Latent dirichlet allocation for label modelling

Thibaud Ehret & Sammy Khalife

Introduction to document analysis

Eventhough the Latent dirichlet method can be used to a wider class of applications, the first one is document labelling. To see the improvments and contributions of the LDA, it is interesting for us to consider the history of document analysis methods.

- **Term Frequency-Inverse Document Frequency (tf-idf),**
  Salton and McGill, 1983
  This natural and basic method reduces the arbitrary length documents to a fixed length of numbers by counting their apparition in the document.

- **Latent semantic indexing**, *Deerwester et al., 1990*
  This method is similar to principal component analysis. Uses decomposition of the tf-idf matrix. Preserving the most important semantic information

- **Probabilistic Latent Semantic Indexing**, *Hoffmann 1999*
  Here, each word in a document is a sample from a mixture of multinomials. The document is represented as a list of mixing proportions. However, there is no providing of generative model, and the complexity is proportionnal to the size of the corpus.

1 Presentation of the model [Journal of Machine Learning, 2003]

In the latent dirichlet model (LDA), which a graphical representation is given in figure 1, there are different levels of representation of the object concerned (documents in our case). It differs from a simple multinomial model since each document can be associated with multiple topics.

Generation of documents

The parameter θ which belongs to $\mathbb{R}^d$ is sampled once per document from a smooth distribution on the topic simplex, and represents the distribution of the topics within the document. θ is generated through a dirichlet distribution of parameter α. We suppose that each word $\mathbf{w}$ is generated through a multinomial probability conditionned on the topic : $\beta \in \mathbb{R}^{K*V}$, with $\beta_{ij} = p(w^j = 1 | z^i = 1)$, V representing the number of possible words, and K the number of topics.

The fundamental theorical result which justifies this representation is De Finetii's theorem that we will admit. We here suppose that the sequence of words is infinitely exchangeable (i.e that every finite subsequence is exchangeable) which allows us to represent the distribution $p(w, z)$ with a random parameter θ of a multinomial over topics :

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

Example

To understand the model, let us consider the example of 3 words and 4 topics. Then, the (V-1) simplex $\{\theta \in \mathbb{R}^3, \sum_i \theta_i = 1\}$ is included in $\mathbb{R}^2$ and can be seen as a 2D-triangle, in which the parameter θ belongs. Through θ, this simplex represents all possible distributions $p(w|\theta,\beta)$ over the three words.

The surface represented over the triangle is an example of distribution over the simplex :

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

We can interpret each of the vertices of the triangle corresponds to a deterministic distribution that assigns probability one to one of the words, and the midpoint of an edge as a probability which gives 0.5 to two of the words, and the centroid of the triangle is the uniform distribution over all three words. The four points marked with an x are the locations of the multinomial distributions $p(w|z)$ for each of the four topics.

$$p(\mathbf{w}|\alpha,\beta) \;=\; \int p(\theta|\alpha,\beta) p(\mathbf{w}|\alpha,\beta,\theta) d\theta$$

Details concerning the model and equations for inference

In the LDA model, we have $\theta \perp \beta$, and $p(w_i|\alpha,\beta,\theta) \perp p(w_j|\alpha,\beta,\theta)$ for $i \neq j$

$$\begin{aligned} p(\mathbf{w}|\alpha,\beta) &= \int p(\theta|\alpha) \prod_{n=1}^{N} p(\mathbf{w}_n|\alpha,\beta,\theta) d\theta \\ &= \int p(\theta|\alpha) \prod_{n=1}^{N} \sum_{z_n} p(z_n|\alpha,\beta,\theta) p(\mathbf{w}_n|\alpha,\beta,\theta,z_n) d\theta \end{aligned}$$

Moreover $z \perp (\alpha,\beta)$ and $\mathbf{w}_n \perp (\alpha,\theta)$, then

$$p(\mathbf{w}|\alpha,\beta) \;=\; \int p(\theta|\alpha) \prod_{n=1}^{N} \sum_{z_n} p(z_n|\theta) p(\mathbf{w}_n|z_n,\beta) d\theta$$

Then the natural inferential problem that we need to solve in order to use LDA, is to compute :

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

Unfortunately, this expression is intractable due to the coupling between θ and β. This is reason of the specific approximation algorithm presented in the next part.

2 The algorithm in practice

We saw the complexity of the distribution associated to this model, therefore one can not simply calculate the posterior distribution for the inference and the parameter estimation. We will briefly present the algorithm used to calculate the inference and an approximation of the parameters. However, even if the distribution can not be computed, the algorithm is based on the EM principle.

0.1 Inference

The trick to be able to calculate the inference for the LDA model to calculate the variational inference which is a lower bound but easily calculable.

The lower bound is computed using the simplified graphical model presented in figure 3 which does not have the coupling between θ and β and $\mathbf{w}$.

