Latent Dirichlet Allocation

Kaspar Märtens **Giuseppe Di Benedetto** **Jack Jewson** **Qinyi Zhang**

Abstract

We describe the Latent Dirichlet Allocation (LDA) for topic modelling and provide implementations in R, using variational and stochastic variational inference. We also demonstrate its performance on a dataset of 1500 NIPS papers.

1 Introduction

In this report we describe and present the results of the implementation of the *Latent Dirichlet Allocation* (LDA) [1], a Bayesian model widely used in the topic modeling problem. The goal of topic modeling is to reveal latent semantic structure in text documents, e.g. to annotate large collections of documents with their main themes, and then cluster or summarize these. For example, the underlying structure for a machine learning paper about image classification may contain words mainly related to the following topics: images (such as pixel, intensity), neural networks (such as layers, backpropagation) and probabilistic terms (such as likelihood, gaussian).

First, let us provide a brief overview of the other latent variable models that have been used in the past as generative models for topic modeling, underlying their inefficiencies. Let V be the number of words in the vocabulary, M be the number of documents in the corpus, each one with N words and k be the number of topics. Each of the following models sample the words according to discrete distributions that can be seen as points in the $(V-1)$-simplex.

- **Mixture of unigrams model**: k points in the simplex are fixed, defining the topic-simplex and for each document, one of this corners is chosen randomly and all the words will be sampled according to that distribution over the words. The main drawback of this model is the unrealistic assumption that each document belongs to just one topic.
- **Probabilistic LSI [4]** is more flexible: each document in the training set is assigned a topic distribution (i.e. a point inside the topic simplex), and each word in the document is sampled as follows: first sample a topic from that topic-distribution and then the word from the topic. Although this model allows each document to contain more than one topic, it is not possible to model the distribution of topics for a document outside the training set. Moreover the number of parameters increases linearly with the number of documents M, which leads to overfitting.

LDA allows documents to contain multiple topics while having a number of parameters independent from M. In fact, for each document a point in the topic-simplex is sampled randomly from a continuous distribution over the topic-simplex. Once the distribution of topics has been sampled, each word in the document is generated according this. Behind these models there is a key assumption, also known as *bag-of-words*, according to which all the words in a document are exchangeable. This assumption is equivalent, by the De Finetti representation theorem, to the existence of a latent variable, such that the exchangeable variables are iid, when conditioned on that latent variable. This is equivalent to considering LDA as a mixture model.

2 Latent Dirichet Allocation

The Latent Dirichet allocation is a three-level hierarchical model (Figure 1) in which the number of topics k is fixed a priori, and there are two corpus-level parameters: $\alpha \in \mathbb{R}^k$ the Dirichlet parameter

for the random multinomial parameters used to sample the topic variables, and $\beta \in \mathbb{R}^{k,V}$ whose rows are the words distribution for each topic.Note that here α and β as treated as fixed parameters, but there is a more Bayesian approach which places a Dirichlet prior over each row of β. Defining a word as a categorical vector of V entries such that $w^i = 1$ if it is the i-th word in the vocabulary and zero otherwise, we can denote a document as a vector of N words $\mathbf{w} = (w_1, \ldots, w_N)$. The generative process for each document in the corpus is the following:

- Sample $\theta \sim \text{Dir}(\alpha)$
- For each word w_n in the document
  - Sample a topic $z_n \sim \text{Multinomial}\,(\theta)$
  - Sample the word $w_n \sim p(w_n \mid z_n, \beta)$ multinomial with parameter the z_n-th row of β

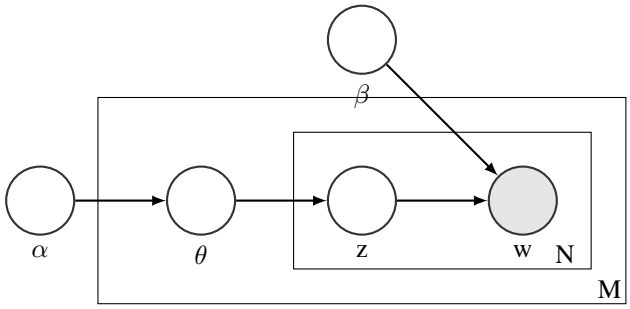

Figure 1: Graphical model representation of LDA.

