Batch and Online Variational LDA for topic modeling on Gutenburg documents

Sahil Tyagi

December 12, 2020

Abstract

This project is part of the course B555-Machine Learning taught by Prof. Roni Khardon at Indiana University, Bloomington, Indiana, USA. The author of this report builds and implements batch and online variational Latent Dirichlet Allocation (LDA) from scratch as postulated in the paper *Online Learning for Latent Dirichlet Allocation* by *Hoffman et al.*. The deviation from the interim report is that the author was able to test and implement generative LDA with Gibbs sampler as well for the sake of batter comparison purposes. The data used for this project is all sets of gutenberg documents under the name of the popular english classic novel writer Jane Austen. The end results to fetch the topics for a given document is done via a web service launched as a serverless cloud function on the Google Cloud Platform (GCP). I would like to thank GCP for providing free credits as part of their university education program.

1 LDA generative model

To understand online variational LDA, we build on top of the LDA generative model that we studied in the course (and as part of our programming assignment 4). The solution is fondly called topic modeling, since we try to infer topics on a set of documents and each topic has an influence on a set of words (vocabulary of the corpus). The solution is a kind of dimensionality reduction where we break down the corpus of documents and their associated words into documents with the associated topics and the topics with their associated words as shown in Fig 1.

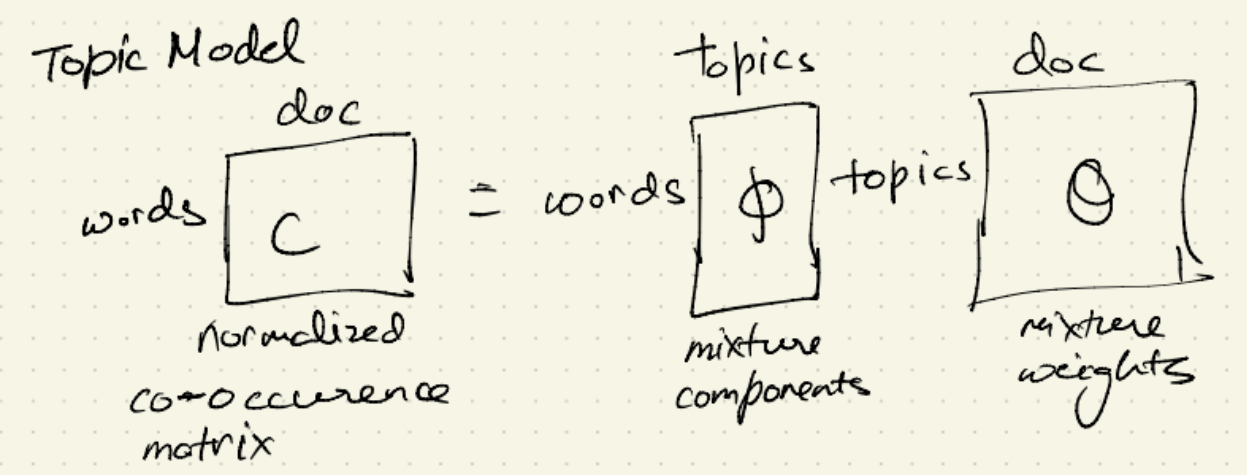

Figure 1: Corpus of documents breakdown

Here, the topic distribution over the documents is a dirichlet parameterized by α and the word distribution over the topics is also a dirichlet parameterized by β:

$\forall$ d draw $\theta_{d|k} \sim$ Dirichlet$(\theta_d|\alpha)$ and $\forall$ k draw $\phi i|k \sim$ Dirichlet$(\phi_d|\beta)$

Then, for every document d and every location j in d:

$$draw \quad z^{j,d} \sim \theta_{k|d} \ and \ draw \quad w^{j,d} \sim \phi k|d = z^{j,d}$$

The illustration of the generative LDA model from lectures is shown in Fig 2.

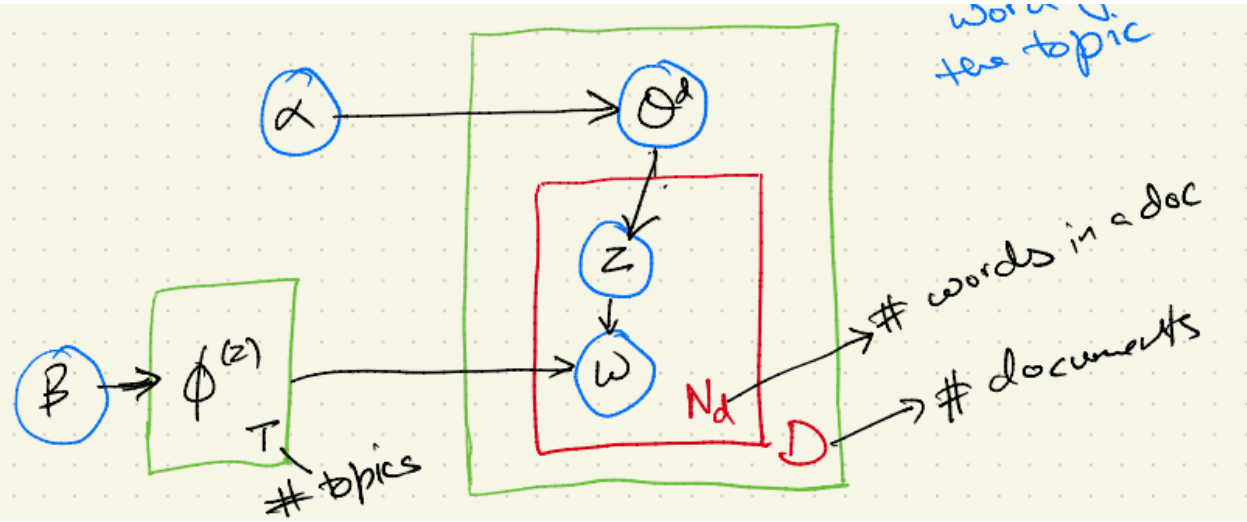

Figure 2: LDA illustration from B555 lectures

Without going over the derivations and calculations, the probability of topics over the documents is given in Fig 3.

