Parameter Estimation for the Latent Dirichlet Allocation

Jaka Špeh, Andrej Muhič, Jan Rupnik
Artificial Intelligence Laboratory
Jožef Stefan Institute
Jamova cesta 39, 1000 Ljubljana, Slovenia
e-mail: {jaka.speh, andrej.muhic, jan.rupnik}@ijs.si

ABSTRACT

We review three algorithms for parameter estimation of the Latent Dirichlet Allocation model: batch variational Bayesian inference, online variational Bayesian inference and inference using collapsed Gibbs sampling. We experimentally compare their time complexity and performance. We find that the online variational Bayesian inference converges faster than the other two inference techniques, with comparable quality of the results.

1 INTRODUCTION

Probabilistic graphical models such as Latent Dirichlet Allocation (LDA) allow us to describe textual documents as a distribution over topics, where the topics are represented as distributions over words. Given a collection of documents, the task of LDA parameter estimation is to find the most likely per-document topic distributions and the most likely topic distributions. The task is based on computing the LDA posterior distribution, which is known to be intractable, but can be tackled by using approximate inference methods.

Modern approximate posterior inference algorithms fall into two categories: sampling approaches and optimization approaches. The sampling approaches are usually based on Markov Chain Monte Carlo (MCMC) sampling. The conceptual idea of these methods is to generate independent samples from the posterior and then reason about the documents and topics. The second category of approaches are the optimization approaches, usually based on variational inference, also called the Variational Bayesian (VB) methods. These methods optimize the closeness (based on the Kullback-Leibler divergence) of the posterior to a simplified parametric distribution.

In this paper, we compare one MCMC and two VB algorithms for approximating the posterior distribution. In the subsequent sections we formally introduce the LDA model and review the inference algorithms. We study the performance of algorithms and make comparisons between them. We use articles from Wikipedia to infer and evaluate the models. We show that Online Variational Bayesian inference is the fastest algorithm. However the accuracy is lower than in the other two, but the results are still good enough for practical use.

2 LDA MODEL

Latent Dirichlet Allocation [1] is a Bayesian probabilistic graphical model, which is regularly used in topic modeling. It assumes M documents are built in the following fashion. First, a collection of K topics (distributions over words) are drawn from a Dirichlet distribution, $\varphi_k \sim \text{Dirichlet}(\beta)$. Then for m-th document, we:

1. Choose a topic distribution $\theta_m \sim \text{Dirichlet}(\alpha)$.
2. For each word $w_{m,n}$ in m-th document:
    i. choose a topic of the word $z_{m,n} \sim \text{Multinomial}(\theta_m)$,
    ii. choose a word $w_{m,n} \sim \text{Multinomial}(\varphi_{z_{m,n}})$.

LDA can be graphically represented using plate notation (Figure 1).

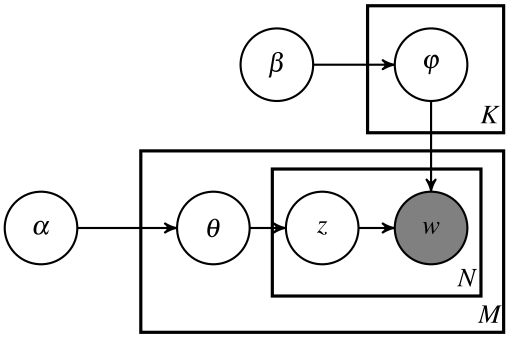

FIGURE 1. Plate notation of LDA.

The total probability of the LDA model is:

$$p(\mathbf{w},\mathbf{z},\theta,\varphi \mid \alpha,\beta) = \prod_{k=1}^{K} p(\varphi_k|\beta) \prod_{m=1}^{M} \left(p(\theta_m|\alpha) \prod_{n=1}^{N_m} p(z_{m,n}|\theta_m) p(w_{m,n}|\varphi_{z_{m,n}}) \right).$$

We can analyze a corpus of documents by computing the posterior distribution of the hidden variables $(\mathbf{z},\theta,\varphi)$ given a document $(\mathbf{w})$. This posterior reveals the latent structure in the corpus that can be used for prediction or data exploration. Unfortunately, this distribution cannot be computed directly [1], and is usually approximated using Markov Chain Monte Carlo (MCMC) methods or variational inference.

