Topic Modelling using Latent Dirichlet Allocation Algorithm

***Abstract*—Topic models allow the probabilistic modeling of term frequency occurrences in documents. The fitted model can be used to estimate the similarity between documents as well as between a set of specified keywords using an additional layer of latent variables which are referred to as topics. The algorithm used is Latent Dirichlet Allocation (LDA) which utilizes the topicmodels library in R. The aim of this paper is to infer the latent topic structure given the words and document.**

***Index Terms*—LDA, Gibbs sampling, R, text analysis, topic model**

I. Introduction

In machine learning and natural language processing, a topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents. Topic modeling is a frequently used text-mining tool for discovery of hidden semantic structures in a text body.

Topic Modelling is useful in a situation in which you are confronted with a large collection of documents but have no idea what they are about. One of the first things you would want to do is to classify the documents into topics or themes. This would help you figure out if there's anything interesting while also directing you to the relevant subset(s) of the corpus. For small collections, one could do this by simply going through each document but this is clearly infeasible for corpuses containing thousands of documents.

Topic modeling deals with the problem of automatically classifying sets of documents into themes. The algorithm used for this is Latent Dirichlet Allocation (LDA), a technique that facilitates the automatic discovery of themes in a collection of documents.

The basic assumption behind LDA is that each of the documents in a collection consist of a mixture of collection-wide topics. LDA infers the latent topic structure given the words and document by recreating the documents in the corpus by adjusting the relative importance of topics in documents and words in topics iteratively.

II. Algorithms and Models

A. Latent Dirichlet Allocation

A brief explanation of how the algorithm works:

- Go through each document, and randomly assign each word in the document to one of the K topics.
- Go through each word w in d and for each topic t, compute two things: 1) p(topic t | document d) = the

proportion of words in document d that are currently assigned to topic t, and 2) p(word w | topic t) = the proportion of assignments to topic t over all documents that come from this word w. Reassign w a new topic, where you choose topic t with probability p(topic t | document d) * p(word w | topic t) (according to our generative model, this is essentially the probability that topic t generated word w, so it makes sense that we resample the current word's topic with this probability).

- After repeating the previous step a large number of times, we will eventually reach a roughly steady state where our assignments are pretty good. These assignments are used to estimate the topic mixtures of each document (by counting the proportion of words assigned to each topic within that document) and the words associated to each topic (by counting the proportion of words assigned to each topic overall).

The iterative process described in the last point above is implemented using a technique called Gibbs sampling.

The term "Dirichlet" in LDA refers to the fact that topics and words are assumed to follow Dirichlet distributions. Dirichlet distributions provide good approximations to word distributions in documents and, perhaps more important, are computationally convenient.

B. Model

With **plate notation**, the dependencies among the many variables can be captured concisely.

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. M denotes the number of documents, N the number of words in a document. Thus:

α is the parameter of the Dirichlet prior on the per-document topic distributions,

β is the parameter of the Dirichlet prior on the per-topic word distribution,

θ_m is the topic distribution for document m,

φ_k is the word distribution for topic k,

z_{mn} is the topic for the nth word in document m, and

w_{mn} is the specific word.

The generative process is as follows:

Documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words. LDA assumes the following generative process for a corpus D consisting of M documents each of length N_{i}:

1. Choose $\theta_i \sim \mathrm{Dir}(\alpha)$, where $i \in \{1,\ldots,M\}$ and $\mathrm{Dir}(\alpha)$ is the Dirichlet distribution for parameter α
2. Choose $\varphi_k \sim \mathrm{Dir}(\beta)$, where $k \in \{1,\ldots,K\}$
3. For each of the word positions i, j, where $j \in \{1,\ldots,N_i\}$, and $i \in \{1,\ldots,M\}$
   (a) Choose a topic $z_{i,j} \sim \mathrm{Multinomial}(\theta_i)$.
   (b) Choose a word $w_{i,j} \sim \mathrm{Multinomial}(\varphi_{z_{i,j}})$.