The joint distribution of the N words in a document, the set of N topics and θ the random parameter of the multinomial over topics is

$$p(\theta, \mathbf{z}, \mathbf{w} \mid \alpha, \beta) = p(\theta \mid \alpha) \left(\prod_{n=1}^{N} p(z_n \mid \theta)\, p(w_n \mid z_n, \beta) \right)$$

therefore the marginal distribution of a document can be obtained integrating over θ and z:

$$p(\mathbf{w} \mid \alpha, \beta) = \int \left(\prod_{n=1}^{N} \sum_{z_n} p(z_n \mid \theta)\, p(w_n \mid z_n, \beta) \right) p(\theta \mid \alpha)\, d\theta\,.$$

It is worth noting that the distribution of words and topics within a document resembles the form of a mixture (infinite mixture in this case) given by the De Finetti theorem:

$$p(\mathbf{w}, \mathbf{z} \mid \alpha, \beta) = \int \underbrace{\left(\prod_{n=1}^{N} p(z_n \mid \theta)\, p(w_n \mid z_n) \right)}_{p(\mathbf{w}, \mathbf{z} \mid \theta, \alpha, \beta)} \underbrace{p(\theta \mid \alpha)\, d\theta}_{\text{De Finetti measure}}\,.$$

2.1 Inference and parameter estimation

We would like to compute the posterior distribution of the latent variables θ and $\mathbf{z}$ given a document $\mathbf{w}$ to explore the structure underlying our data, i.e.

$$p(\theta, \mathbf{z} \mid \mathbf{w}, \alpha, \beta) = \frac{p(\theta, \mathbf{z}, \mathbf{w} \mid \alpha, \beta)}{p(\mathbf{w} \mid \alpha, \beta)},$$

however the denominator of this quantity is intractable. We introduce a possible solution to this problem, given by the mean-field variational inference.

2.1.1 Variational EM

In variational inference, the posterior distribution of the latent variables is approximated by an element from a family of distributions, $q(\theta, \mathbf{z} \,|\, \gamma, \phi)$, parametrized by the so-called variational parameters (γ, ϕ). These are tuned to pick the closest distribution to the target posterior with respect to the Kullback-Leibler distance. This is called the variational distribution. Thanks to the Jensen inequality, it can be proved that minimising the KL distance between the p and q is equivalent to maximising $\log(\mathbf{w} \,|\, \alpha, \beta)$, although in practice a lower bound of this log-likelihood is maximised.

In our implementation of the following algorithm, we have adopted an empirical Bayes approach, treating β as a parameter. In this context the variational distribution, which is a simplification of the "true" posterior, would be

$$q(\theta, \mathbf{z} \,|\, \gamma, \phi) = q(\theta \,|\, \gamma) \prod_{n=1}^{N} q(z_n \,|\, \phi_n) \,.$$

Thanks to the model assumption that the full conditional distributions of the latent variables belong to the exponential family, from the mean-field theory we can optimise the lower bound of the log-likelihood through a coordinate ascent. This results in a closed form update for the variational parameters:

$$\phi_{ni} \propto \beta_{iw_n} \exp\Big(\mathrm{E}_q[\log(\theta_i) \,|\, \gamma]\Big) \tag{1}$$

$$\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_{ni} \tag{2}$$

Once optimal document level parameters $\gamma^*(\mathbf{w})$ and $\phi^*(\mathbf{w})$ have been found, these can then be used to find the corpus level parameters α and β by maximising the lower bound for the log-likelihood of the observed corpus. This is again maximised, giving the analytical expression for beta, while the alpha is found using an efficient Newton-Raphson explained in [1]. As will be explained later, both β and the ϕ correspond to probabilities so the proportional relationship is dealt with by normalising.

The expressions for α and β both depend on values of γ and ϕ and vice-versa, therefore this maximization is done iteratively and this algorithm is known as the Variational EM. First, in the E-step, the values of γ and ϕ are maximised using the above equations, conditional on the current value of α and β. Then, in the M-step, the values of α and β are updated given the current values of γ and ϕ. Iterating between these E and M steps till convergence result in parameters that maximise the lower bound of the log-likelihood and therefore approximately maximise the likelihood.

2.1.2 Stochastic variational EM

The algorithm described in the previous section requires a full pass over all the corpus in each iteration. To save computational time on large datasets, and to suit a setting where new documents are constantly arriving, online learning for LDA was introduced in [3].

The variational EM is replaced with a stochastic variational EM algorithm. I.e. at each iteration it selects one document randomly and carries out E-step on this (i.e. updates ϕ and γ). Then, in M-step it assumes the whole corpus contains M copies of the sampled document, and updates the parameter accordingly. It can be shown that online LDA corresponds to a stochastic gradient algorithm on the variational log-likelihood. Naturally, an online version can be extended to process a mini-batch at a time.

