# Topic Modelling - A Tutorial

Jeevesh Juneja

May 2021

## 1 Introduction

The problem of topic modelling, as its name suggests, is about finding the central topics in a given document. Why not call it topic extraction, then? Because, in addition to figuring out what topics compose a particular document, we also need to figure out *what these topics are.*

Now, an obvious approach to defining what topics are is to define them as words, for example, topics can be ”pets”, ”dogs”, ”sun”, etc. The presence of a topic in a document then simply corresponds to presence of the corresponding word in the document. What we have done just now, is that we defined a model, describing what a topic is, and then used it to carry out our subsequent task of topic extraction from the document.

When we call the problem ”topic modelling” instead of ”topic extraction”, the emphasis is on the fact that the central problem that we need to solve is how to define a *statistical* model for what a topic is and its relation with the document. Once that model is defined, the problem of using that model to figure out what topics are present in the document is pretty straightforward and can be solved using existing statistical methods. The importance of statistical modelling is highlighted by the following quotes:

> ”The majority of the problems in statistical inference can be considered to be problems related to statistical modeling”. [6, p. 75]
> ”How [the] translation from subject-matter problem to statistical model is done is often the most critical part of an analysis”. [4, p. 197]

## 2 The basic model

Any statistical model is the result of a number of assumptions, some of these assumptions are more basic than other. Here we list some basic assumptions that nearly every model assumes:

1. We assume that we have a finite sized vocabulary, $V$ of words. That all the words in the documents belong to this vocabulary.

2. Any document can be seen as either a sequence of words, $\mathbf{s} \in V^L$, where $L$ is the length of the document; or it can be seen as a bag of words[1], $\mathbf{b} \in \mathbb{Z}^V_{\geq 0}$, where $b_i$ is the frequency of the $i^{th}$ word in the current document.

3. A "topic" is assumed to be a probability distribution over words in the vocabulary. The words in a document are assumed to be generated from a distribution that combines together the distributions corresponding to each of the topics in the document, **in some way**.

Each of the following models, adds to the above set of assumptions. Or it may build upon these assumptions, for example, by specifying what exactly is the "**some way**" of 3. New research often includes coming up with these new assumptions which leads to new models, or often how to infer in better ways from any of the previously constructed models.

# 3 Evaluation

To figure out how to evaluate the topic models, we must ask ourselves, what are we going to use it for? Topic models can be used for finding out what topics compose each document in a given set of documents(*a.k.a.* corpus), or to identify topics composing a new document, or to generate new documents focused on particular topic(s).

## 3.1 A First Approach: Log Likelihood

The most widespread method for automatically evaluating the strength of a topic model was evaluating the log likelihood of some held-out documents under the model learnt from the training corpus. The higher the log-likelihood of the held-out documents were, the stronger the model was. But, this approach is only a measure of the capacity of the model that is sometimes correlated to the quality of the topics identified by the model. But often times, it is not correlated, or even anti-correlated with the human interpret-ability of the topics and the human notion of topics being present in a document as can be seen in Figure 1 from [3] .

## 3.2 Topic Coherence and Topic Assignment

After observing the above effect, [3] proposed two human evaluation methods to determine the human interpret-ability of the topics identified by the model(topic coherence) and how well does the model's decomposition of a document into topics agrees with the human associations of the topics to that document(topic assignment).

