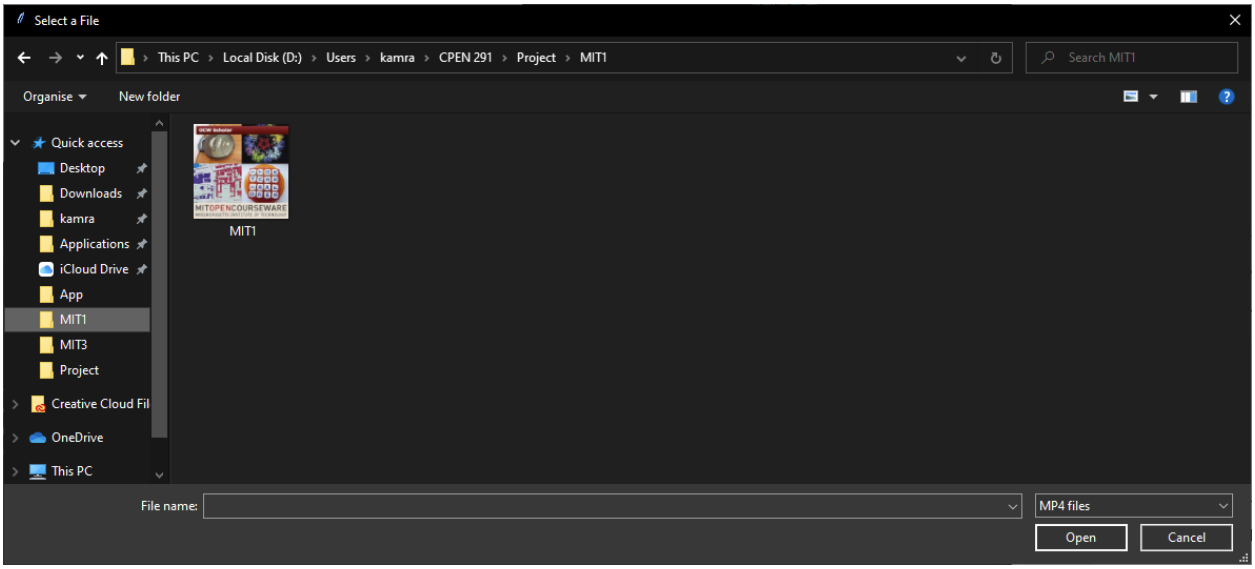
1. Run the application in the App folder where the google cloud key is stored

   https://github.com/UBC-CPEN291/project-team-ankylosaurus/tree/main/App

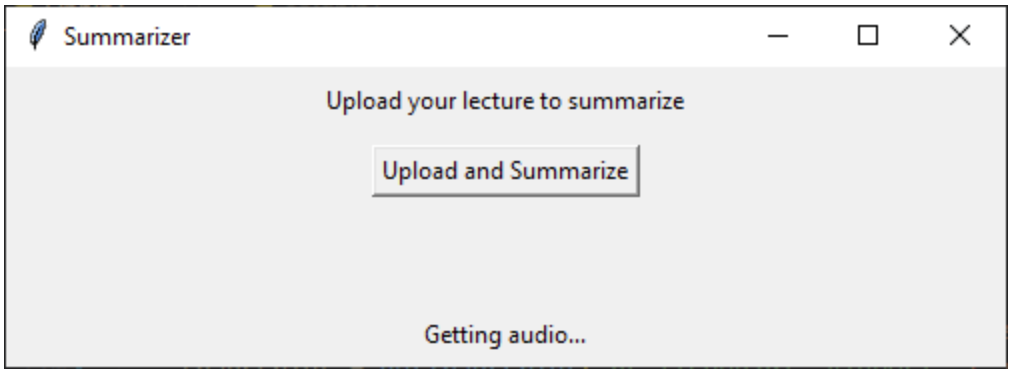
2. The application opens a window where users can upload their video lectures.