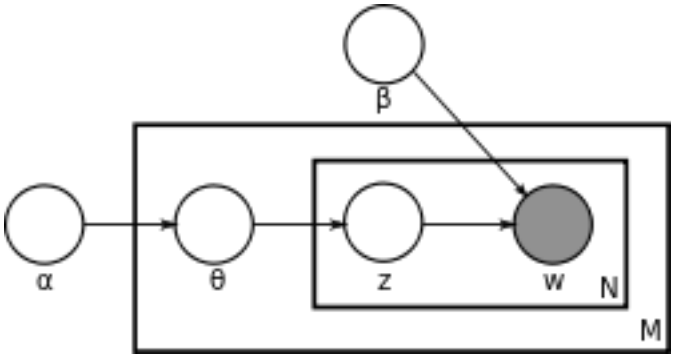

Plate notation representing the LDA model.

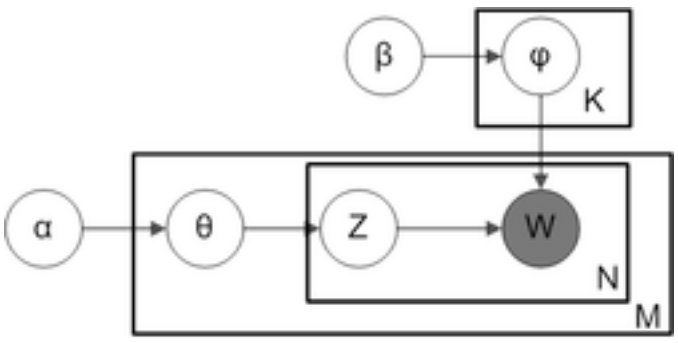

Plate notation for smoothed LDA

Following is the derivation of the equations for collapsed Gibbs sampling:

According to the model, the total probability of the model is:

$$P(\boldsymbol{W},\boldsymbol{Z},\boldsymbol{\theta},\boldsymbol{\varphi};\alpha,\beta)=\prod_{i=1}^{K}P(\varphi_i;\beta)\prod_{j=1}^{M}P(\theta_j;\alpha)\prod_{t=1}^{N}P(Z_{j,t}|\theta_j)P(W_{j,t}|\varphi_{Z_{j,t}}),$$

where the bold-font variables denote the vector version of the variables. First, $\boldsymbol{\varphi}$ and $\boldsymbol{\theta}$ need to be integrated out.

$$\begin{aligned}P(\boldsymbol{Z},\boldsymbol{W};\alpha,\beta)&=\int_{\boldsymbol{\theta}}\int_{\boldsymbol{\varphi}}P(\boldsymbol{W},\boldsymbol{Z},\boldsymbol{\theta},\boldsymbol{\varphi};\alpha,\beta)\,d\boldsymbol{\varphi}\,d\boldsymbol{\theta}\\&=\int_{\boldsymbol{\varphi}}\prod_{i=1}^{K}P(\varphi_i;\beta)\prod_{j=1}^{M}\prod_{t=1}^{N}P(W_{j,t}\mid\varphi_{Z_{j,t}})\,d\boldsymbol{\varphi}\int_{\boldsymbol{\theta}}\prod_{j=1}^{M}P(\theta_j;\alpha)\prod_{t=1}^{N}P(Z_{j,t}\mid\theta_j)\,d\boldsymbol{\theta}.\end{aligned}$$

All the θs are independent to each other and the same to all the φs. So we can treat each θ and each φ separately. We now focus only on the θ part.

$$\int_{\boldsymbol{\theta}}\prod_{j=1}^{M}P(\theta_j;\alpha)\prod_{t=1}^{N}P(Z_{j,t}\mid\theta_j)d\boldsymbol{\theta}=\prod_{j=1}^{M}\int_{\theta_j}P(\theta_j;\alpha)\prod_{t=1}^{N}P(Z_{j,t}\mid\theta_j)\,d\theta_j.$$

We can further focus on only one θ as the following:

$$\int_{\theta_j}P(\theta_j;\alpha)\prod_{t=1}^{N}P(Z_{j,t}\mid\theta_j)\,d\theta_j.$$

Actually, it is the hidden part of the model for the j^{th} document. Now we replace the probabilities in the above equation by the true distributic the explicit equation.

$$\int_{\theta_j}P(\theta_j;\alpha)\prod_{t=1}^{N}P(Z_{j,t}\mid\theta_j)\,d\theta_j=\int_{\theta_j}\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\prod_{i=1}^{K}\theta_{j,i}^{\alpha_i-1}\prod_{t=1}^{N}P(Z_{j,t}\mid\theta_j)\,d\theta_j.$$