3 ALGORITHMS

In the following subsections, we will derive one MCMC algorithm and two variational Bayes algorithms for the approximation of the posterior inference.

3.1 Collapsed Gibbs sampling

In the collapsed Gibbs sampling we first integrate θ and φ out.

$$p(\mathbf{z},\mathbf{w} \mid \alpha,\beta) = \int_\theta \int_\varphi p(\mathbf{z},\mathbf{w},\theta,\varphi \mid \alpha,\beta)\, d\theta\, d\varphi.$$

The goal of collapsed Gibbs sampling here is to approximate the distribution $p(\mathbf{z} \mid \mathbf{w},\alpha,\beta)$. The conditional probability $p(\mathbf{w} \mid \alpha,\beta)$ does not depend on $\mathbf{z}$, therefore Gibbs sampling equations can be derived from $p(\mathbf{z},\mathbf{w} \mid \alpha,\beta)$ directly. Specifically, we are interested in the following conditional probability:

$$p(z_{m,n} \mid \mathbf{z}_{\neg(m,n)},\mathbf{w},\alpha,\beta),$$

where $\mathbf{z}_{\neg(m,n)}$ denotes all z-s but $z_{m,n}$. And furthermore we assume that the omitted word is the v^{th} word in the vocabulary of size V. Note that for collapsed Gibbs sampling we need only to sample a value for $z_{m,n}$ according to the above probability. Thus we only need the probability mass function up to scalar multiplication. Moreover we simplify the model by taking $\alpha_k = \alpha$, $\beta_k = \beta$ for all k. The distribution can be simplified [4, page 22] as:

$$\text{(1)} \qquad p(z_{m,n}=k \mid \mathbf{z}_{\neg(m,n)},\mathbf{w},\alpha,\beta) \propto \frac{n^{(v)}_{k,\neg(m,n)}+\beta}{\sum_{t=1}^{V}(n^{(t)}_{k,\neg(m,n)}+\beta)}\,(n^{(k)}_{m,\neg(m,n)}+\alpha),$$

where $n_k^{(v)}$ refers to the number of times that term v has been observed with topic k, $n_m^{(k)}$ refers to the number of times that topic k has been observed with a word of document m, and $n^{(\cdot)}_{\cdot,\neg(m,n)}$ indicate that the n-th token in m-th document is excluded from the corresponding $n_k^{(v)}$ or $n_m^{(k)}$.

The topics and document topic mixtures can be obtained by [4, page 23]:

$$\varphi_{k,v} = \frac{n_k^{(v)}+\beta}{\sum_{t=1}^{V}(n_k^{(t)}+\beta)}, \quad \theta_{m,k} = \frac{n_m^{(k)}+\alpha}{\sum_{k=1}^{K}(n_m^{(k)}+\alpha)}.$$

In collapsed Gibbs sampling algorithm, we need to remember values of three variables: $z_{m,n}$, $n_m^{(k)}$, and $n_k^{(v)}$, and some sums of these variables for efficiency. The algorithm first initializes $\mathbf{z}$ and computes $n_m^{(k)}$, $n_k^{(v)}$ according to the initialized values. Then in each iteration, the algorithm makes a pass over all the words in all the documents, samples values of $z_{m,n}$ according to Equation (1), and recomputes $n_m^{(k)}$ and $n_k^{(v)}$. Then one has to decide when the Markov chain has converged and which initial samples to discard ("burn in" process).

3.2 Variational Bayesian inference

This algorithm was proposed in the original LDA paper [1]. In Variational Bayesian inference (VB) the true posterior is approximated by a simpler distribution $q(\mathbf{z},\theta,\phi)$, which is indexed by a set of free parameters [6]. The simplified distribution is illustrated using plate notation in Figure 2. We choose a fully factorized distribution q of the form:

$$q(z_{m,n}=k) = \psi_{m,n,k},$$
$$q(\theta_m) = \text{Dirichlet}(\theta_m \mid \gamma_m),$$
$$q(\varphi_k) = \text{Dirichlet}(\varphi_k \mid \lambda_k).$$

The posterior is parameterized by ψ, γ and λ. We refer to λ as corpus topics and γ as documents topics.