The parameters γ and ϕ for this new model are estimated by minimizing the Kullback-Leibler (KL) divergence which has been shown to be good lower bound for the log-likelihood. Minimizing the KL-divergence gives us the two following equation for γ and ϕ:

$$\phi_{ni} = \frac{1}{Z}\beta_{iw_n} exp\left(E[log(\theta_i|\gamma]\right)$$

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

As we can see the expression of γ and ϕ depend from each other and we have to calculate the fix point for these equation. This is done using an algorithm which compute this fix point by iterating over the solution, the algorithm is presented in Algorithm 1.

```
Data: Initial $\gamma$ and $\phi$
Result: $\gamma$ and $\phi$ verifying the fix point equation
repeat
    for all n and i do
        $\phi_{ni} = \frac{1}{Z}\beta_{iw_n} exp\left(E[log(\theta_i|\gamma]\right)$
        $\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_{ni}$
    end
until Covergence;
```

Algorithm 1: Algorithm computing the fix point for γ and ϕ

These are the parameters that we will use in the next part in order to estimate the parameters.

0.2 Parameter estimation

The goal here is to compute the lower bound of the likelihood. For that, we use a modified version of the EM algorithm.

The expectation part of the algorithm consists in computed the best variational parameters γ and ϕ for the simplified model presented in the previous section. The maximization part consists in optimizing the model parameter α and β with the variational parameter fixed. The optimization on α is done using an algorithm like Newton-Raphson to compute the optimal value whereas for β we have a close form expression for the optimal value:

$$\beta_{ij} = \sum_{d=1}^{M}\sum_{n=1}^{N_d} \phi_{ni} w_{dn}^{j}$$

Just like in the EM algorithm, both step are then iterated until the lower bound of the loglikelihood converges.

1 Experiments

Now that we have seeen how this model can be implemented in practice, we tried it on different type of data. The first type of data was 2 toy examples that we used in order to see how the algorithm would react in practice on data that we created and known to have a specific property so that we can expect some king of result and see if the result given by the code is actually the same than the one we could have been expecting. The second type of example was made using a real dataset called 20Newsgroups and try to analyze the result of the algorithm.

1.1 Toy examples

1.1.1 First toy example

The first toy example is simply a corpus made of 4 documents. Each of these documents containing a different word. We tried to find the repartition of the word into 4 different topics. The result that we were expecting was that the algorithm would put each word into its specific class just as this toy corpus seems to be doing. After running the algorithm we found the following result:

topic 1	topic 2	topic 3	topic 4
word1	word2	word3	word4

Table 1: Table representing the result found by the model on the first toy example

This result was exactly the one we were expecting which validates the mmodel for at least a really simple dataset.

1.1.2 Second toy example

The second toy example was not much complicated than the first one, but we wanted to see how the model would react on a slightly more complicated dataset but still easy enough so one could expect some result for the model. This second toy corpus is still made of 4 documents and overall contains 4 different words. This time two documents contain two words while the other two documents contains the two other words. We then try to find the repartition of the words into 2 topics. The result that we were expecting was that the two words being together in the same document would be in the same topic. After running the algorithm multiple time, we found two possible results that we present below:

topic 1	topic 2
word1	word3
word2	word4

Table 2: Table representing one of the possible outcome of the algorithm on the second toy example

topic 1	topic 2
word1	word3
word2	
word4	

Table 3: Table representing the second possible outcome of the algorithm on the second toy example

The result presented in table 2 being the result that we got for almost all the run but for one run we had the the result presented in table 3. While the result in table 2 is exactly the one that we were expecting, the one presented in table 3 could be considered wrong. This can show one of the limit of the algorithm which only maximizes a lower bound and therefore depending on the initialization we can have a wrong result for the model.

After having trieed on these easy example, we tried the model on a real dataset in order to be able to analyze the result.

1.2 Experiment on a real dataset

The dataset that we used is called 20Newsgroups and is made of approximatively 20000 documents that correspond to forums post from different language that can be classified into 6 major categories that are computers, hobbies, science, politics, religion and classified ads. Our idea was to run the algorithm with 6 topics on this corpus and analyze the classes that would be found by the model and see if these classes would make sense compared to the actual classes of the dataset. We present an extract of the result of the run of the algorithm on the full dataset in the following table:

topic 1	topic 2	topic 3	topic 4	topic 5	topic 6
Atheism	Foundation	Archive	Article	Journalism	Plastic
Atheist	Telephone	Galactic	Edu	Disappointing	Prometheus
Religion	Evolution	Files	Wolkswagon	Elysium	Citroen
Evolution	Books	Information	Autobahns	Madonna	Tractors
Chritians	History	Article	Locomotive	Game	Wolkswagen
Bible	Planets	Writes	Motorway		Bicycling
Biblical	Newsletter	Selection	Sportscar		Competitive
Clerical	National	Create	Municipalities		Window
Church	Science	Integrated	Shaman		Thread
Christianity	National	Goto	Schoolteacher		Course
Christian	Magazine	Xgif			
Catholic	Scientific	Header			
Christ	Complex	Panel			
Theism	Random	Algorithmic			
Believer	Silicon				
	Space				
	Dinosaurs				

Table 4: Table representing an extract of the result found by the model on the 20Newsgroups dataset

