An Introduction to Latent Dirichlet Allocation

Shubham Shukla shubh77@ufl.edu
Department of Computer Science
University of Florida
Gainesville, FL 32608, USA

Editor:

Abstract

This paper is a tutorial on latent Dirichlet Allocation (LDA), a generative probabilistic model for text corpora such as a collection of documents with discrete data. A document can be modeled as a finite mixture of underlying topics, and each topic in itself can be modeled as an infinite mixture of a set of topic probabilities. These topic probabilities distributed over a document can then be viewed as a representation of a document.

Keywords: Graphical models, Mixture of models, Variational Inference

1. Introduction

The idea behind this process is to come up with hidden descriptions within a collection of documents, which depicts the important statistical relationships between the members of a document, for instance how each document in a collection can be assigned a topic, and similarly given each topic how likely the words in the collection could be generated from it. These relationships can then be put to use in many different tasks such as text classification, sentiment analysis, text generation and many more.

2. Related Work

Before the paper describing latent Dirichlet allocation by (Blei, Ng, Jordan) was published, researchers working in the field of latent Dirichlet allocation had already developed many successful algorithms which aimed to perform document analysis. One successful methodology that was put to great use was to convert each document in a text corpora to a vector of real numbers, which basically just represented the frequency of the words in the corpus. Another methodology introduced by (Salton and Mcgill, 1983) popularized the use of tf-idf scheme, which aimed to represent documents in a fixed length vector instead of arbitray lengths for each document. The idea was to fix a vocabulary with a given set of words or terms and then firstly calculate the term frequency count of each of these words in a given document, which is just a count of number of times a word is contained in a document. After which it needs to be normalized and then compared to the inverse document frequency count, which gives the frequency of each word in the entire corpus (generally performed on a logarithmic scale). This gives us an idea of how each word is more likely related to a particular document in the corpus. The final output of this process is a term-document matrix, whose columns give tf-idf values of a term in each document in the collection. The tf-idf scheme gives us a good insight of certain sets of words that are related to each of the documents, but doesn't gives us much idea about the statistical structure that exist within a document or in entire corpus.

Scientists have proposed methods like *latent semantic indexing* (LSI) as discussed in (Deerwester et al., 1990) that tries to understand the underlying structure within documents in a corpus, by performing singular value decomposition of the term document matrix from tf-idf scheme, to identify a linear subspace that exists in the space of tf-idf features that capture most of the variance in the text corpora. This not only can be useful in compressing a large collection but the author also suggests that LSI can also help in identifying basic lingusitic ideas like polysemy and synonymy, since the derived features from LSI, are just linear combinations of the original tf-idf features. These claims can usually be verified by studying the probabilistic generative model of the text corpora and then using LSI to recover features form this generative model given some data.

First significant approach towards modeling these collection of documents in a probabilistic way was taken by Hoffmann (1999), by introducing a *probabilistic LSI (pLSI)* model. In this approach, each word is a sample from a mixture model, where the mixture components are themselves multinomially distributed such that every word has some probability of belonging to one of these mixture components. Intuitively, its like saying that a word in a document could belong to one of these few given topics. In a generative sense, each word in the corpus is generated from a single mixture component or topic, and different words in the document could be generated from these given different topics. Thus each document can be viewed as a mixture of these mixing components, where each component has a certain proportion and sum of all these mixture proportions sum to one. This reduces a document to be a probability distribution over a fixed set of topics.

This approach does pave the way in terms of probabilistic modeling of text, and allows us to study and analyze the corpus better, but it still has certain limitations which restricts its usage in real-world applications. pLSI gives us the intuition behind how a document can be represented in terms of topics or mixture proportions of these mixture components (a reduced description), where these proportions are just numbers. But, it fails to describe how these numbers or proportions themselves are generated, or what is the idea behind this generative process of mixture proportions. So there is certain probabilistic model with a given document, but no probabilistic model at the document level. Now this can lead to certain problems including overfitting as there is no process to describe what happens at the document level, the parameter size would greatly increase with the increase in data. Another big problem, is there is no clear way described as to how to assign these mixture proportions or probabilities that are outside this training set.