$$P(k) = \frac{C_t(k,word)+\beta}{V\beta+\sum_j C_t(k,j)} \frac{C_d(doc,k)+\alpha}{K\alpha+\sum_l C_d(doc,l)}$$

Figure 3: Key compute metric in LDA taken from Programming Problem 4

2 Online Variational LDA

As suggested by Prof. Khardon, I looked into online variational algorithms since the size of the corpus was enormous. To understand how variational LDA works, I referred to *Online Learning for Latent Dirichlet Allocation* by *Hoffman et al.*. Online variational LDA is an extension of batch variational LDA based on online stochastic optimization with a natural gradient step. The advantage of online variational LDA is that it has much better computational efficiency while still having the same (or even better) statistical efficiency as batch variational LDA.

In variational LDA, we try to maximize the Expectation Lower Bound (ELBO) in order to minimize the KL divergence between the distribution $q(z, \theta, \beta)$ and the true posterior $p(z, \theta, \beta|w, \alpha, \eta)$.

Hoffman et al. derive the expectations of topic distribution $log\theta_{dk}$ and word distribution $log\beta_{kw}$ as show. below:

$$E_q[log\theta_{dk}] = \Psi(\gamma_{dk}) - \Psi(\Sigma^K \gamma_{di}) \quad and \quad E_q[log\beta kw] = \Psi(\lambda_{kw}) - \Psi(\Sigma^K \lambda_{ki})$$

where Ψ is the digamma function (the first derivative of the logarithmic of the gamma function), and γ and λ functions are given as:

$$\gamma_{dk} = \alpha + \Sigma_w n_{dw} \phi_{dwk} \quad and \quad \lambda_{kw} = \eta + \Sigma_d n_{dw} \phi_{dwk}$$

The variational parameter ϕ is given as:

$$\phi_{dwk} \propto exp[E_q[log\theta_{dk}] + E_q[log\beta kw]]$$

From the Hoffman paper, the pseudo algorithm for batch variational LDA is shown in Fig 4.

Algorithm 1 Batch variational Bayes for LDA

Initialize $\boldsymbol{\lambda}$ randomly.
while relative improvement in $\mathcal{L}(\boldsymbol{w}, \boldsymbol{\phi}, \boldsymbol{\gamma}, \boldsymbol{\lambda}) > 0.00001$ **do**
  E step:
  for $d = 1$ to D **do**
    Initialize $\gamma_{dk} = 1$. (The constant 1 is arbitrary.)
    repeat
      Set $\phi_{dwk} \propto \exp\{\mathbb{E}_q[\log \theta_{dk}] + \mathbb{E}_q[\log \beta_{kw}]\}$
      Set $\gamma_{dk} = \alpha + \sum_w \phi_{dwk} n_{dw}$
    until $\frac{1}{K} \sum_k |\text{change in} \gamma_{dk}| < 0.00001$
  end for
  M step:
  Set $\lambda_{kw} = \eta + \sum_d n_{dw} \phi_{dwk}$
end while

Figure 4: Batch Variational LDA from Hoffman paper

From the same paper, the algorithm for online variational LDA is shown in Fig 5.

Algorithm 2 Online variational Bayes for LDA

Define $\rho_t \triangleq (\tau_0 + t)^{-\kappa}$
Initialize $\boldsymbol{\lambda}$ randomly.
for $t = 0$ to ∞ **do**
  E step:
  Initialize $\gamma_{tk} = 1$. (The constant 1 is arbitrary.)
  repeat
    Set $\phi_{twk} \propto \exp\{\mathbb{E}_q[\log \theta_{tk}] + \mathbb{E}_q[\log \beta_{kw}]\}$
    Set $\gamma_{tk} = \alpha + \sum_w \phi_{twk} n_{tw}$
  until $\frac{1}{K} \sum_k |\text{change in} \gamma_{tk}| < 0.00001$
  M step:
  Compute $\tilde{\lambda}_{kw} = \eta + D n_{tw} \phi_{twk}$
  Set $\boldsymbol{\lambda} = (1 - \rho_t)\boldsymbol{\lambda} + \rho_t \tilde{\boldsymbol{\lambda}}$.
end for

Figure 5: Online Variational LDA from Hoffman paper

3 Code and implementation

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 (in Fig 6), 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. These documents can be found inside the **books** directory.

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

Figure 6: NLTK stopwords filters used

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 LDA generative model with Gibbs sampler is implemented in **LDA_gibbssampler.ipynb**. Just like programming assignment 4, I compute the matrices **C_d** and **C_t**.

To run both Gibbs sampler as well as online variational LDA, we selected the number of topics to be 100. 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. This is the only function from the library scipy in my project. 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).

4 Webservice deployment

For each document processed, I build a webservice to return the top-3 topics given the document name. For example, the accesible link is *https://us-central1-cobalt-ion-289207.cloudfunctions.net/function-1?name=Emma*.

Some other names to use are: Emma, Lady Susan, Love and Freindship, Mansfield Park, Northanger Abbey, Persuasion, Pride and Prejudice, Sense and Sensibility, The Letters of Jane Austen, The Watsons By Jane Austen and Concluded by L. Oulton