Here, instead of considering β as a parameter, we take a fuller Bayesian approach and model it as a variable: each row is sampled from an exchangeable Dirichlet distribution with parameter η. Therefore the following family of approximate posterior distributions is considered, introducing another variational parameter λ. Again, using the mean-field theory, we obtain similar closed form update rules.

3 Implementation

Our implementation with various versions of the LDA is available as an R package available in `https://github.com/kasparmartens/oxwaspLDA`. Specifically, we have implemented

the ordinary LDA (i.e. batch LDA), the online LDA (for one document at a time) and the mini-batch version of the latter.

We outline some specific details of our implementation:

- To fit the model, the user is required to provide as input a document $\times$ term matrix, and fix the number of topics.
- To make use of multiple CPUs, we have parallelised the E-step of the algorithm.
- To find the optimal value for parameter α, we have considered two options: all elements α_i being equal (as in the implementation by Blei et al[1]), or treating it as a vector with different elements. Both of these approaches use Newton-Rhapson on the log-scale, as our experiments showed poor performance on the original scale.

4 Results

In order to implement and test the LDA model it is fitted and examined using a data set of NIPS papers ranging from 1988 to 1999. The 'stop' or common words used in every document were removed from the data set in order to help to model extract meaningful topics more easily and words with a frequency less than 20 were also removed. This left a data set of 1500 documents containing 7605 distinct words. This was randomly split into a training set of 1350 papers and a test set of 150 papers. To provide insight into the results, we run LDA on the training data with various number of topics, and display the results for the 10-topic model, i.e. the model parametrised by $\alpha = (\alpha_1, ..., \alpha_{10})$ and $\beta = \beta_{ij}$, $i \in \{1, ..., 10\}$ and $j \in \{1, ..., 7605\}$. We saw that within the first 20-30 iterations of the EM algorithm there increase log-likelihood, whereas later the changes were negligible.

The β contain the information about how likely each word is given each topic. Table 1 shows the top 15 most likely words for each of the 10 topics. It can be seen at this stage that the LDA groups similar words together, for example topic in 4th column appears to be a probability orientated topic while the 8th column appears to have something to do with pictures. The problem which becomes apparent at this stage is that some words appear so often in the data and therefore have a high probability of occurring in many topics, for example "network", appear to be in the top 15 most likely in 5 of the 10 categories.

Table 1: 15 most likely words in each of the 10 topics

network	function	learning	model	neuron	network	recognition	object	network	algorithm
system	learning	algorithm	data	cell	training	network	visual	neural	set
model	network	action	distribution	model	unit	classifier	model	input	function
learning	weight	function	gaussian	input	input	pattern	image	output	problem
neural	error	policy	parameter	synaptic	hidden	feature	motion	circuit	point
control	algorithm	problem	algorithm	pattern	set	image	field	analog	data
input	result	reinforcement	function	activity	error	set	direction	weight	learning
dynamic	set	system	method	spike	output	system	unit	chip	vector
output	neural	optimal	mean	firing	neural	classification	map	system	space
recurrent	number	control	component	network	data	neural	position	current	method
rules	parameter	model	probability	response	weight	training	system	neuron	number
rule	case	step	likelihood	frequency	learning	input	eye	signal	examples
attractor	input	result	density	signal	performance	features	images	function	tree
point	bound	states	mixture	function	layer	character	representation	layer	local
trajectory	training	method	matrix	neural	model	images	view	bit	model

[1] https://www.cs.princeton.edu/ blei/lda-c/index.html

As well as allowing corpus level parameters to be fitted, LDA also provides a framework for doing inference on 'unseen' documents contained in the test set. This requires the values of the document specific parameters γ and ϕ to be computed for the new document. As the corpus level parameters α and β have already been calculated, this just requires an E-step from the LDA algorithm to maximise the likelihood of the new document with respect to γ and ϕ. This procedure was run for the 1st document in the training set, corresponding to the 18th document in the whole corpus. In [1] it's explained that considerable difference between the corpus level α_i and the document level γ_i indicate a strong presence of that topic within the document. For test document 1 topics 1,2 and 6 stand out as making up most of the document with topics 9 and 10 also contributing but not in such a big way. Further evidence of this can be obtained by examining the ϕs for the new document, [1] explain these are approximate values for the probability a word belongs to a specific topic, here the conditioning is the inverse of that for β. Words with ϕ's greater than 0.8, for example, could be taken as coming from just the one topic. In this specific training document there are many words form topics 1 and 2 with probabilities above 0.8, less from 6,9 and 10 and none from any others further demonstrating the spread of topics within this document. Below Figure 2 shows the abstract of the document with words highlighted by topic if there corresponding $\phi \geq 0.8$. Notice words such as "network" and "algorithm" are not color coded as they are likely to appear in multiple topics so can't be assigned to a single one with high probability.

