Assignment 3: Topic Modeling Using Gibbs Sampler

April 13, 2017

Description of the Assignment

In this assignment, we implement latent Dirichlet allocation (LDA) for topic modeling of text data. Our goal in this assignment is to implement a Gibbs sampler for LDA topic inference using their representative words. We will use a subset of the 20 newsgroups dataset (given with this assignment). The subset consists of 200 documents that have been pre-processed and "cleaned" so they do not require any further manipulation, i.e., you only have to read the space-separated strings from the ASCII text files. Each document belongs to one of two classes. The file index.csv holds the true labels of each document.

Recall that Gibbs sampling is a type of inference method that can be used in cases where the exact posterior distribution is intractable. The output of this process is a sample from the true posterior distribution. In the case of LDA, the output represents a sample of the (hidden) topic variables for each word. This sample of topic variables can be used to calculate topic representations per document. Algorithm 1 describes one version of a Gibbs sampler.

For this assignment, fix the Dirichlet hyperparameter for the topic distribution is $\alpha = \frac{50}{K}$ and $\eta = .1$ where K is the number of topics. We have provided an additional smaller dataset, artificial.zip, for developing your implementation. Running your sampler on this dataset with $K = 2$ and the above parameters, you should find the three most frequent words per topic to be {bank, river, water} and {dollars, bank, loan} (not necessarily in those orders). Once you have verified that your implementation works correctly, run your sampler with $K = 20$ on the 20 newsgroups dataset. After the sampler has finished running, output the 5 most frequent words of each topic into a CSV file, topicwords.csv, where each row represents a topic. Include these results in both your report and submission. All the necessary notations, equations and pseudocodes are given below. Please allow yourself sufficient time to implement this. Running the Gibbs sampler is likely to be a time consuming task.

Number of topics K
Number of documents D
Number of entries in vocabulary V
Total number of words in corpus (collection of D documents) N
Word i in the corpus is w_i
Number of words generated from topic t in document δ is n_t^{δ}
Pseudo count of words generated from any topic in document δ is α
Number of times word w is generated from topic t in the corpus is $n_{t,w}$
Pseudo count of times any word is generated from topic$t$ is η
Fraction of words generated from topic (proportion of topic) t in document δ is $\theta_{\delta,t}$
Probability of a word w in topic t is $\beta_{t,w}$
An estimate of $\theta_{\delta,t}$ using
Probability of a word w in topic t is $\beta_{t,w}$

$$n_t^{\delta} = \sum_{i=1}^{N} \mathbb{1}\left[z_i^{\delta} = t\right]$$

$$n_{t,w} = \sum_{i=1}^{N} \mathbb{1}[z_i = t \ \wedge w_i = w]$$

$$\theta_{\delta,t} = \frac{\alpha + n_t^{\delta}}{K\alpha + \sum_{k=1}^{K} n_k^{\delta}}$$

$$\beta_{t,w} = \frac{\eta + n_{t,w}}{V\eta + \sum_{v=1}^{V} n_{t,v}}$$

$$\theta_{\delta,t}^{-i} = \frac{\alpha + n_t^{\delta,-i}}{K\alpha + \sum_{k=1}^{K} n_k^{\delta,-i}}$$

$$\beta_{t,w}^{-i} = \frac{\eta + n_{t,w}^{-i}}{V\eta + \sum_{v=1}^{V} n_{t,v}^{-i}}$$

$$p\left(z_i^{\delta} = t \middle| \mathbf{W}, \mathbf{Z}_{-i}\right) = \frac{\frac{\left(\alpha + n_t^{\delta,-i}\right)}{K\alpha + \sum_{k=1}^{K} n_k^{\delta,-i}} \frac{\left(\eta + n_{t,w}^{-i}\right)}{V\eta + \sum_{v=1}^{V} n_{t,v}^{-i}}}{\sum_{t=1}^{K} \frac{\left(\alpha + n_t^{\delta,-i}\right)}{K\alpha + \sum_{k=1}^{K} n_k^{\delta,-i}} \frac{\left(\eta + n_{t,w}^{-i}\right)}{V\eta + \sum_{v=1}^{V} n_{t,v}^{-i}}}$$

Algorithm 1: Gibbs Sampler for Latent Dirichlet Allocation

Initialize topic indices **Z**
Initialize word indices **W**
Initialize $D \times K$ matrix with n_t^{δ}
Initialize $K \times V$ matrix with $n_{t,v}$
Initialize $1 \times K$ array of probabilities P (to zero)
for $r = 1$ to N_{iters} **do**
  for $i = 1$ to N **do**
    $w \leftarrow w_i^{\delta}$
    $t \leftarrow z_i^{\delta}$
    $n_t^{\delta} \leftarrow n_t^{\delta} - 1$
    $n_{t,v} \leftarrow n_{t,v} - 1$
    for $t = 1$ to K **do**
      $P[t] = \frac{\left(\alpha + n_t^{\delta,-i}\right)}{K\alpha + \sum_{k=1}^{K} n_k^{\delta,-i}} \frac{\left(\eta + n_{t,w}^{-i}\right)}{V\eta + \sum_{v=1}^{V} n_{t,v}^{-i}}$
    Normalize P
    Sample $t \sim P$
    $z_i^{\delta} \leftarrow t$
    $n_t^{\delta} = n_t^{\delta} + 1$
    $n_{t,v} = n_{t,v} + 1$
return

Instructions for Report Writing

To understand the behavior of our Gibbs sampler we want to investigate following scenarios.

1. When does the sampler converge if you don't consider burn-in and lag? (The 5 most frequent words of each topic remain same after certain iterations). Test for $N_{iters} = 500, 1000, 2000, 5000, 10000$ …
2. Is there any effect of burn-in? (The 5 most frequent words of each topic make more sense if we discard burn-in iterations). Test for $N_{burn-in} = 100, 200, 400, 1000, 2000$ …
3. Is there any effect of stride/lag? Test for lag $L = 5, 10, 20, 50$ …

Special Instructions

1. Don't Copy anything! If you do copy from internet or from any other person or from any other source, you will be severely punished and it is obvious. More than that, we expect Fairness and honesty from you. Don't disappoint us!
2. The report should be in .docx/.pdf (No hardcopy is required). Write precisely in your own language and keep it as simple as possible.
3. Upload the code and report in Moodle within 10:00 P.M. of 12th March, 2017 (Sunday). This is a strict system-imposed deadline for both section A and B.
4. For Python and Matlab, you may not get supporting software in the lab. If you do program in these languages, bring your computer in the sessional.
5. You are allowed to show the assignment in your own laptop during the final submission. But in that case, ensure an internet connection as you have to instantly download your code from the Moodle and show it.

Instructions for Moodle upload

1. Upload the assignment within the specified time. Otherwise, we can't accept it.
2. If you write code in a single file, then rename it as <Student id> <code>.<extension>. For example, if your student id is 1105123 and you have done in java, then your file name should be "1105123 code.java".
3. The report name should be <Student id> <report>.<extension>. For example, if your student id is 1105123 and it is in pdf format, then the report name should be "1105123 report.pdf".
4. Finally make a main folder, put the code and report in it, and rename the main folder as your <Student id> <Programming language>. For example, "1105123 Java". Then zip it and upload it.