Brian Reinbold
brianjr3

Progress Report

Task	Time Range Estimate (Min-Max)	Estimated Time Remaining
Download transcripts	1-2 hours	0
Parse transcripts	6-8 hours	0
Construct time series based on cosine similarity	1-2 hours	0
Construct time series based on Jenson-Shannon divergence	1-2 hours	0
Create visuals of time series	1-2 hours	0
Implement Bayesian change point detection of Barry and Hartigan (1993)	1-2 hours	1-2
Implement the nonparametric, hierarchical divisive algorithm of James and Matteson (2015)	1-2 hours	1-2
Compare clusters from proposed segmentations with baselines	2-3 hours	2-3
Substitute TF-IDF weighting	1-2 hours	1-2
Substitute Okapi-B25	1-2 hours	1-2
Generalize code so can be ran on all lectures	2-3 hours	1
Create documentation	1-2 hours	1.5
Prepare presentation	1-2 hours	1-2
Fit LDA Model	-	1-2
Total	**20-34 hours**	**10.5-17.5 hours**

The Progress Made

I have downloaded all the transcripts for cs-410 using the coursera-dl repo.

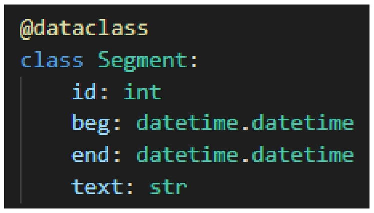
I have parsed the raw transcript files. Below is a sample of what an "srt" file looks like:

```
1
00:00:00,000 --> 00:00:05,293
[MUSIC]

2
00:00:10,067 --> 00:00:15,310
In this lecture, we continue
the discussion of vector space model.

3
00:00:15,310 --> 00:00:18,810
In particular, we're going to
talk about the TF transformation.
```

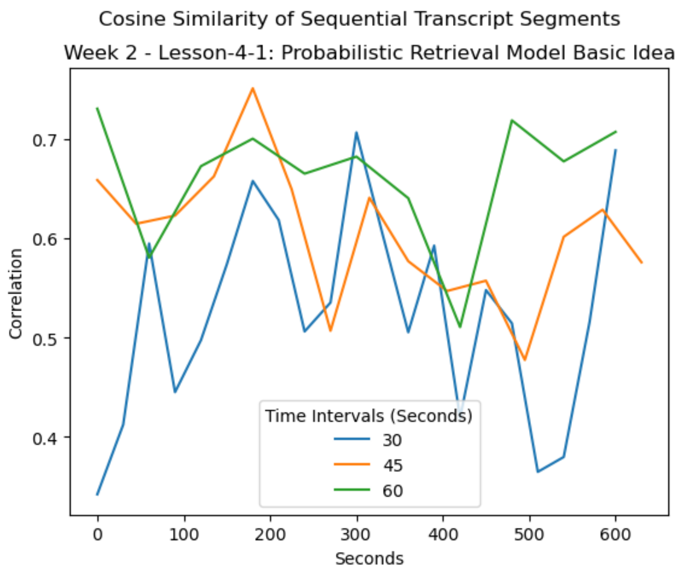
Each segment is separated by a new line. The first line of a segment is the segment ID, the second line gives the beginning time and end time of the segment, and the rest of the lines are the text in that segment. I defined a *dataclass* to hold the relevant components each segment:



```python
@dataclass
class Segment:
    id: int
    beg: datetime.datetime
    end: datetime.datetime
    text: str
```

I used the *datetime* module to format the times. Each transcript becomes a list of *Segments*. From here, I defined a time interval and merged the segments within the interval using *dataetime.timedelta*. I have experimented with 30, 45, and 60 second intervals. I used the *NLTK* module to tokenize the text and remove stop words. Finally, I calculated the term-document matrix and used it to calculate the term frequencies by documents. I used these term frequencies by document vectors to calculate similarity between documents.

I have also constructed time series of document similarities using cosine similarity and Jenson-Shannon divergence. I used the *spatial* module form *scipy* for them. I have made some quick line plots of these time series.



The above plot shows the cosine similarity of sequential transcript segments for the lesson *Probabilistic Retrieval Model Basic Idea.* The time series of 30 second segments clearly have higher variance than the time series of 60 second segments.

I have generalized the code to process all raw transcript files in CS410 and have calculated all time series.

Remaining Tasks

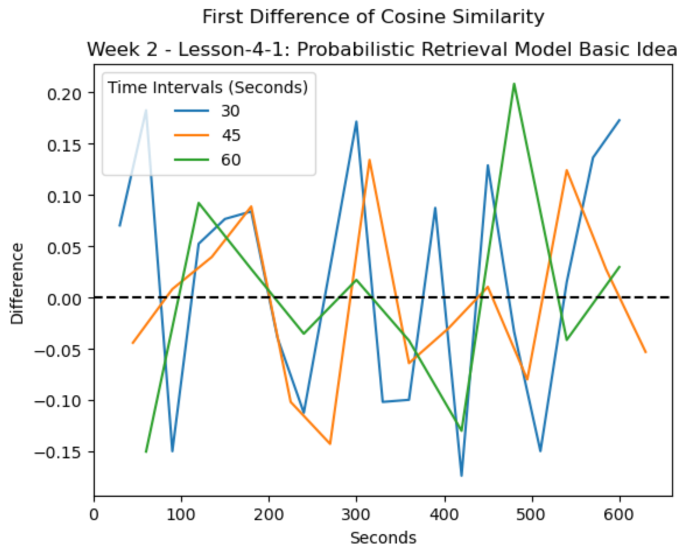
I still need to implement the change point detection algorithms. The ruptures library will be helpful.

I need to develop methods to evaluate the model derived topic transitions. I plan to compare the clusters from the proposed segmentations with the current segments.

If I have time, I can experiment using TF-IDF weighting and Okapi-B25.

I have begun generalizing the code and creating documentation, but there is still more to do for that task.

Looking at the above graph again, it seems that the time series evolve in phases of increasing and decreasing correlation. By taking the first difference, these phase changes are more apparent (see the graph below).



Besides estimating time series change points, I could fit a Markov switching model and use its states to determine topic transitions.

Once I know how many topic transitions there are in a lecture, I can use the number of topics to then fit a Latent Dirichlet Allocation model using gensim. Then I can compare which topics the LDA model picks for each transcript segment. If the topic changes between segments, then that suggests a topic transition. I can then compare the topic transitions suggested by the LDA model versus using change point detection algorithms. An additional benefit of the LDA model is that I can use it find which words are associated with each topic.

Any Challenges/Issues Being Faced

One challenge that I faced but have since resolved was that the cousera-dl script was not working. I received the following error: “HTTPError 404 Client Error: Not Found for url: https://api.coursera.org/api/onDemandCourseMaterials.v1/”. After doing some digging, it turns out that just a few days before I started working on the project, Coursera had updated their API endpoints, and the coursera-dl script did not reflect this update[1]. I updated the *api.py* file to point to the updated API endpoint, and then it worked. Thankfully, I figure it out or else I would have had to manually download all the files.

I have no other issues.

[1] See https://github.com/coursera-dl/coursera-dl/issues/834 for more info.