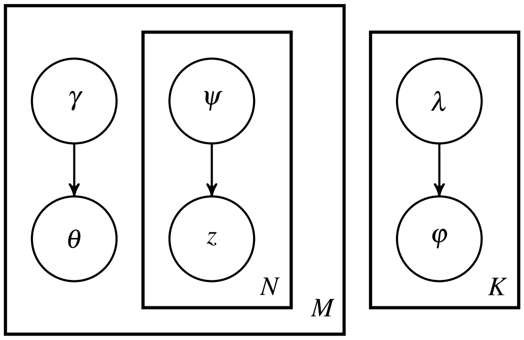

FIGURE 2. Plate notation of parameterized distribution q.

The parameters are optimized to maximize the Evidence Lower Bound (ELBO):

$$\text{(2)} \quad \log p(\mathbf{w} \mid \alpha,\beta) \geq \mathscr{L}(\mathbf{w},\psi,\gamma,\lambda) = \mathbb{E}_q[\log p(\mathbf{w},\mathbf{z},\theta,\varphi \mid \alpha,\beta)] - \mathbb{E}_q[\log q(\mathbf{z},\theta,\varphi)].$$

Maximizing the ELBO is equivalent to minimizing the Kullback-Leibler divergence between $q(\mathbf{z},\theta,\varphi)$ and the posterior $p(\mathbf{z},\theta,\varphi \mid \mathbf{w},\alpha,\beta)$.

ELBO $\mathscr{L}$ can be optimized using coordinate ascent over the variational parameters (detailed derivation in [1, 2]):

$$\text{(3)} \qquad \psi_{m,v,k} \propto \exp\left\{\mathbb{E}_q[\log\theta_{m,k}] + \mathbb{E}_q[\log\varphi_{k,v}]\right\},$$

$$\text{(4)} \qquad \gamma_{m,k} = \alpha + \textstyle\sum_{v=1}^{V} n_{m,v}\psi_{m,v,k},$$

$$\text{(5)} \qquad \lambda_{k,v} = \beta + \textstyle\sum_{m=1}^{M} n_{m,v}\psi_{m,v,k},$$

where $n_{m,v}$ is the number of terms v in document m. The expectations are

$$\mathbb{E}_q[\log \theta_{m,k}] = \Psi(\gamma_{m,k}) - \Psi\left(\sum_{\widetilde{k}=1}^{K} \gamma_{m,\widetilde{k}}\right),$$
$$\mathbb{E}_q[\log \varphi_{k,v}] = \Psi(\lambda_{k,v}) - \Psi\left(\sum_{\widetilde{v}=1}^{V} \lambda_{k,\widetilde{v}}\right),$$

where Ψ denotes the digamma function (the first derivative of the logarithm of the gamma function).

The updates of the variational parameters are guaranteed to converge to a stationary point of the ELBO. We can make some parallels with Expectation-Maximization (EM) algorithm [3]. Iterative updates of γ and ψ until convergence, holding λ fixed, can be seen as the "E"-step, and updates of λ, given γ and ψ, can be seen as the "M"-step.

The VB inference algorithm first initializes λ randomly. Then for each documents it does the "E"-step: initializes γ randomly and then until γ converges does the coordinate ascent using Equations (3) and (4). After γ converges, the algorithm performs the "M"-step: sets λ using Equation (4). Each combination "E" and "M"-step improves ELBO. VB inference finishes after relative improvement of $\mathscr{L}$ is less than a pre-prescribed limit or after we reach maximum number of iterations. We define an iteration as "E" + "M"-step. After the algorithm converges, the parameters γ represent document topics and λ represents corpus topics.

3.3 Online Variational Bayesian inference

The previously described algorithm has constant memory requirements. It requires a full pass through the entire corpus on each iteration. Therefore, it is not naturally suited to the online setting. We now present a variant of the algorithm that is more suitable in this case.