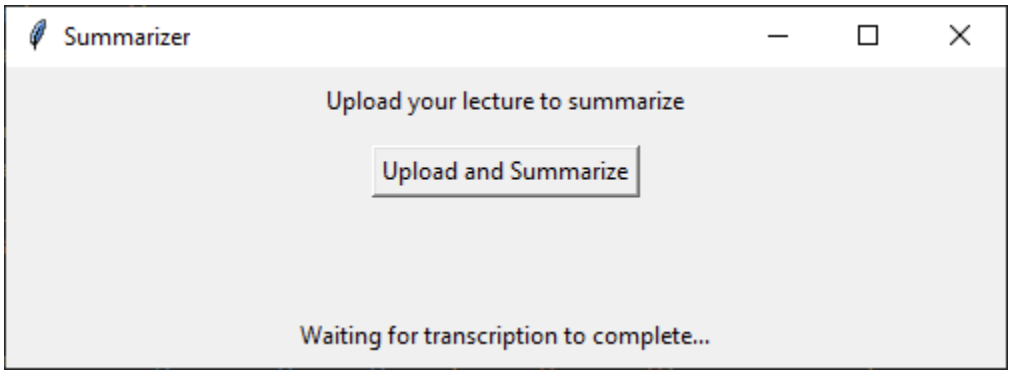


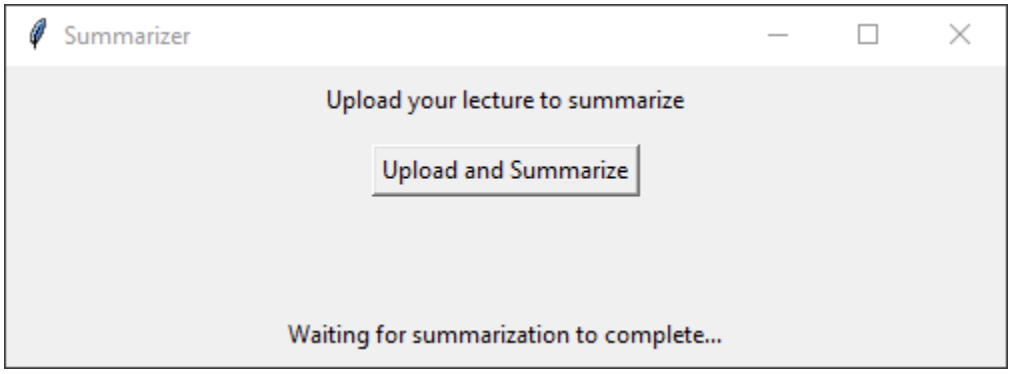
3. Here, users can select an mp4 file to summarize (A computer science video lecture).