It is important to note that pLSI and LSI methods are all based on "bag-of-words" assumption i.e., the order of words in a document can be ignored and similarly the order of documents in the corpus can be neglected as it doesn't matter too for these implementations. This is defined in terms of probability theory as the notion of "exchangeability". The author in the original paper successfully shows how a classical representation theorem established by de Finetti (1990) can be used to used in accordance with the implementation of LDA, such that it can result to methods that can simplify computing joint distributions and how this theorem can be used to capture significant intra-document statistical structure via mixing distribution.

3. Terminology

In this paper, similar terminology and notations have been used as was in the original paper (Blei, Ng, Jordan). Most of the terms are derived from the language of text collections. This greatly helps to understand the concepts intuitively especially when the idea is represented through a graphical model with latent variables which in itself is an abstract representation of topics.

The following terms are used in the paper:

- A *word* which is a basic unit of data in the corpus. It is an item in a given set of words, represented as a vocabulary {1,...,V} for the entire corpora or for which we need topic modeling. Words are represented as w = [0,...1,...0], in terms of basis unit vectors, where the position v at which the word is set to 1, represents the vth word in the vocabulary.
- A *document* is a sequence of N words denoted by $\mathbf{w} = (w_1, w_2, ..., w_N)$.
- A *corpus* is a collection of M documents denoted by D = ($\mathbf{w}_1$, $\mathbf{w}_2$,..., $\mathbf{w}_M$).

The goal is to compute a probabilistic model of the corpus, that in addition to assigning high probabilities to members of the corpus, also assigns high probability to other documents which are similar in some sense.

4. Latent Dirichlet Allocation

In order to understand the intuition and the process behind LDA, it is a good idea to first see how it can be used to generate data. Latent Dirichlet Allocation is the generative probabilistic model of the text corpora. The idea is to represent a document in terms of random mixtures of latent topics, and how each word in a document can be associated with a topic.

4.1 Generative Model

1. Choose the number of words that you need to generate. N can be characterized by a Poisson Distribution or any other appropriate document length distribution that is desired. It is independent all other random variables that facilitates in generating data, so N is actually an ancillary variable that can be ignored when it comes to randomness for the rest of the derivation. N $\sim$ Poisson(ξ).
2. Choose $\theta \sim$ Dir(α). Dirichlet Distribution takes in a vector parameter α which is of length k, and produces a distribution over vectors of length k θ, where summation of all entries in the vector θ is equal to 1, $\sum^k \theta_i = 1$. So Dirichlet distribution is good way to model probability distributions, and it also has some convenient properties

including that it is conjugate to multinomial distribution. It is a distribution over distributions. The following is the probability density of θ, a topic mixture

$$p(\theta|\alpha) = \frac{\Gamma\left(\sum_{i=1}^{k}\alpha_i\right)}{\prod_{i=1}^{k}\Gamma(\alpha_i)}\theta_1^{\alpha_1-1}\cdots\theta_k^{\alpha_k-1}, \tag{1}$$

Each sample θ is a distinct representation of how the topics are distributed over the documents, or formally the mixture proportions of the mixture components.

3. Now for each of the N words w_n:

   (a) Choose a topic $zn \sim$ Multinomial(θ). It is equivalent to rolling a k-sided die, with each face having a specific probability to show up after rolling the die. The outcome of this sampling results in a selection of a topic z_n.

   (b) Next step is to generate a word from a distribution of words conditioned on the selected topic zn in the previous step and a parameter β, which is again a parameter for a Dirichlet distribution over the words in the vocabulary, such that a sample from this distribution gives, the probability of all words appearing in a document. Now choose a word w_n from p($w_n|z_n, \beta$), a probability distribution over words, conditioned on the latent topic z_n, and the parameter β.