The first step is to factorize the ELBO (Equation (2)) into:

$$\begin{aligned}
&\mathscr{L}(\mathbf{w}, \psi, \gamma, \lambda) = \\
&\quad \textstyle\sum_{m=1}^{M} \left\{ \mathbb{E}_q[\log p(\mathbf{w}_m \mid \theta_m, \mathbf{z}_m, \varphi)] + \mathbb{E}_q[\log p(\mathbf{z}_m \mid \theta_m)] \right. \\
&\quad - \mathbb{E}_q[\log q(\mathbf{z}_m)] + \mathbb{E}_q[\log p(\theta_m \mid \alpha)] - \mathbb{E}_q[\log q(\theta_m)] \\
&\quad \left. + \left(\mathbb{E}_q[\log p(\varphi \mid \beta)] - \mathbb{E}_q[\log q(\varphi)]\right)/M \right\}.
\end{aligned}$$

Note that we bring the per corpus topics terms into the summation over documents, and divide them by the number of documents M. This allows us to look at the maximization of the ELBO according to the parameters ψ and γ for each document individually. Therefore, we first maximize ELBO according to the ψ and γ as in the batch algorithm with λ fixed. Then fix ψ and γ and maximize the ELBO over λ, as we will now describe. Let $\gamma(w_m, \lambda)$ and $\psi(w_m, \lambda)$ be the values of γ_m and ψ_m produced by the "E"-step. Our goal is to find λ that maximizes

$$\mathscr{L}(\mathbf{w}, \lambda) = \textstyle\sum_{m=1}^{M} \ell_m\left(\mathbf{w_m}, \gamma(\mathbf{w_m}, \lambda), \psi(\mathbf{w_m}, \lambda), \lambda\right),$$

where $\ell_m(\mathbf{w_m}, \gamma(\mathbf{w_m}, \lambda), \psi(\mathbf{w_m}, \lambda), \lambda)$ is the m-th document's contribution to ELBO.

Then we compute $\widetilde{\lambda}$, the setting of λ that would be optimal with given ψ if our entire corpus consisted of a single document w_m repeated M times:

$$\widetilde{\lambda}_{k,v} = \beta + M n_{m,v} \psi_{m,v,k}.$$

Here M is the number of available documents, the size of the corpus. Then we update λ using a convex combination of its previous value and $\widetilde{\lambda}$: $\lambda = (1 - \rho_m)\lambda + \rho_m \widetilde{\lambda}$, where the weight is defined as $\rho_m := (\tau_0 + m)^{-\kappa}$. The parameters κ and τ_0 have the following interpretation: $\tau_0 \geq 0$ slows down the early iterations of the algorithm and $\kappa \in (0.5, 1]$ controls the rate at which old values $\widetilde{\lambda}$ are forgotten. This choice of parameters is essential to ensure convergence, see [5, Subsection 2.3].

To sum up, the algorithm first initializes λ randomly. Then, given a document, it performs the "E"-step as in Variational Bayesian inference. Next it updates λ as discussed above. Finally it moves on to the new document and repeats the process. The algorithm terminates after all documents have been processed. Online Variational Bayesian inference (Online VB) was proposed by Hofffman, Blei and Bach in [5].

4 EXPERIMENTS

We ran several experiments to evaluate algorithms of the LDA model. Our purpose was to compare the time complexity and performance of previously described algorithms. For training and testing corpora we used Wikipedia.

Effectiveness was measured by using perplexity on held-out data, which is defined as

$$\text{perplexity}(\mathbf{w}_{\text{test}}, \lambda) = \exp\left\{ -\frac{\sum_{m=1}^{M} \log p(\mathbf{w_m} \mid \lambda)}{\sum_{m=1}^{M} N_m} \right\},$$

where N_m denotes number of words in m-th document. Since we cannot directly compute $\log p(\mathbf{w_m} \mid \lambda)$, we use ELBO as approximation:

$$\begin{aligned}
&\text{perplexity}(\mathbf{w}_{\text{test}}, \lambda) \\
&\leq \exp\left\{ -\textstyle\sum_{m=1}^{M} (\mathbb{E}_q[\log p(\mathbf{w_m}, \mathbf{z_m}, \theta_m \mid \varphi)] \right. \\
&\quad \left. - E_q[\log q(\mathbf{z_m}, \theta_m \mid \varphi)]) / \textstyle\sum_{m=1}^{M} N_m \right\}.
\end{aligned}$$

We tested three algorithms and ran experiments with varying sizes of training sets: 10,000, 20,000, ..., 80,000. Later we evaluated perplexity on 100 held-out documents. Size of vocabulary was approximately 150,000 words.