This paper provides a systematic analysis of the recurrent backpropagation (RBP) algorithm, introducing a number of new results. The main limitation of the RBP algorithm is that it assumes the convergence of the network to a stable fixed point in order to backpropagate the error signals. We show by experiment and eigenvalue analysis that this condition can be violated and that chaotic behavior can be avoided. Next we examine the advantages of RBP over the standard backpropagation algorithm. RBP is shown to build stable fixed points corresponding to the input patterns. This makes it an appropriate tool for content addressable memories, one-to-many function learning, and inverse problems.

Figure 2: Abstract from Ottoway, Simard and Ballard (1988) 'Fixed Point Analysis for Recurrent Networks' with coloured words coming from their corresponding topic with probability greater than 0.8

4.1 Document Modeling

To further assess the performance of the model, we compute the perplexity of a test set given the model trained on the training set. Formally, for an unseen set of M documents, the perplexity is defined as

$$\text{perplexity}(\mathcal{D}_{test}) = \exp\left\{-\frac{\sum_{d=1}^{M}\log p(w_d|\alpha,\beta)}{\sum_{d=1}^{M}N_d}\right\} = \exp\left\{\frac{\mathcal{L}(\mathbf{w}_d)}{\sum_{d=1}^{M}N_d}\right\}$$

where N_d is the number of words in document d. This is a decreasing function of the likelihood of test data. As we wish to achieve high likelihood on the test set for the purpose of density estimation, the better model is given by one with lower perplexity.

The exact computation of $\log p(w_d|\alpha,\beta)$ is intractable, we refer to [6] for various sampling methods to approximate this probability. For our purpose, we approximate the likelihood with variational log-likelihood $\mathcal{L}(\alpha_{train},\beta_{train},\phi_{test},\gamma_{test})$ for a fixed number of topics, results shown in Figure 3 for both the variation and stochastic LDA models.

For our LDA model, we considered the hyperparameter α for topic distribution take the same value across all k dimensions v.s. different values, but allowing different values for α seemed to have negligible effect (we see a slight difference for 50 topics). When increasing the number of topics, the perplexity on test documents decreases, indicating a better fit. However, in language modeling a better fit is not necessarily a desirable result for interpretation.

For stochastic LDA model (Figure 3(b)), processing one document at a time the model estimation has not yet converged within the first 1000 iterations. However, in this case we have iterated through every of the 1350 documents less than once, compared to the batch LDA which has seen all the documents for at least 30 times. Comparing to [3] where it is shown that online LDA reaches the

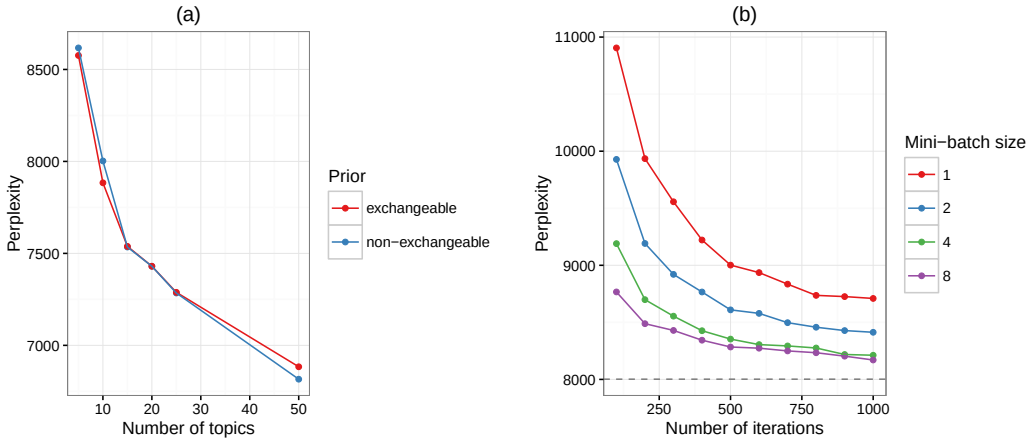

Figure 3: (a) Perplexity of the test documents (y-axis) for various number of topics (x-axis) for two models: all components of α equal (red line, "exchangeable prior") and allowing different values for α (blue line, "non-exchangeable prior"). The models were trained on 90% of the documents, and tested on the remaining 10%. (b) Perplexity for the stochastic LDA models (y-axis) when increasing the number of iterations (x-axis) for different mini-batch sizes (coloured lines). The dashed horizontal line represents the perplexity of the batch LDA model fitted with the same number of topics ($k = 10$).

accuracy of batch LDA on 10^5 documents already in 10^4 iterations (i.e. having seen approximately 10% of the documents), we believe that our example data is not big enough to demonstrate so efficiently the benefits of online learning. As expected, using a mini-batch of size 4 or 8, we reach convergence notably faster.

