Exploring better ways to segment lecture videos based on topic transition

Documentation

Milad Parvaneh
Email address: `sparva2@illinois.edu`

December 16, 2022

Introduction

There is an existing in-progress project at the University of Illinois at Urbana-Champaign, Smartmoocs, in which the goal is to improve students' learning experience in MOOCs. Here, we aim to develop on top of the Smartmoocs project. The lecture videos in MOOCs are currently divided into 1-minute segments. We aim to explore alternative ways using which we can segment the videos according to the various topics covered by each lecture. To achieve this, we would need to identify the transition points of topic changes and then make the segmentation based on these points.

Datasets, algorithms and techniques

For this project, the lecture videos of the course CS410, Text Information Systems, are used. For CS410, lecture transcripts and time stamps are available on the Coursera platform. These transcripts are used to extract various topics within a given lecture as well as their corresponding topic change points. Then, the time stamps associated with these transition points can be used to segment the videos.

Probabilistic Latent Semantic Analysis (PLSA) [Mei et al., 2007; Lu et al., 2011] is among the probabilistic topic models Blei [2012] that can be used to mine multiple topics from text. It can be considered a mixture model with k unigram language models in which k represents the number of topics. Therefore, this method can be an ideal candidate for our video segmentation task. Also, the Expectation-Maximization (EM) algorithm is employed, which is a practical algorithm for computing the maximum likelihood estimate of mixture models. An alternative approach to extract topics can be using the Latent Dirichlet Allocation (LDA) which is an extension of PLSA. For this project, we chose PLSA as our model to mine lecture topics.

The source code is developed on top of the homework MP3 of the course CS410, in which the original PLSA algorithm is implemented using python.

Overview of the functions

The source code uses three basic python libraries: `Numpy`, `Pathlib`, and `Collections`. Listed below are the functions used and a brief overview of their functionality:

- `main`: Calls the main functions of the code. This includes preliminary functions for initialization and building corpus and vocabulary as well as the PLSA algorithms. Some primary parameters are also set here by the user, including the path for lecture transcripts, number of topics to be considered, and maximum iterations allowed.
- `normalize`: Borrowed from CS410-MP3, it normalizes a given two-dimensional matrix along the rows. This is needed in order to satisfy the probability constraints that are required for word distributions and document topic coverages.
- `__init__`: Initializes the parameters. It also reads the stop words. Stop words are filtered out from the transcripts. This, then, allows us to avoid using a background language model. Hence, a less expensive algorithm in terms of computation cost can be used.
- `build_corpus`: Reads the documents, and saves them as a list of list-of-words. An example would look like: [['today' ,'is', 'snowy', ...], [ ], ...].
- `build_vocabulary`: Forms a list of unique words from the corpus.
- `build_term_doc_matrix`: Forms the term-document matrix, in which each row represents a document and each column is for a vocabulary term.
- `initialize`: Initializes the matrices document_topic_prob and topic_word_prob. Two initialization options are available: uniform and random. However, random is used for all cases.
- `expectation_step`: This is the E-step from the EM algorithm, in which topic_prob is calculated.
- `maximization_step`: This is the M-step from the EM algorithm, in which topic_word_prob and document_topic_prob are updated.
- `calculate_likelihood`: Calculates the log-likelihood of the model.
- `plsa`: Implements the original PLSA algorithm. After reaching the maximum number of iterations, it takes the words with the highest probabilities of each topic_word_prob, and uses them as transition points to segment the transcripts.

Implementation

This code is built on top of the PLSA implementation of CS410-MP3. However, some remarks should be noted:

- The course CS410 is a 12-week course. Each week consists of several lecture videos where lecture transcripts are provided as text files. Each week's transcripts are considered a separate corpus and fed into the code separately. This approach is picked because the course materials for each week are mostly designed around unique topics (or at least this is assumed).
- Stopwords are used to filter out the common words from the transcripts. An alternative approach could be to employ a background language model.
- The number of topics to be extracted are chosen to be a fixed number, ten, for all set of documents. In other words, it is assumed that ten topics are presented during each week's lectures. Different numbers of topics can also be examined.
- One of the outputs of the PLSA algorithm is the word distributions for the predefined number of topics. The word with the highest probability from each topic_word distribution is used as transition points of subjects in the transcripts. One can think of better strategies for choosing these transition points.

Usage of the software

While running the code, it has been observed that each run results in a different distribution of topic_word probabilities. This is likely due to the fact that PLSA has many local optimum estimates. Employing the LDA may resolve the problem. Another possible reason is that our corpus is very small, with between five to ten documents for each week's session. As a result, it may be less of a properly-sized corpus for PLSA to extract useful topics from.

The log_likelihood has been observed to increase as the EM algorithm is running. However, the likelihood has been seen to follow a decreasing pattern for a large number of maximum iterations allowed in which the EM algorithm is allowed to run for longer. The decrease amount was relatively small, and the words with the highest probabilities were not affected.

References

D. M. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

Y. Lu, Q. Mei, and C. Zhai. Investigating task performance of probabilistic topic models: an empirical study of plsa and lda. *Information Retrieval*, 14(2):178–203, 2011.

Q. Mei, X. Shen, and C. Zhai. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 490–499, 2007.