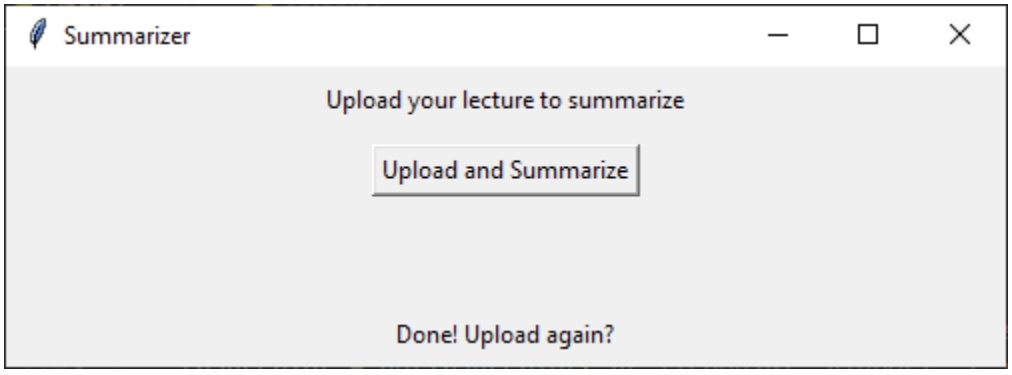


4. Wait for the code to generate a summary.

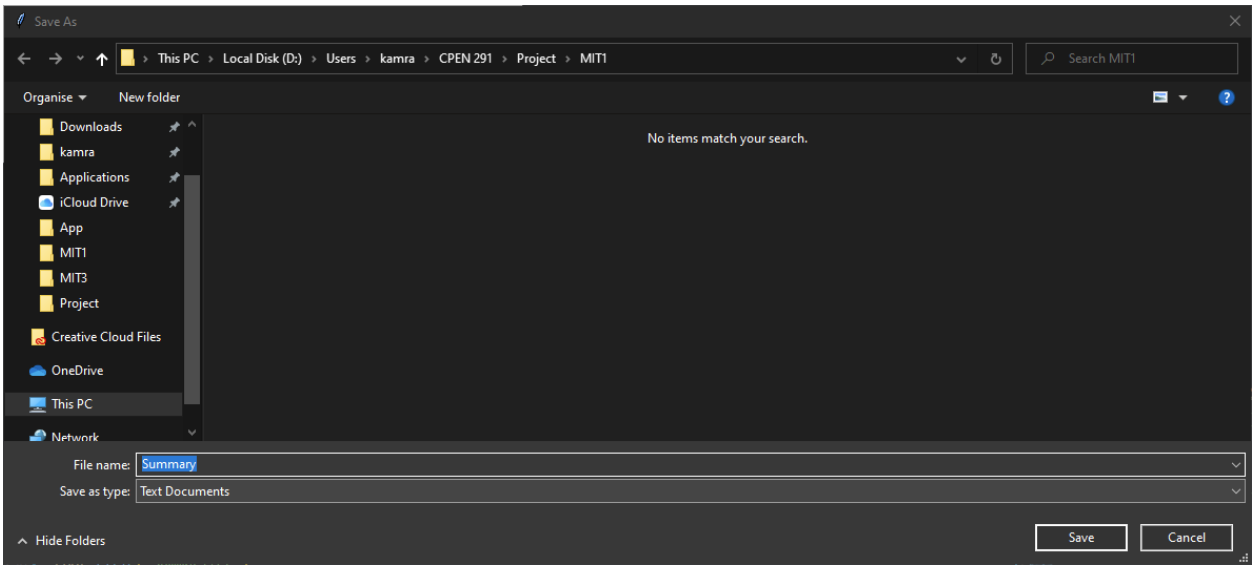




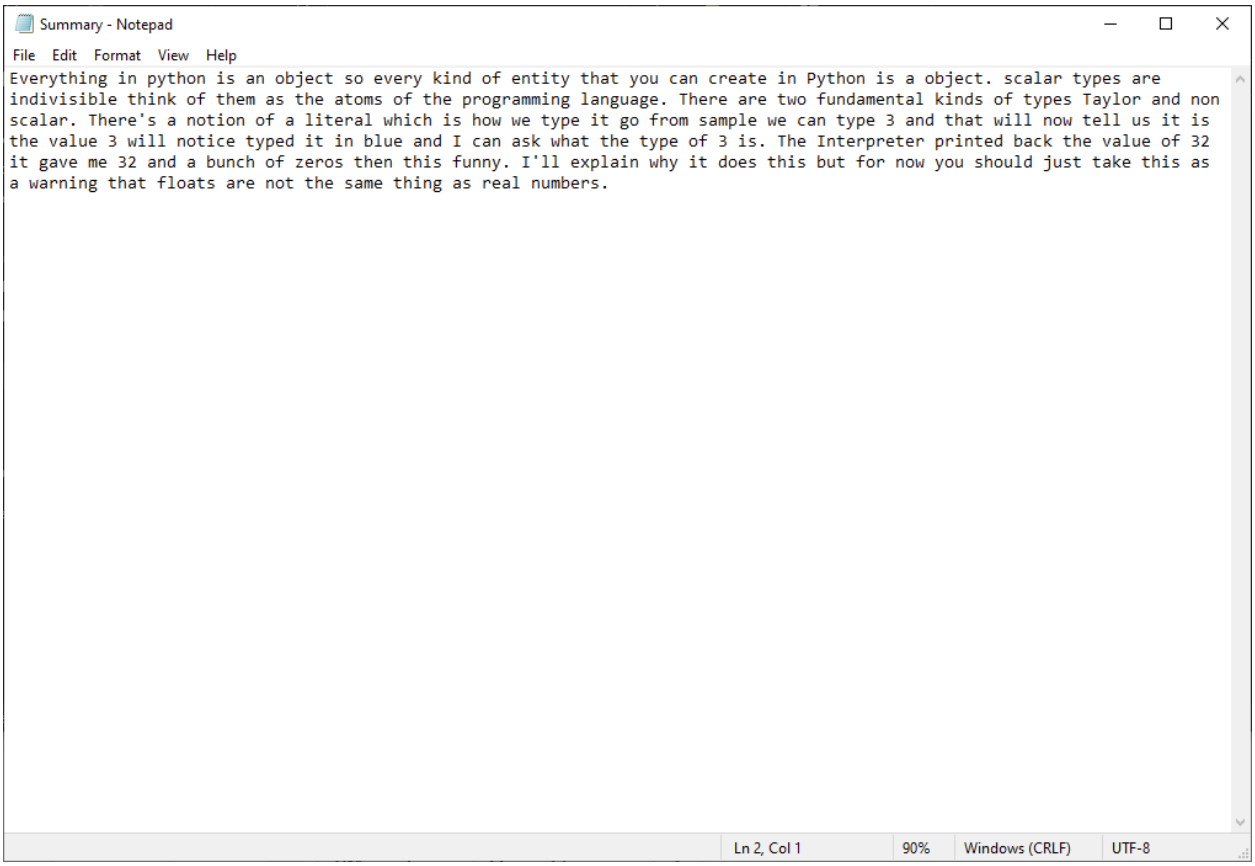




5. Then, the generated summary can be saved as a text document where needed.



6. The summary is stored in a text file as shown.



Components used

- Tkinter: A python user interface toolkit with which we made the application
    - It is used to make the GUI of the summarizer application.
    - It allows the user to upload a video file, and to save the generated summary.
    - This was used as it is a rather simple way of creating a GUI which can use a trained machine learning model to generate an output.
    - Using threads, the application is able to run while code is running in the background.
    - Using the Nuitka package, the python file was converted into an executable
    - The trade-off was that while a web server would be faster when running code, a tkinter app would be easier to make and smaller in overall size.
- FFmpeg: A video to audio converter which can be integrated into python
    - It is used to convert the video lecture uploaded by the user into a temporary audio file which can then be transcribed.
    - This tool was used
- Google Cloud speech-to-text API: An API that will transcribe audio
    - This will convert an audio file given a GCS URI.