Let $n_{j,r}^{i}$ be the number of word tokens in the j^{th} document with the same word symbol (the r^{th} word in the vocabulary) assigned to the i^{th} dimensional. If any of the three dimensions is not limited to a specific value, we use a parenthesized point $(\cdot)$ to denote. For example, $n_{j,(\cdot)}^{i}$ word tokens in the j^{th} document assigned to the i^{th} topic. Thus, the right most part of the above equation can be rewritten as:

$$\prod_{t=1}^{N}P(Z_{j,t}\mid\theta_j)=\prod_{i=1}^{K}\theta_{j,i}^{n_{j,(\cdot)}^{i}}.$$

So the θ_j integration formula can be changed to:

$$\int_{\theta_j}\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\prod_{i=1}^{K}\theta_{j,i}^{\alpha_i-1}\prod_{i=1}^{K}\theta_{j,i}^{n_{j,(\cdot)}^{i}}\,d\theta_j=\int_{\theta_j}\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\prod_{i=1}^{K}\theta_{j,i}^{n_{j,(\cdot)}^{i}+\alpha_i-1}\,d\theta_j.$$

Clearly, the equation inside the integration has the same form as the Dirichlet distribution. According to the Dirichlet distribution,

$$\int_{\theta_j}\frac{\Gamma\left(\sum_{i=1}^{K}n_{j,(\cdot)}^{i}+\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(n_{j,(\cdot)}^{i}+\alpha_i)}\prod_{i=1}^{K}\theta_{j,i}^{n_{j,(\cdot)}^{i}+\alpha_i-1}\,d\theta_j=1.$$

Thus,

$$\begin{aligned}\int_{\theta_j}P(\theta_j;\alpha)\prod_{t=1}^{N}P(Z_{j,t}\mid\theta_j)\,d\theta_j&=\int_{\theta_j}\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\prod_{i=1}^{K}\theta_{j,i}^{n_{j,(\cdot)}^{i}+\alpha_i-1}\,d\theta_j\\&=\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\frac{\prod_{i=1}^{K}\Gamma(n_{j,(\cdot)}^{i}+\alpha_i)}{\Gamma\left(\sum_{i=1}^{K}n_{j,(\cdot)}^{i}+\alpha_i\right)}\int_{\theta_j}\frac{\Gamma\left(\sum_{i=1}^{K}n_{j,(\cdot)}^{i}+\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(n_{j,(\cdot)}^{i}+\alpha_i)}\prod_{i=1}^{K}\theta_{j,i}^{n_{j,(\cdot)}^{i}+\alpha_i-1}\,d\theta_j\\&=\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\frac{\prod_{i=1}^{K}\Gamma(n_{j,(\cdot)}^{i}+\alpha_i)}{\Gamma\left(\sum_{i=1}^{K}n_{j,(\cdot)}^{i}+\alpha_i\right)}.\end{aligned}$$

Now we turn our attentions to the φ part. Actually, the derivation of the φ part is very similar to the θ part. Here we only list the steps of the derivatio

$$\begin{aligned}\int_{\boldsymbol{\varphi}}\prod_{i=1}^{K}P(\varphi_i;\beta)\prod_{j=1}^{M}\prod_{t=1}^{N}P(W_{j,t}\mid\varphi_{Z_{j,t}})\,d\boldsymbol{\varphi}&=\prod_{i=1}^{K}\int_{\varphi_i}P(\varphi_i;\beta)\prod_{j=1}^{M}\prod_{t=1}^{N}P(W_{j,t}\mid\varphi_{Z_{j,t}})\,d\varphi_i\\&=\prod_{i=1}^{K}\int_{\varphi_i}\frac{\Gamma\left(\sum_{r=1}^{V}\beta_r\right)}{\prod_{r=1}^{V}\Gamma(\beta_r)}\prod_{r=1}^{V}\varphi_{i,r}^{\beta_r-1}\prod_{r=1}^{V}\varphi_{i,r}^{n_{(\cdot),r}^{i}}\,d\varphi_i\\&=\prod_{i=1}^{K}\int_{\varphi_i}\frac{\Gamma\left(\sum_{r=1}^{V}\beta_r\right)}{\prod_{r=1}^{V}\Gamma(\beta_r)}\prod_{r=1}^{V}\varphi_{i,r}^{n_{(\cdot),r}^{i}+\beta_r-1}\,d\varphi_i\\&=\prod_{i=1}^{K}\frac{\Gamma\left(\sum_{r=1}^{V}\beta_r\right)}{\prod_{r=1}^{V}\Gamma(\beta_r)}\frac{\prod_{r=1}^{V}\Gamma(n_{(\cdot),r}^{i}+\beta_r)}{\Gamma\left(\sum_{r=1}^{V}n_{(\cdot),r}^{i}+\beta_r\right)}.\end{aligned}$$