Given the parameter α and β, the joint distribution of a topic mixture θ, a set of N topics z, and a set of N words w, can be given by:

$$p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta) = p(\theta|\alpha)\prod_{n=1}^{N}p(z_n|\theta)p(w_n|z_n, \beta), \tag{2}$$

This joint distribution is also over a topic mixture θ, such that, p($\theta|\alpha$) is probability that for a given parameter α, a distribution is defined over all possible topic mixture θ, and p($z_n|\theta$) signifies a distribution over all the latent topics given a topic mixture θ is observed, and p($w_n|z_n, \beta$) is a distribution over the words in the vocabulary once a particular latent topic is observed. So once, we sample a topic mixture θ, we sample a topic and subsequently a word, for N times, where N is the document length in words and multiply together.

Theoretically we can obtain a marginal distribution of a document, and the corpus from this joint distribution. If we integrate the joint function over θ, we basically obtain, a marginal distribution over a particular document, which is represented as W = $(w_1, w_2, ..., w_N)$, where W is just a specific set of N words, and in such a way each document can be considered unique.

$$p(\mathbf{w}|\alpha, \beta) = \int p(\theta|\alpha)\left(\prod_{n=1}^{N}\sum_{z_n}p(z_n|\theta)p(w_n|z_n, \beta)\right)d\theta. \tag{3}$$

Given a particular topic mixture θ, we can generate a particular W. And to calculate the total probability of W, given a α and β, intuitively we need to sum the probabilities of all possible topic mixtures, generating the same W. And so, we can just compute the marginal distribution of a document simply by integrating over all θ.

To calculate the probability of a corpus, given a set of parameters α and β, we can do the product of marginal probabilities of every single document in the corpus.

$$p(D|\alpha,\beta) = \prod_{d=1}^{M} \int p(\theta_d|\alpha) \left(\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn}|\theta_d) p(w_{dn}|z_{dn},\beta) \right) d\theta_d.$$

4.2 Graphical Model Representation

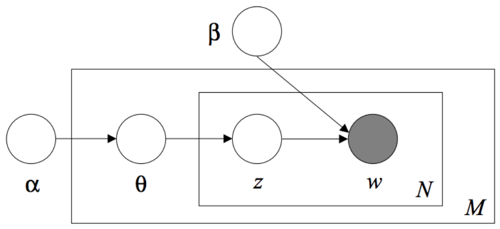


Figure 1: Graphical model representation of LDA. The boxes are "plates" representing replicates. The outer plate represents documents, while the inner plate represents the repeated choice of topics and words within a document.

The LDA model can be represented as a graphical model, as shown in Figure 1. As evident in the Figure, it a hierarchical three level model. The first level starts off by generating the corpus level parameters α and β, which is only sampled once throughout the LDA implementation. The second level or plate in the Figure, deals with generating the document level parameters θ_d which is sampled once for every document in the corpus. And finally, the third level generates word level random variables, z_n, w_n and are sampled for every word in the document. One important takeaway from LDA is how different it is from a Dirichlet-multinomial clustering model, such that how documents can be associated with multiple topics unlike the latter in which a document is restricted to a particular topic. They are also referred to as conditionally independent hierarchical models.

4.3 LDA and Exchangeability

If a joint distribution over a finite set of random variables $\{z_1, z_2, ..., z_n\}$ is invariant to permutation, it is said to be "exchangeable" in terms of probability theory. Similarly, an infinite set of random variables is infinitely exchangeable, if all finite subsequences is exchangeable. De Finetti's representation theorem states that a joint distribution over an infinite exchangeable sequence of random variables, is such that some random parameter is drawn from a distribution and then the random variables in this infinite exchangeable sequence, are independent and identically distributed conditioned on this sampled random parameter. In LDA terms, this theorem is valid, since we are trying to model a "bag-of-words". The words that are generated from these topics are from some distribution conditioned over the topic mixture θ, and also the topics z_n are drawn from a multinomial distribution which has a parameter θ. Also these topics are infinitely exchangeable within documents.

