AUTOMATIC CHAPTERIZATION OF VIDEOS

CS 410 Text Information Systems – UIUC – Final Project

DECEMBER 6, 2020

NITISH KUMAR JAIN / VIBHOR JAIN
Nitishj2@illionois.edu / vibhorj2@illinois.edu

Table of Contents

Introduction

Videos are proliferating in our everyday life, starting from pure entertainment like Netflix, Hulu to more involved videos like Khan Academy, Udemy, Pluralsight, videos are everywhere. However, despite the ubiquity of videos, navigating videos is still a difficult proposition. We have experience with documents and can easily search and navigate in documents by searching, using Table of contents, Bookmarks, Navigation hierarchies etc. However same is not yet available with videos. With the ever-increasing adoption of '*everything online life*' this has become a necessity. Nobody wants to browse through hours of videos to get to the relevant section of the video. Our goal is to Chapterize a video, i.e., automatically preparing a Table of Content for a video.

Approach

We plan to do this in the 3 major phases.

Speech to Text: We used Azure Cognitive service to transcribe videos – Speech to Text

Preprocess: Now using the available text we would preprocess the text removing Stop words etc. Use Glove word embedding vectors for semantic meaning of sentences. Use Azure Cognitive services API to get timestamps for each beginning sentence of a section.

The first step of this project was to set up the Video Upload Front end and do the transcription. For the Front-End environment, we used the python-based web framework Flask since the text processing and NLP was going to be done in Python. We then set up the upload page and applied the azure cognitive services API to do the speech to text conversion.

The next step was to do the section partitioning. This involved preprocessing the text and then using Glove word vectors, which were generated by training on large datasets of Wikipedia pages to represent the semantic meaning of a word in a 100-dimension vector. After partitioning the transcription into sections, we used the cognitive services API to get the timestamps for the beginning sentence of each section.

Finally, we got the title phrase for each section using a simple extraction-based method. Currently the section title is simply the most commonly occurring words in a section, removing **stop words,** since these should theoretically summarize and encompass a section. Then once we had the timestamps and titles of each section, it just had to be displayed on the front end.

Sentence Partitioning:

By using the Glove Word vectors that I described briefly in the previous slide, I averaged all the word vectors for each sentence, to create sentence vectors. Then I defined a similarity threshold of 0.78, where the scale is from 0 being the least like 1 being the most similar. This number was one that I experimented with until we found one that worked the best for most test cases.

Now the way it works is it pretty much starts with the first sentence and compares it with the next and if the similarity is at least 0.78, then it adds the second sentence to a list with the first sentence. Then it checks the third sentence to see if it is within the similarity threshold when comparing it with the average of the sentence vectors for the first section so far, which is consistent of sentence 1 and sentence 2. Let us say this time the third sentence's similarity to the current section is not high enough. Then it ends the first section at sentence 1 and 2 and then adds sentence 3 as the first sentence of the next section. This process repeats itself and that is the algorithm!

Display the Table of Contents: Finally get the title phrase for each section, using an extraction-based method. Display section timestamps and titles on a webpage.

Tools & Programming Language

We used the Python programming language, Flask for the front end.

Python Glove: Global Vectors for Word Representation. GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

Azure Cognitive service to convert videos to Audio (i.e., Speech to Text)

Results:

We could successfully build Table of Content (Chapters) for videos (and audios) using the combination of various NLP and Text Analytics techniques. The results are very promising and can be used in several scenarios. This can go a long way in democratizing information which is deep embedded in videos.

References:

https://azure.microsoft.com/en-us/services/cognitive-services

https://nlp.stanford.edu/projects/glove/