[1] Often times, when considering the document as bag of words, we remove words that are not related to any topics or infeasible to consider as individual topics, for example, stop words or words that occur in less than 5 documents

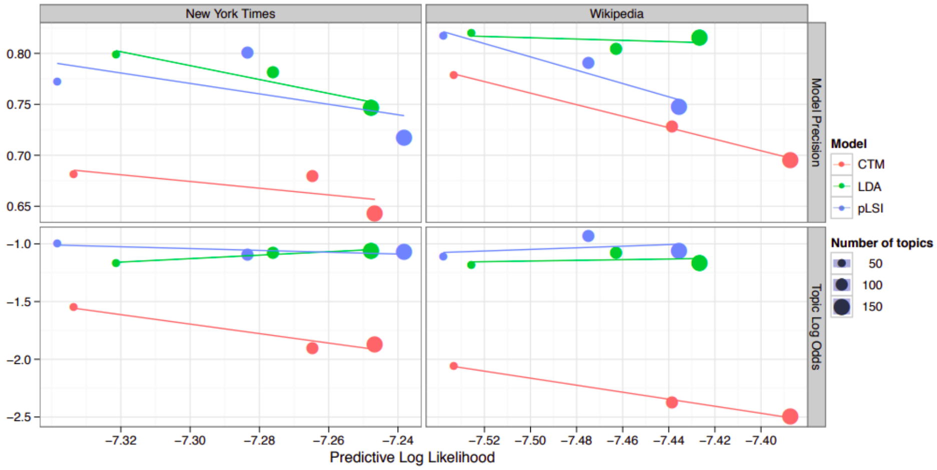

Figure 1: On X-axis the log-likelihood of unseen documents under the learnt model. The downward sloping lines means that as log-likelihood increases(with increasing model capacity), the actual human evaluation of the topics decreases.

For measuring Topic Coherence, we adopt the following way: For each topic learned by the model, we pick 5 most probable words add an additional random word sampled from the pool of words having low probability under the current topic, but high probability under some other topic and shuffle all those. A human is asked to identify the "intruder" word. The precision of the model is defined as the number of humans that correctly identify the intruder word.

For measuring Topic Assignment, we adopt the following way: For each document, we present its title, a snippet from its beginning, the three most probable topics predicted by our model mixed with a random topic sampled from the topics having low probability for this document, as predicted by the model. [2] The four topics are presented along with snippet and title of document, to a human who is asked to identify the topic that doesn't correspond to the document's snippet and title(the "intruder" topic). The "topic log odds" for the model is defined as the average of the log ratio of probability mass assigned(by the topic model) to the true intruder to the probability mass assigned to the intruder selected by the human.

## 3.3 Automating the Identification of Intruder Words

The only part of the above evaluation procedures where human was required was identifying the intruder words. So, in an attempt to design an automated evaluation method, [7] develops a model to identify the intruder word/topic from a given list. Each word of a sample(having one intruder word in $N$ words)

[2] Each topic is represented by 8 of the most probable words under that topic.

is converted to a feature vector by computing the following metrics:

$$\text{PMI}(w_i) = \sum_{j}^{N-1} log \frac{P(w_i, w_j)}{P(w_i)P(w_j)} \tag{1}$$

$$\text{CP1}(w_i) = \sum_{j}^{N-1} \frac{P(w_i, w_j)}{P(w_j)} = \sum_{j}^{N-1} P(w_i|w_j) \tag{2}$$

$$\text{CP2}(w_i) = \sum_{j}^{N-1} \frac{P(w_i, w_j)}{P(w_i)} = \sum_{j}^{N-1} P(w_j|w_i) \tag{3}$$

Here, PMI stands for point-wise mutual information, and the other two equations give the conditional probabilities. All these measure the association of the $i^{th}$ word with all the other words in the sample. $P(w_i, w_j)$ is the probability of getting the $i^{th}$ and $j^{th}$ word together in some context window. An external corpus with a particular context window size is used to learn these conditional and joint probabilities. $P(w_i|w_j)$ is the probability of finding $w_i$ in a context window, given it already has $w_j$, and similarly the other conditional probability is defined.

The above three metrics are combined in a vector and used as the representation of word, $w_i$. Then, an SVM-Rank[3] uses the representations of all the words to predict the intruder word. (Since we ourselves added the intruder word we already know the correct intruder word and can use it as signal for training the SVM). Once trained, the predictions of the SVM-Rank are shown to be highly correlated with that of the human.