$$p(\mathbf{w},\mathbf{z}) = \int p(\theta) \left(\prod_{n=1}^{N} p(z_n|\theta) p(w_n|z_n) \right) d\theta,$$

5. Inference

Inference in the model is solving for the posterior distribution of the hidden variables, given a document. The distribution function to derive that is as following:

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

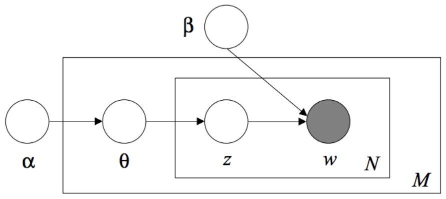
But in order to achieve this we need to normalize the distribution which in turn requires us to compute distribution over the document, and this requires marginalizing over the hidden variables in the model as demonstrated follows:

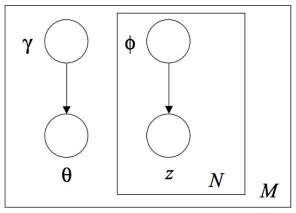
$$p(\mathbf{w} \mid \alpha, \beta) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \int \left(\prod_{i=1}^{k} \theta_i^{\alpha_i - 1} \right) \left(\prod_{n=1}^{N} \sum_{i=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) d\theta,$$

The problem with coming up with this marginal distribution is that it is intractable, as there is a coupling between the parameters θ and β, and so deriving exact inference is not realistic. In this paper, we will however see how an approximate inference can be computed using the variation inference method as described in (Blei, Ng, Jordan).

5.1 Variational Inference

Variation Inference is a convexity-based variational algorithm which makes use of Jensen's inequality to come up with the best possible lower bound on the log-likelihood of the desired distribution. The algorithm first comes up with a set of lower bounds and each is identified by some sort of variational parameter. These parameters are computed by an optimization algorithm that attempts to find the tightest possible lower bound. Now the question arises, how to come up with this family of lower bounds. A good way to approach this is to see what is the exact problem we are facing. Computing the exact inference would always be intractable as long as we have dependency between θ and β in the posterior distribution function. In order to for us to proceed, its a good idea to simply the graphical model that we are working with. If the edges between θ, z, and w are removed in the model, we can break this coupling and it results in a much simplified version of the original LDA graphical model, with similar properties.





This new model however now includes a set of new parameters, called variational parameters as discussed and they characterize the family of distributions over these latent variables. And now we can try to work with this family of distributions which are conditioned on the free variational parameters γ and ϕ where $\phi = (\phi_1, \phi_2, ..., \phi_N$.

$$q(\theta, \mathbf{z} \mid \gamma, \phi) = q(\theta \mid \gamma) \prod_{n=1}^{N} q(z_n \mid \phi_n), \tag{4}$$

Having set up this family of distributions dictated by these variational parameters, the next step is to covert this into an optimization problem in order to derive the variational parameters γ and ϕ, which can find a tight lower bound on the like-likelihood. The exact steps taken to derive this was shown in the paper by (Blei, Ng, Jordan) and the resulting optimization problem is as follows:

$$(\gamma^*, \phi^*) = \arg\min_{(\gamma,\phi)} \mathrm{D}(q(\theta, \mathbf{z} \mid \gamma, \phi) \parallel p(\theta, \mathbf{z} \mid \mathbf{w}, \alpha, \beta)). \tag{5}$$