In all experiments components of α and β were set to 0.01 and the number of topics K was set to 100. Collapsed Gibbs sampling exhibited problems with convergence of the model parameters: the relative change in $\mathbf{z}$ variable was never dropped bellow 20% in 1000 iterations. In VB inference, the "E"-step and the "M"-step converge if relative change in γ is under 0.001 and relative improvement of the ELBO is under 0.001, respectively. In the Online VB, the convergence of "E" step is determined the same way is in the batch VB inference. Batchsize was 100 documents, τ_0 was 1024 and κ was equal to 0.7 as proposed in [5].

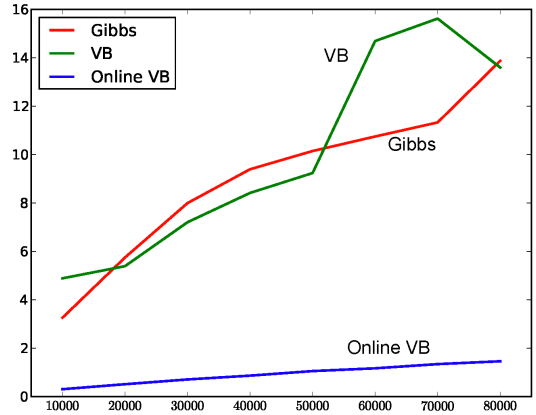

FIGURE 3. Time used by the algorithms (in hours) given the number of the documents.

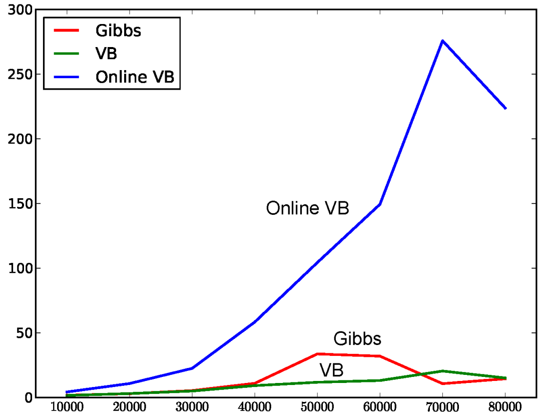

FIGURE 4. Perplexity on held-out documents as a function of number of documents analyzed.

The fastest algorithm is Online VB (see Figure 3), batch VB has a higher time complexity, while Gibbs sampling algorithm did not fully converge.

We would like to compare our results to [5]. So we choose the perplexity on held-out data as the model fit. When measuring the perplexity on held-out data (lower perplexity corresponds to a "better" model) we observed two things: the perplexity slightly increased as the training set size increased in case of batch VB and collapsed Gibbs sampling, while it dramatically increased for the Online VB method (see Figure 4). The results are unexpected: increasing the training set size from 10,000 to 98,000 for the batch VB (Figure 2 in [5]) decreased the perplexity, whereas an increase was observed in our experiments. The behaviour of online VB is drastically different than the one reported in [5]. Note however, that we only computed an upper bound on the perplexity, since computing it exactly is not tractable. This means that the particular method of evaluation gives us very little information on the performance of Online VB. The quality of the topics learned by Online VB was estimated as good based on visual inspection, which could be evidence of the perplexity bound being loose or some instability in computation. The other reason for the different behaviour of the perplexity bound when comparing our work and [5] might lie in the big difference between the vocabulary sizes: 150,000 in our study vs 4,253. Our future goal is to gain a further insight into this issue.

Based on the experiments, the authors recommend using the online VB algorithm for large corpora with large sizes of vocabularies, since scalability becomes an important factor.

5 ACKNOWLEDGMENT

The authors gratefully acknowledge that the funding for this work was provided by the project XLIKE (ICT-257790-STREP).

REFERENCES

[1] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[2] Wim De Smet and Marie-Francine Moens. Cross-language linking of news stories on the web using interlingual topic modelling. In *Proceedings of the 2nd ACM workshop on Social web search and mining*, SWSM '09, pages 57–64, New York, NY, USA, 2009. ACM.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.

[4] Gregor Heinrich. Parameter estimation for text analysis. Technical report, Fraunhofer IGD, Darmstadt, Germany, 2005.

[5] Matthew D. Hoffman, David M. Blei, and Francis R. Bach. Online learning for latent dirichlet allocation. In *NIPS*, pages 856–864. Curran Associates, Inc., 2010.

[6] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Mach. Learn.*, 37(2):183–233, November 1999.