For identifying intruder topics, we can make representation of a topic as the concatenation of all the feature vectors corresponding to all the words representing the topic. Also note that the summation in the above equations is done over all the words representing all the topics. The rest of the procedure remains same.

# 4 Modelling Approaches

## 4.1 Latent Semantic Analysis: Simple SVD

The simplest method that comes to mind to model topics and extract them from documents is by doing SVD of the matrix having the **b** vectors corresponding to all the documents. The top-$K$ eigenvectors[4] from the SVD, when normalized to sum to 1, can be used as the estimates for top-$K$ topics present in document. The components of a document vector, **b** along these $K$-topics, give the distribution over topics for that document, when normalized. These $K$-topics are

[3] Basically an SVM that operates on pairs of word representations and predicts which one is more likely to be the representation of the intruder

[4] from the matrix in the decomposition having size $|\mathbf{b}| \times C$, where $C$ is the number of documents in the corpus and $|\mathbf{b}|$ is the number of elements in $|\mathbf{b}|$

top/optimal in the sense that among all possible sets of $K$-topics, this particular set result in the maximum joint probability of all document vectors in the corpus, that can be obtained by defining a multinomial distribution over the set of $K$-topics. In literature, this method is known as Latent Semantic Analysis [5], since we are trying to find latent semantic information in the bag-of-words of the documents.

## 4.2 Latent Dirichlet Allocation

### 4.2.1 Dirichlet Distribution

Consider all possible multinomial distribution over two variables $[x_1, x_2]$. They consist of the segment of line $x_1 + x_2 = 1$ where $x_1, x_2 \geq 0$. This is a straight line. Similarly, for three variables, the set of all possible multinomial distributions is $\{(x_1, x_2, x_3) : x_1 + x_2 + x_3 = 1; x_1, x_2, x_3 \geq 0\}$, which is a triangle in 3-D space. Similarly, for $n$-variables, the set of all possible multinomial distributions is a $(n-1)$-dimensional simplex. The support of the $K$-dimensional Dirichlet distribution is the $(K-1)$-dimensional simplex, and the distribution has the following pdf:

$$f\left(x_1, \ldots, x_K; \alpha_1, \ldots, \alpha_K\right) = \frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{i=1}^{K} x_i^{\alpha_i - 1} \tag{4}$$

where $\mathrm{B}(\boldsymbol{\alpha})$ is a normalizing factor. It is the multi-dimensional beta function which can be expressed in terms of the gamma function as :

$$\mathrm{B}(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}, \qquad \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K) \tag{5}$$

As can be seen from above equation, a higher $\alpha_i$ would imply more probability density is concentrated over higher values of $x_i$ than is there on lower values of $x_i$. This is visualised in Figure 2.

### 4.2.2 The Model

As noted in 1, maximizing likelihood as done by SVD, doesn't necessarily mean that the topics the document-wise topic distribution are human interpretable. Neither does it mean that the model will generalise well to new documents. [2] attempted to solve this problem by providing a prior on the possible per-topic multinomial distribution over vocabulary, that gave higher probability to multinomial distributions that were sparse, i.e., to distributions that will allow each topic to correspond to a few words. A similar prior is put on the per-document topic distributions, to encourage each document to correspond to a small number of topics. This prior that encourages sparsity, is a Dirichlet distribution(over the set of all multi-nomial distributions over vocabulary/topics) with a $\boldsymbol{\alpha}$, all of whose entries are less than 1, which ensures all the exponents in equation (4) are negative, which concentrates the density near the vertices of the simplex, among other sparse points.