To obtain these values, we need to minimize the Kullback-Leibler(KL) Divergence between the variational distribution and the true posterior distribution as show in the above formula. This minimization can be achieved via an iterative fixed point method. The update steps of this optimization is also derived in the paper (Blei, Ng, Jordan) and are as follows:

$$\phi_{ni} \propto \beta_{iw_n} \exp\{\mathrm{E}_q[\log(\theta_i) \mid \gamma]\} \tag{6}$$

$$\gamma_i = \alpha_i + \textstyle\sum_{n=1}^{N} \phi_{ni}. \tag{7}$$

Also the expectation of the multinomial update can be computed as follows :

$$\mathrm{E}_q[\log(\theta_i) \mid \gamma] = \Psi(\gamma_i) - \Psi\left(\textstyle\sum_{j=1}^{k} \gamma_j\right), \tag{8}$$

where ψ is the first derivative of the log function. Also the variational distribution is a conditional distribution as the parameters $\gamma*$ and $\phi*$ have been computed for a fixed w. In that sense, it is easy to the see that the variational distribution can actually be written as q($\theta, z|\gamma^*$(w), ϕ^*(w)) and is an approximation of the true posterior distribution p($\theta, z|w, \alpha, \beta$).

The variation inference algorithm for LDA implementation is stated as follows:

```
(1)  initialize $\phi_{ni}^0 := 1/k$ for all $i$ and $n$
(2)  initialize $\gamma_i := \alpha_i + N/k$ for all $i$
(3)  repeat
(4)      for $n = 1$ to $N$
(5)          for $i = 1$ to $k$
(6)              $\phi_{ni}^{t+1} := \beta_{iw_n} \exp(\Psi(\gamma_i^t))$
(7)          normalize $\phi_n^{t+1}$ to sum to 1.
(8)      $\gamma^{t+1} := \alpha + \sum_{n=1}^{N} \phi_n^{t+1}$
(9)  until convergence
```

5.2 Parameter Estimation

The goal now is to come up with a good set of parameters for $\alpha and \beta$, which can represent our corpus D of documents, where D = ($\mathbf{w}_1$, $\mathbf{w}_2$,..., $\mathbf{w}_M$). This is equivalent to maximizing the log-likelihood of data and finding the parameters that maximize it by setting the derivative of $l(\alpha, \beta)$ to be zero:

$$\ell(\alpha, \beta) = \sum_{d=1}^{M} \log p(\mathbf{w}_d \mid \alpha, \beta).$$

As shown previously, the quantity p(w—α, β) can't be computed tractably, so instead the lower bound on the log-likelihood computed using the variational inference algorithm can be maximized with respect to model parameters $\alpha and \beta$. In order for us to compute these approximate estimates, we use the alternating variational EM algorithm where we first maximize the lower bound log value with respect to parameters $\gamma and \phi$ and, then for these fixed parameters, maximize the lower bound with respect to model parameters $\alpha and \beta$. The variation EM algorithm can be stated succinctly as follows:

1. (E-step) : For a fixed document d, find the optimizing values of the variational parameters $\gamma*$ and $\phi*$.

2. (M-step) : Maximize the lower bound on the log-likelihood with respect to the model parameters $\alpha and \beta$. This finds the maximum likelihood estimates with respect to the approximate posterior computed in E-step, for each document in the corpus.

The above steps are repeated until a convergence is reached on the log-likelihood. The M-step for updating the conditional multinomial parameter β is computed as follows:

$$\beta_{ij} \propto \sum_{d=1}^{M} \sum_{n=1}^{N_d} \phi^*_{dni} w^j_{dn}. \tag{9}$$

6. Summary

This paper is an introduction to the probabilistic generative model Latent Dirichlet Allocation. In the full version of this paper, more detailed explanation of various inference methods including collapsed Gibbs Sampling and Markov Chain Monte Carlo Simulation(MCMC) would be discussed along with necessary examples. Also the source code for implementation of LDA in python would also be provided.

Acknowledgments

I would like to acknowledge the support and guidance of Dr. Arunava Banerjee, Department of Computer and Information Science and Engineering, University of Florida, in helping me understand the required theory and background behind probabilistic generative models and unsupervised learning, and the intuitions behind them.

References

David M. Blei, Andrew Y. Ng and Michael I. Jordan. Latent Dirichlet Allocation in Journal of Machine Learning Research 3 (2003) 993-1022