For clarity, here we write down the final equation with both $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ integrated out:

$$P(\boldsymbol{Z},\boldsymbol{W};\alpha,\beta)=\prod_{j=1}^{M}\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\frac{\prod_{i=1}^{K}\Gamma(n_{j,(\cdot)}^{i}+\alpha_i)}{\Gamma\left(\sum_{i=1}^{K}n_{j,(\cdot)}^{i}+\alpha_i\right)}\times\prod_{i=1}^{K}\frac{\Gamma\left(\sum_{r=1}^{V}\beta_r\right)}{\prod_{r=1}^{V}\Gamma(\beta_r)}\frac{\prod_{r=1}^{V}\Gamma(n_{(\cdot),r}^{i}+\beta_r)}{\Gamma\left(\sum_{r=1}^{V}n_{(\cdot),r}^{i}+\beta_r\right)},$$

The goal of Gibbs Sampling here is to approximate the distribution of $P(\boldsymbol{Z}\mid\boldsymbol{W};\alpha,\beta)$. Since $P(\boldsymbol{W};\alpha,\beta)$ is invariable for any of Z, Gibbs Sampling equations can be derived from $P(\boldsymbol{Z},\boldsymbol{W};\alpha,\beta)$ directly. The key point is to derive the following conditional probability:

$$P(Z_{(m,n)}\mid\boldsymbol{Z}_{-(m,n)},\boldsymbol{W};\alpha,\beta)=\frac{P(Z_{(m,n)},\boldsymbol{Z}_{-(m,n)},\boldsymbol{W};\alpha,\beta)}{P(\boldsymbol{Z}_{-(m,n)},\boldsymbol{W};\alpha,\beta)},$$

where $Z_{(m,n)}$ denotes the Z hidden variable of the n^{th} word token in the m^{th} document. And further we assume that the word symbol of it is the v^{th} word in the vocabulary. $\boldsymbol{Z}_{-(m,n)}$ denotes all the Zs but $Z_{(m,n)}$. Note that Gibbs Sampling needs only to sample a value for $Z_{(m,n)}$, according to the above probability, we do not need the exact value of

$$P\left(Z_{m,n}\mid\boldsymbol{Z}_{-(m,n)},\boldsymbol{W};\alpha,\beta\right)$$

but the ratios among the probabilities that $Z_{(m,n)}$ can take value. So, the above equation can be simplified as:

$$
\begin{aligned}
&P(Z_{(m,n)} = k \mid \boldsymbol{Z}_{-(m,n)}, \boldsymbol{W}; \alpha, \beta) \\
&\propto P(Z_{(m,n)} = k, \boldsymbol{Z}_{-(m,n)}, \boldsymbol{W}; \alpha, \beta) \\
&= \left(\frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\right)^{M} \prod_{j\neq m}\frac{\prod_{i=1}^{K}\Gamma\left(n^{i}_{j,(\cdot)}+\alpha_i\right)}{\Gamma\left(\sum_{i=1}^{K}n^{i}_{j,(\cdot)}+\alpha_i\right)} \left(\frac{\Gamma\left(\sum_{r=1}^{V}\beta_r\right)}{\prod_{r=1}^{V}\Gamma(\beta_r)}\right)^{K} \prod_{i=1}^{K}\prod_{r\neq v}\Gamma\left(n^{i}_{(\cdot),r}+\beta_r\right) \frac{\prod_{i=1}^{K}\Gamma\left(n^{i}_{m,(\cdot)}+\alpha_i\right)}{\Gamma\left(\sum_{i=1}^{K}n^{i}_{m,(\cdot)}+\alpha_i\right)} \prod_{i=1}^{K}\frac{\Gamma\left(n^{i}_{(\cdot),v}+\beta_v\right)}{\Gamma\left(\sum_{r=1}^{V}n^{i}_{(\cdot),r}+\beta_r\right)} \\
&\propto \frac{\prod_{i=1}^{K}\Gamma\left(n^{i}_{m,(\cdot)}+\alpha_i\right)}{\Gamma\left(\sum_{i=1}^{K}n^{i}_{m,(\cdot)}+\alpha_i\right)} \prod_{i=1}^{K}\frac{\Gamma\left(n^{i}_{(\cdot),v}+\beta_v\right)}{\Gamma\left(\sum_{r=1}^{V}n^{i}_{(\cdot),r}+\beta_r\right)} \\
&\propto \prod_{i=1}^{K}\Gamma\left(n^{i}_{m,(\cdot)}+\alpha_i\right) \prod_{i=1}^{K}\frac{\Gamma\left(n^{i}_{(\cdot),v}+\beta_v\right)}{\Gamma\left(\sum_{r=1}^{V}n^{i}_{(\cdot),r}+\beta_r\right)}.
\end{aligned}
$$