We will now try analyze the result given by the algorithm. First of all some of the categories can clearly be seen, such as religion for the topic 1, science for the topic 2 and computers for the topic 3. The last 3 topics are not as easy to analyze because of the lack of constitency in the whole topic. For example, when considering the 6th topic, one can see some words that would most likely appear in the science category such as Plastic, some words that could belong in the computer category like Window or Thread, some words that could belong in the hobby category like Bicycling or Competitive when talking about sport. One could therefore ask himself if there is actually a common structure behind all these words that the algorithm actually found a good topic for these words (we have to keep in mind that the model does only group into topics not necesseraly the one that we can see at first sight), even so it seems unlikely with the knowledge we have on these words and the possible contest one could have use them, or if this problem comes from the approximation, just like we saw in with the seecond toy example.

Conclusion

In this project we studied the Latend Dirichlet Allocation which allows to model documents in a collection such as a corpus of text. Because of the complexity of the model, the parameter are hard to estimate therefore we had to study a second model used to do a variational inference so that we can compute a lower of the likelihood of the initial model. We then used the model on toy examples as well as on the 20Newsgroup dataset to be able to analyze the result of the algorithm.

One improvement which could have been interesting to study after this LDA model is the composite model made from the LDA model as well as the HMM model, which combine the topic modeling as long as the syntax and try to see the differences on the same dataset.

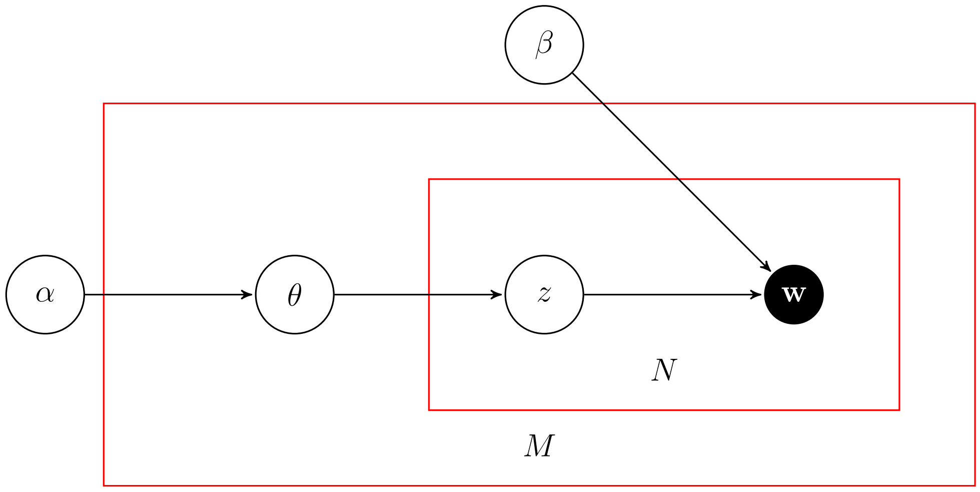

Figure 1: LDA model

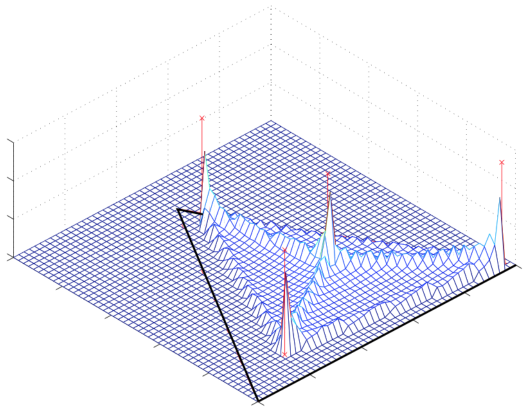

Figure 2: 2D-simplex representation of the example

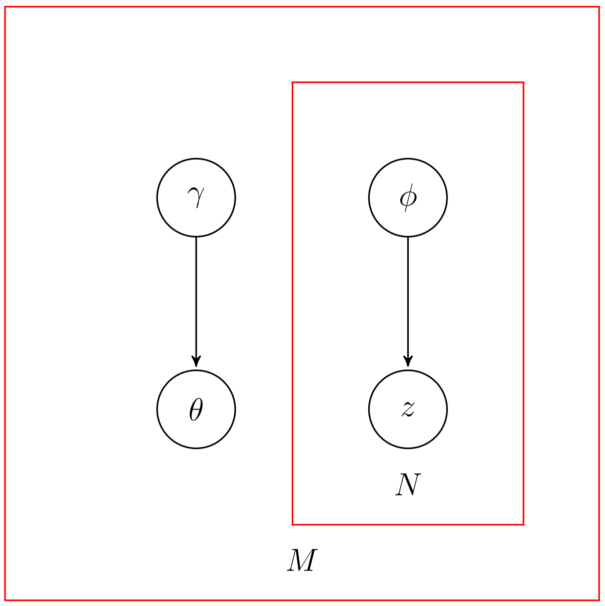

Figure 3: Graphical model used to calculate the lower bound of the log-likelyhood for the LDA graphical model