    - It is used to transcribe the temporary audio file into a transcript for summarization
- BART: A sequence-to-sequence machine learning model which summarizes texts.
    - Pretrained on the CNN/Daily Mail dataset created by the Facebook AI Research team.

- It is used to generate a summary of the video lecture by converting the transcript into a list of tokens, where these tokens are fed into the model. A new list of tokens is returned, which is then translated back into a readable summary.
- Tried to use fastai2 to train and fine tune the model on the transcripts of computer science lectures collected and summaries written by the team. More information can be found in the "Training Code" section in "Components not used."

- ROUGE: Recall-Oriented Understudy for Gisting Evaluation is a set of metrics by which we evaluate generated summaries. It takes into account precision, recall, n-grams, and longest matching sequences with respect to a man-made summary.
  - It is used to better train and fine tune the BART model to work with video lectures on the topic of computer science.
  - Used the ROUGE-2 score to evaluate our text. ROUGE-2 generates a score based on the percentage of overlap of bigrams between the generated and reference summaries. Bigrams are just two consecutive words that occur in the text.

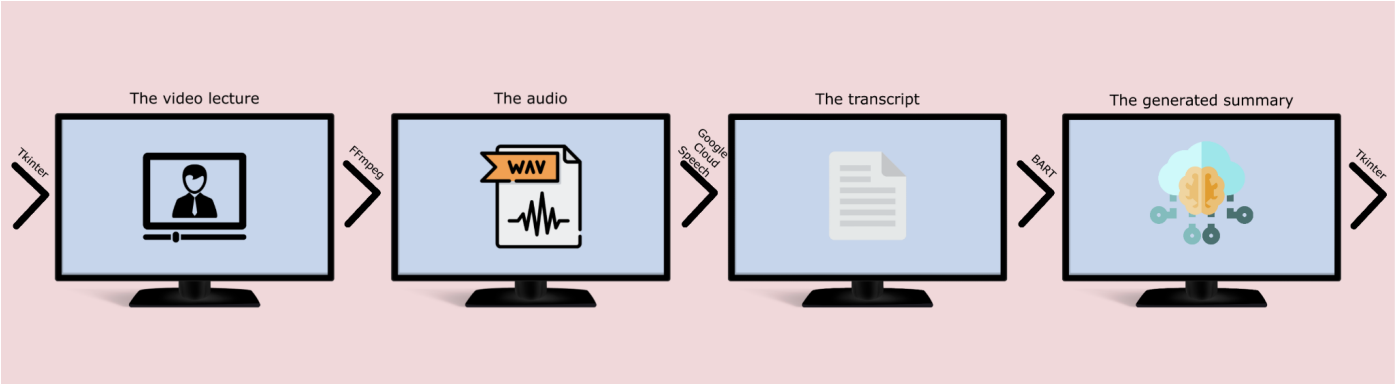


Diagram of components and how they flow together

Components not used:

- SpeechRecognition
    - Not used due to its inability to transcribe some audio files for possibly the length of the file; however, similar files of similar length were transcribed. Thus, its inconsistency led to it not being used in the final product.
- Training Code: tried multiple different training codes to fine tune our BART model, with none of them working.
    - https://github.com/huggingface/transformers/tree/master/examples/pytorch/summarization: threw unknown errors when executed.
    - https://github.com/ohmeow/ohmeow_website/blob/master/_notebooks/2020-05-23-text-generation-with-blurr.ipynb: training could not be ran because the Google Collab notebook did not have enough memory for the CUDA GPU.
    - https://discuss.huggingface.co/t/train-bart-for-conditional-generation-e-g-summarization/1904: dataset was not compatible with the training code.
    - https://github.com/CurationCorp/curation-corpus/blob/master/examples/bart/finetuning-bart.ipynb: dataset was not compatible with the training code.
    - https://github.com/huggingface/notebooks/blob/master/examples/summarization.ipynb: dataset was not compatible with the training code.

Overall contributions

- Kamran Alam
    - Made the Tkinter application and compiled it into an exe file using Nuitka.
    - Found lectures from MIT OpenCourseWare to have a workable dataset for model training.

    - Integrated ffmpeg to generate temporary audio files from video lectures.
    - Set up the BART pretrained model and wrote code to calculate ROUGE score based on generated summaries with relation to man-made summaries.
- Shashi Kiran Gonugunta
    - Found lectures to have a workable dataset for model training
    - Checked outputs of transcript generation code
    - Checked for errors in code and organized generated transcripts to work with the BART model.
- Sandon Li
    - Tried to train and fine tune the BART machine learning model to work with lectures related to the field of computer science
    - Found lectures from UBC CPSC 221 for a workable dataset for model training
    - Checked outputs of transcript generation code.
    - Helped generating transcripts and setting up the BART pretrained model
- Ken Liang
    - Found lectures from Harvard CS50x to have a workable dataset for model training
    - Tested SpeechRecognition for M1-M3, converting video files, but decided against it afterwards.
    - Uploaded video to Google Cloud Bucket storage to transcribe.
    - Used Google Cloud speech-to-text API to transcribe audio files.
    - Generated video transcripts and debugged errors in the corresponding code

Resources

- Tkinter
    - https://www.geeksforgeeks.org/file-explorer-in-python-using-tkinter/
    - https://www.geeksforgeeks.org/python-askopenfile-function-in-tkinter/
    - https://www.geeksforgeeks.org/python-asksaveasfile-function-in-tkinter/
    - https://www.geeksforgeeks.org/how-to-use-thread-in-tkinter-python/
- Nuitka
    - https://nuitka.net/doc/user-manual.html
- FFmpeg
    - http://ffmpeg.org/documentation.html
    - https://github.com/kkroening/ffmpeg-python
- SpeechRecognition
    - https://pypi.org/project/SpeechRecognition/
    - https://www.thepythoncode.com/article/using-speech-recognition-to-convert-speech-to-text-python
- BART
    - https://github.com/CurationCorp/curation-corpus/blob/master/examples/bart/finetuning-bart.ipynb
    - https://medium.com/areomoon/text-summarization-with-pretrained-bart-model-b961f3f0d8fb
    - https://github.com/huggingface/transformers/tree/master/examples/pytorch/summarization

    - https://discuss.huggingface.co/t/train-bart-for-conditional-generation-e-g-summarization/1904
    - https://github.com/huggingface/notebooks/blob/master/examples/summarization.ipynb
    - https://github.com/ohmeow/ohmeow_website/blob/master/_notebooks/2020-05-23-text-generation-with-blurr.ipynb
    - https://huggingface.co/transformers/model_doc/bart.html
- ROUGE
    - https://kavita-ganesan.com/what-is-rouge-and-how-it-works-for-evaluation-of-summaries/
    - https://github.com/tagucci/pythonrouge
- Google Cloud Storage
    - https://cloud.google.com/speech-to-text/docs/async-recognize#speech_transcribe_async_gcs-python
    - https://cloud.google.com/storage/docs/uploading-objects#storage-upload-object-code-sample
    - https://stackoverflow.com/questions/49376846/error-in-google-cloud-speech-api-synchronous-speech-recognition-docs