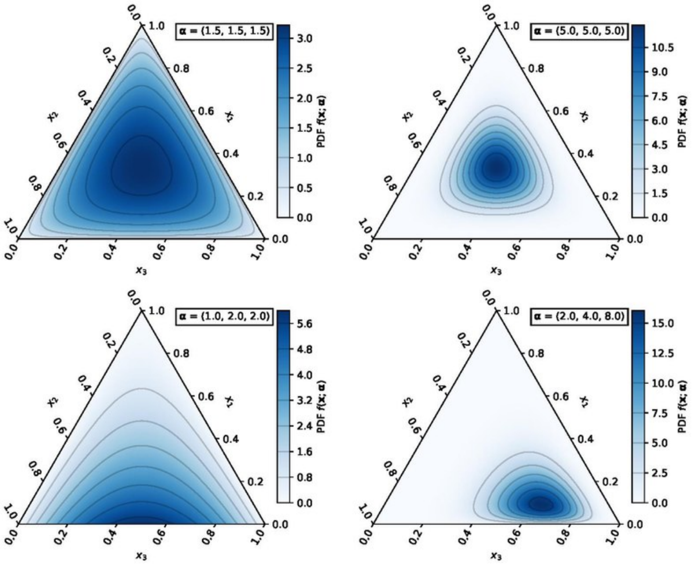

Figure 2: The Dirichlet distribution for various values of $\boldsymbol{\alpha}$. Notice how the density concentrates over higher values of $x_3$ in the bottom right plot, since $\alpha_3$ is the highest.

The generative process the model assumes, is as follows: two hyperparameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are selected. We sample $M$ multi-nomial distributions $\{\theta_i\}_{i=1}^{M}$ from Dir($\boldsymbol{\alpha}$) each one corresponding to a document, and $K$ multi-nomial distributions $\{\phi_i\}_{i=1}^{K}$ are sampled from Dir($\boldsymbol{\beta}$), one corresponding to each topic. Each word in each document is then generated by first selecting a topic, $z$, from the multi-nomial distribution selected for that document, followed by sampling a word from the multi-nomial distribution(corresponding to topic $z$) over the vocabulary. The inference problem then, is to determine the latent multi-nomial distributions $\{\theta_i\}_{i=1}^{M}$ and $\{\phi_i\}_{i=1}^{K}$ that maximize the probability of the given corpus of documents.

## 4.3 VAE Based Topic Models

VAE based topic models try to learn latent representations of documents by auto-encoding their bag-of-words vectors, **b**. But how to use this latent representation to get the per-topic word distributions and the distribution over topics for the words? [8] solve this problem by changing the architecture of the decoder to a simple linear transformation with bias, followed by a soft-max, i.e., decoder($z$) = softmax($T^T z + c$). When defined this way, the matrix $T \in \mathbb{R}^{K \times V}$ can be considered as defining relative importance of each of the $V$ words in the vocabulary for each of the $K$ topics, and the vector $z \in \mathbb{R}^K$ can be considered as defining relative importance of each of the $K$ topics in the current document.

The likelihood of the document is predicted under the distribution over vocabulary predicted by the decoder, and the loss is back-propagated to maximize the likelihood.

The main advantage and use of VAE based topic models is that they fit well with other deep neural networks, mainly because of the simple back-propagation based optimization they use, rather than carrying out Bayesian inference like other models. This allows for efficient joint/multi-task training along with other DNNs (for e.g. [9]).

# 5 Inference Methods

The problem of inference is to infer the distribution $p(h|v)$ over the latent variables, using the prior $p(h)$ and given the observed variables $v$, using Bayes Theorem:

$$p(h|v) = \frac{p(v|h)p(h)}{Z} \tag{6}$$

where $Z$ is the marginalised[5] probability of the observed variables, also known as, *partition function.*

$p(v|h)$ is usually easy to find and $p(h)$ is already given, but the calculation of $Z$ may involve intractable[6] integrals. Inference methods usually fall into two broad categories: one category consists of techniques which try to find an approximation to the partition function(Monte Carlo Methods).

The other category consists of techniques that view inference as finding a distribution $q(h)$ similar to $p(h|v)$ from some family of distributions, by optimizing some distance measure between the distributions, for e.g. $D_{KL}(q(h)||p(h|v)$ (Variational Methods). We have the freedom to choose any of these inference techniques for any of the models described in section 4.