However, perplexity is useful for evaluating the predictive model and enable selection of model parameters such as the number of topics. But, as pointed out by [2], better preplecity often have less interpretable latent space. We will refer to [2] for quantitative methods of measuring sematic meaning in inferred topics.

Collaborative Filtering

Another way in which this LDA model can be applied is collaborative filtering. The idea of collaborative filtering is to hide one of the words in an unseen document from the LDA, fit the document level parameters using the $(N_d - 1)$ remaining words and then use these to find the likelihood of the removed word.

This requires the derivation of the likelihood of the removed word given the words that have been observed, $p(w|w_{obs})$. [1] give this by looking at the likelihood of the the hidden word as well as the hidden variable z and θ and marginalasing over them.

$$p(w|w_{obs}) = \int \sum_z p(w|z)p(z|\theta)p(\theta|w_{obs})d\theta \tag{3}$$

In the variational inference step, described in the previous sections, $p(\theta|w_{obs})$ is approximated using the variational distribution $q(\theta|\gamma(w_{obs}))$. This, and the fact that the hidden z is per word so only 1 draw from the multinomial is required, simplify this to a linear combination of Dirichlet expectations. Finally note that $p(w|z)$ is simply given by the corpus level $\beta_{z,w}$.

$$p(w|w_{obs}) = \sum_z \beta_{z,w} \frac{\gamma(w_{obs})_z}{\sum_i \gamma(w_{obs})_i} \tag{4}$$

Equation (4) shows us how the LDA goes about assigning how likely a word is to be the hidden word. The Dirichlet expectation ascertains which topics a prevalent within the document and then

the β term assigns high likelihood to words that are likely given the topics present in the document. Unfortunately as we are using the bag of words approximation the model cannot do much better than that.

This collaborative filtering was then applied to the documents in the test set. A word was removed and recorded at random from the data set, the model level parameters, γ and ϕ, were calculated on the remainder of the document and the likelihoods of all of the 7605 being the removed word were calculated. An example is presented below using the 2^{nd} test document: Here the word "hand" was randomly removed from the document. The corpus level βs show that "hand" is the 36th most probable word in topic 7 and the 51st most probable word in topic 1 while the γs show that this test document is predominantly made up of topics 1 and 8 with small contributions form topics 3, 5 and 10. Finally examining the likelihoods show that the removed word "hand" as the 92^{nd} most likely word to be the final word.

Figure 4 shows the 20 most likely words to be the removed words. An intersection between the top 50 most likely words for each topic and the top 20 most likely words for the hidden word shows 12 of the 20 words are very likely given topic 1 and 15 of the 20 are very likely given topic 8. As mentioned before unfortunately there is a lot of overlap between topics so these words are often also very likely in topics not prevalent in the document. The model has done a reasonable job of picking out words that are likely given the topics contributing to the document, but as "hand" is not hugely likely given topics 1 or 8 it is not picked out as being very likely.

model, network, system, learning, neural, object, input, visual, control, unit, field, motion, image, function, dynamic, direction, point, map, position, output

Figure 4: List of the 20 most likely words, in order, to be the removed word from test document 2. The actual removed word "hand" is the 92^{nd} most likely.

Interestingly, increasing the number of topics to 50 allows the model to differentiate between topic on a much finer scale and makes the actual missing word the 23^{rd} most likely given the rest of the document.

5 Discussion

We have discussed LDA and provided implementations for fitting LDA in R. In literature, either variational inference or MCMC-based methods have been used. We have implemented the batch LDA using variational inference, and online LDA with stochastic variational approach.

To use LDA, the user must pre-specify the number of topics, and it is not a straight-forward decision to make. This problem could be solved by using Hierarchical Dirichlet Processes [5].

References

[1] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[2] Jonathan Chang, Jordan Boyd-Graber, Chong Wang, Sean Gerrish, and David M. Blei. Reading tea leaves: How humans interpret topic models. In *Neural Information Processing Systems*, 2009.

[3] Matthew Hoffman, Francis R Bach, and David M Blei. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pages 856–864, 2010.

[4] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[5] Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Hierarchical dirichlet processes. *Journal of the american statistical association*, 101(476), 2006.

[6] Hanna M. Wallach, Ruslan Salakhutdinov, Iain Murray, and David Mimno. Evaluation methods for topic modelling. In *Proceedings of the 26th International Conference on Machine Learning*, 2009.