Online Variational LDA for topic modeling on Gutenburg documents

Sahil Tyagi

December 4, 2020

The tasks of this project can be broadly categorized into one of the two below:

0.1 Web scraping and Data Preprocessing

The implementation of this section uses Python3 and available in the jupyter notebook ***scraper_doc_parse.ipynb***. After Prof. Khardon's suggestion, I chose all the books authored by Jane Austen as a starting point to perform topic modeling for a subset of gutenburg documents. We start with the URLs of all books written by Jane Austen in *jane_austen.txt* and use the popular python library ***beautifulsoup*** for parsing and web scraping. Each book is written to a plain text file located in the *books* directory.

After obtaining all the required documents locally, I use the python library ***nltk*** for cleaning and preprocessing. My motivation for writing this segment of the project in Python3 was that nltk is only compatible with this version of python. Using nltk's stopwords corpus, I iterate through each book and remove common stopwords (like the, a, is, if, etc.). In addition, I got rid of useless and redundant pronouns, prepositions, conjunctions, interjections, verbs and adverbs. After performing all of the above, I finally got a clean relevant set of documents to use for topic modeling.

```
what_to_filter_out = ['CC','EX','IN','PRP','PRP$','UH','WDT','WP','WP$','WRB','VB','VBG','VBD','VBP','VBZ',
                      'DT','EX','.','RB']
```

(a) NLTK stopwords filters used

0.2 Online Variational LDA implementation

I had initially intended to implement the conventional batch Latent Dirichlet Allocation (LDA) for topic modeling on gutenberg documents as we did in programming assignment 4. As per Prof. Khardon's suggestion, I moved from typical LDA to online variational LDA for faster performance on larger corpus of documents, which applies well to the gutenberg documents dataset that I use for thing project. The variational LDA algorithm is inspired from the paper *Online Learning for Latent Dirichlet Allocation* by Hoffmann et al. Online variational LDA is implemented in ***Online_Variational_LDA.ipynb*** using popular python2 libraries like numpy and scipy.

The topic selection for LDA was based on the top-10 frequency counts from each document. Hence, for 10 documents we have 100 topics that are to be modeled. The topics are listed in *frequent_topics.txt*. The vocabulary size is 16128 for the case of Jane Austen. Thus, for 10 documents, the size of the document-word matrix is (10 x 16128). I define θ as the Dirichlet distribution over the topics for each document as a (10 x 100) matrix, i.e., $\theta \sim \text{Dirichlet}(\alpha)$. Similarly, I define a topic distribution over the words as a (100 x 16128) matrix such that $\beta \sim \text{Dirichlet}(\eta)$. Like the paper, I implement the ***E-step*** and the ***M-step***, where the dirichlet expectation for $\log\theta$ and $\log\beta$ are calculated using scipy's digamma function. From the dirichlet expectations, I calculate the ϕ which is the exponential sum of expected $\log\theta$ and $\log\beta$. Finally in the E-step, γ is calculated using dot product of ϕ and the document-word matrix; repeating until changes in γ drop to < 0.00001 (as used in the main paper).