## 5.1 Monte Carlo Methods

The basic principle is to represent integrals as expected values and then estimate those expected values by averaging over samples from the distribution over which expected value is taken. Note here that **1)** increasing the number of samples increases the quality of the estimate; and **2)** the number of samples required for similar quality estimate, increases exponentially with the dimension of the integral, and the integral becomes intractable again. However, if a $d$-dimensional can be factored into $d$, 1-dimensional integrals, the number of samples required increases linearly, and the integral remains tractable. If the integral is of the form:

$$s = \int p(x)f(x) = E_{x \sim p}[f(x)] \tag{7}$$

[5] marginalised over the latent variables

[6] i.e., either we can't find analytic expression for the integral, or it takes exponential time to calculate a Monte Carlo approximation to it

then, the "Monte Carlo" estimate is of the form:

$$\hat{s} = \frac{1}{n}\sum_{i=1}^{n} f(x_i); \qquad \mathrm{Var}[\hat{s}] = \frac{\mathrm{Var}[f(x)]}{n} \tag{8}$$

where Var[.] denoting the variance of the estimate, measures the quality of estimate.

### 5.1.1 Importance Sampling

Sometimes, it may be difficult to sample from the distribution $p(x)$, in which case the method of importance sampling allows us to sample from a distribution of our choice $q(x)$, and still estimate the same quantity:

$$\int p(x)f(x) = \int \frac{q(x)p(x)f(x)}{q(x)} = E_{x \sim q}\left[\frac{p(x)f(x)}{q(x)}\right] \tag{9}$$

. The estimate, and its variance in this case are :

$$\hat{s} = \frac{1}{n}\sum_{i=1}^{n} \frac{p(x_i)f(x_i)}{q(x_i)}; x_i \sim q(x) \qquad \mathrm{Var}[\hat{s}] = \frac{1}{n}\mathrm{Var}\left[\frac{p(x)f(x)}{q(x)}\right] \tag{10}$$

A good $q(x)$, i.e., a $q(x)$ that results in a good estimate with fewer number of samples, is one that is proportional in some way to $p(x)f(x)$.

Sometimes, we can find only such a $q(x)$ in its un-normalized form, or sometimes, we can calculate $p(x)$ in its un-normalized form only(for e.g., when using complex energy-based undirected graph models), in that case we can use the following biased estimate of the expectation:

$$\hat{s}_{\mathrm{BIS}} = \frac{\frac{1}{n}\sum_{i=1}^{n} \frac{p(x_i)f(x_i)}{q(x_i)}}{\frac{1}{n}\sum_{i=1}^{n} \frac{p(x_i)}{q(x_i)}} = \frac{\sum_{i=1}^{n} \frac{p(x_i)f(x_i)}{\tilde{q}(x_i)}}{\sum_{i=1}^{n} \frac{p(x_i)}{\tilde{q}(x_i)}} = \frac{\sum_{i=1}^{n} \frac{\tilde{p}(x_i)f(x_i)}{\tilde{q}(x_i)}}{\sum_{i=1}^{n} \frac{\tilde{p}(x_i)}{\tilde{q}(x_i)}}; x_i \sim q(x) \tag{11}$$

where $\tilde{p}$ and $\tilde{q}$ are the un-normalized forms of the corresponding distributions. The expected value of the numerator, as $n \rightarrow \infty$, tends to the value of the integral and the expected value of the denominator tends to 1. When $n$ is finite, the expected value of the expression does not equal the value of the integral unlike previous estimates.