III. Results and Discussion

We will use the **topicmodels** package written by Bettina Gruen and Kurt Hornik. Specifically, we will use the LDA function with the Gibbs sampling method.

For the most part, we will use the default parameter values supplied by the LDA function, custom setting only the parameters that are required by the Gibbs sampling algorithm. Gibbs sampling works by performing a random walk in such a way that reflects the characteristics of a desired distribution. Because the starting point of the walk is chosen at random, it is necessary to discard the first few steps of the walk (as these do not correctly reflect the properties of distribution). This is referred to as the burn-in period. We set the burn-in parameter to 4000. Following the burn-in period, we perform 2000 iterations, taking every 500th iteration for further use.

The reason we do this is to avoid correlations between samples. We use 5 different starting points (nstart=5) – that is, five independent runs. Each starting point requires a seed integer (this also ensures reproducibility), so we have provided 5 random integers in my seed list.

Finally, we set best to TRUE, which instructs the algorithm to return results of the run with the highest posterior probability. There is an important parameter that must be specified upfront: *k*, the number of topics that the algorithm should use to classify documents.

The LDA algorithm returns an object that contains a lot of information. Of particular interest to us are the document to topic assignments, the top terms in each topic and the probabilities associated with each of those terms.

The table below lists the document to (primary) topic assignments:

	V1
d1.txt	4
d10.txt	2
d11.txt	1
d12.txt	1
d13.txt	3
d14.txt	5
d15.txt	3
d16.txt	2
d17.txt	3
d18.txt	5
d19.txt	5
d2.txt	1
d20.txt	1
d21.txt	5
d22.txt	5
d23.txt	2
d24.txt	3
d25.txt	5
d26.txt	3
d27.txt	3
d28.txt	2
d29.txt	5

d3.txt	5
d30.txt	4
d31.txt	3
d32.txt	2
d4.txt	2
d6.txt	2
d7.txt	3
d8.txt	1
d9.txt	3

The table below lists the top 6 terms in topics 1 through 5.

A	B	C	D	E	F
	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
1	neural	analysis	model	artificial	learning
2	network	kernel	theory	intelligence	machine
3	algorithm	classification	decision	system	data
4	networks	vector	markov	computer	mining
5	function	classifier	clustering	and	software
6	evolutionary	for	latent	cognitive	language

From a quick perusal of the two tables it appears that the algorithm has done a pretty decent job.

For example, topic 4 is about artificial intelligence, and the documents assigned to it are on topic.

The table below lists the topic probabilities by document:

	V1	V2	V3	V4	V5
1	0.1842105	0.1578947	0.1666667	**0.254386**	0.2368421
2	0.2	**0.325**	0.15	0.125	0.2
3	**0.5448718**	0.1282051	0.1025641	0.1089744	0.1153846
4	**0.3768116**	0.1932367	0.1304348	0.1304348	0.1690821
5	0.1964286	0.1785714	**0.25**	0.1964286	0.1785714
6	0.1935484	0.1774194	0.1935484	0.1935484	**0.2419355**
7	0.201087	0.0869565	**0.5054348**	0.0978261	0.1086957
8	0.1992754	**0.4202899**	0.1449275	0.0869565	0.1485507
9	0.1473684	0.2	**0.3368421**	0.1368421	0.1789474
10	0.1287129	0.1584158	0.1980198	0.1188119	**0.3960396**
11	0.1440397	0.2036424	0.1456954	0.0993377	**0.4072848**
12	**0.6422764**	0.0785908	0.0677507	0.1111111	0.100271
13	**0.3417722**	0.1772152	0.1518987	0.1518987	0.1772152
14	0.1875	0.15625	0.171875	0.21875	**0.265625**
15	0.1184834	0.1753555	0.1232227	0.0805687	**0.5023697**
16	0.2234043	**0.3829787**	0.1170213	0.1489362	0.1276596
17	0.0991736	0.214876	**0.4380165**	0.1157025	0.1322314
18	0.171875	0.15625	0.1875	0.234375	**0.25**
19	0.1521739	0.1376812	**0.4275362**	0.1014493	0.1811594
20	0.1803279	0.1967213	**0.2295082**	0.1639344	**0.2295082**
21	0.1818182	**0.3116883**	0.1428571	0.1298701	0.2337662
22	0.0932642	0.2331606	0.0984456	0.119171	**0.4559585**
23	**0.2077922**	0.1558442	0.1818182	0.1558442	0.2987013
24	0.0746269	0.0729685	0.0630182	**0.6733002**	0.1160862
25	0.135	0.08	**0.54**	0.14	0.105
26	0.0927835	**0.5618557**	0.1134021	0.1134021	0.1185567
27	0.1688312	**0.512987**	0.1103896	0.0844156	0.1233766
28	0.1833333	**0.2166667**	0.1833333	0.2	**0.2166667**
29	0.1785714	0.2142857	**0.2321429**	0.1785714	0.1964286
30	0.2666667	**0.2533333**	0.1733333	0.1733333	0.1333333
31	0.1927711	0.1566265	**0.3012048**	0.1927711	0.1566265

In the table, the highest probability in each row is in **bold**. In general, if a document has multiple topics with comparable probabilities, it simply means that the document speaks to all those topics in proportions indicated by the probabilities.

IV. Conclusion and Future Scope

Topic modelling provides a quick and convenient way to perform unsupervised classification of a corpus of documents. As always, though, one needs to examine the results carefully to check that they make sense. The LDA model is highly modular and can therefore be easily extended. Variations on LDA have been used to automatically put natural images into categories, such as "bedroom" or "forest", by

treating an image as a document, and small patches of the image as words; one of the variations is called Spatial Latent Dirichlet Allocation. Recently, LDA has been also applied to bioinformatics context. LDA is often prone to creating topics that are not easily interpretable by humans. Recent extensions include generating topics from a user-defined seed set of keywords.

V. References

[1] Blei, David M.; Ng, Andrew Y.; Jordan, Michael I (January 2003). Lafferty, John, ed. "Latent Dirichlet Allocation". Journal of Machine Learning Research. 3 (4–5): pp. 993–1022. doi:10.1162/jmlr.2003.3.4-5.993.

[2] Pritchard, J. K.; Stephens, M.; Donnelly, P. (June 2000). "Inference of population structure using multilocus genotype data.". Genetics. 155 (2): pp. 945–959. ISSN 0016-6731.

[3] Google Scholar". scholar.google.ca. Retrieved 2016-02-10.

[4] Google Scholar". scholar.google.ca. Retrieved 2016-02-10.

[5] http://download.springer.com/static/pdf/248/chp%253A10.1007%252F978-3-319-03164-4_13.pdf?originUrl=http%3A%2F%2Flink.springer.com%2Fchapter%2F10.1007%2F978-3-319-03164-4_13&token2=exp=1481401888~acl=%2Fstatic%2Fpdf%2F248%2Fchp%25253A10.1007%25252F978-3-319-03164-4_13.pdf%3ForiginUrl%3Dhttp%253A%252F%252Flink.springer.com%252Fchapter%252F10.1007%252F978-3-319-03164-4_13*~hmac=d700b4352267bece6970865eb58fdfc6b16c053ad35df90b815da5c9d5b0dc6d

[6] https://eight2late.wordpress.com/2015/05/27/a-gentle-introduction-to-text-mining-using-r/

[7] https://cran.r-project.org/web/packages/topicmodels/vignettes/topicmodels.pdf

[8] https://www.cs.princeton.edu/~blei/topicmodeling.html