### 5.1.2 Markov Chain Monte Carlo

This method is used when we can neither sample from $p(x)$, nor find a good importance sampling distribution $q(x)$ for our Monte Carlo estimate. This method uses a Markov chain to draw samples from a distribution. The space over which probability distribution is divided into "states" , $S$. Now imagine a transition function $T(x'|x)$, giving the probability of transition to state $x'$ from state $x$. We start at a random sample from all the states $x_0$, then we sample a state from $x' \sim T(x|x_0)$, followed by sampling $x'' \sim T(x|x')$ and so on. If we try to find the probability of us ending in state $x$ after $n$ steps, for each state $x$; we will get a probability distribution over all states for each $n$. As $n \to \infty$ this probability distribution doesn't change much with $n$[7] and is called the *stationary distribution* for the Markov chain with the given transition function $T(x'|x)$. So, sampling one of our states at random, and then running our Markov chain for a large number of steps is equivalent to sampling from this stationary distribution. The stationary distribution $q$, must satisfy the constraint that the probability of landing in state $x'$ after a transition (RHS of equation 12), must equal the probability of state $x'$ under $q$.

$$q(x') = E_{x \sim q}[T(x'|x)], \qquad \forall x' \in S \tag{12}$$

Usually it is a difficult to find the transition function whose stationary distribution is the one we want to sample from, but there is a nice transition function in case of undirected probabilistic graphical models(UPGM) that results in the stationary distribution being the one that the UPGM induces. This is the *Gibbs Sampling Technique*. We initialize all variable in the UPGM, we random values, then we **repeatedly** update all variables, one-by-one. Each update of a variable consists of sampling that variable, conditioned on the values of all the other variables' values. This makes a Markov chain, whose stationary distribution is the one that is induced by the UPGM.

## 5.2 Variational Methods

In the previous section, we were trying to approximate the partition function for estimating our posterior. Variational methods, try to approximate the entire posterior instead of just the partition function, and then use this approximate posterior to learn the parameters of the model further. We choose a family of distributions and try to find the member distribution of it that is closest to the actual posterior. Mathematically, we can say that we are trying to minimize $D_{KL}(q(h)||p(h|v))$, or equivalently, maximizing:

$$\text{ELBO}(v, \theta, q) = \text{log}p(v;\theta) - D_{KL}(q(h|v)||p(h|v;\theta)) \tag{13}$$

where $\theta$ are the parameters of the learnable model, $v$ are the observations and $h$ the latent variables, and $q(h|v)$ is the variational distribution to be optimized. Adding the constant $logp(v;\theta)$ , allows us to write the expression as:

$$\text{ELBO}(v, \theta, q) = E_{h \sim q}[\text{log}p(h, v)] + H(q) \tag{14}$$

which is easier to deal with as we can calculate $p(h, v)$ more readily that $p(h|v)$.

# References

[1] Yoshua Bengio, Ian J Goodfellow, and Aaron Courville. "Deep learning, book in preparation for mit press (2015)". In: *Disponıvel em http://www. iro. umontreal. ca/bengioy/dlbook* (2015).

[7] for proof see [1, Section 17.3]

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

[3] Jonathan Chang et al. "Reading Tea Leaves: How Humans Interpret Topic Models". In: *Advances in Neural Information Processing Systems*. Ed. by Y. Bengio et al. Vol. 22. Curran Associates, Inc., 2009.

[4] David Roxbee Cox. *Principles of statistical inference*. Cambridge university press, 2006.

[5] Susan T Dumais et al. "Using latent semantic analysis to improve access to textual information". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. 1988, pp. 281–285.

[6] Sadanori Konishi and Genshiro Kitagawa. *Information criteria and statistical modeling*. Springer Science & Business Media, 2008.

[7] Jey Han Lau, David Newman, and Timothy Baldwin. "Machine Reading Tea Leaves: Automatically Evaluating Topic Coherence and Topic Model Quality". In: *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Gothenburg, Sweden: Association for Computational Linguistics, Apr. 2014, pp. 530–539. DOI: 10.3115/v1/E14-1056. URL: https://www.aclweb.org/anthology/E14-1056.

[8] Yishu Miao, Lei Yu, and Phil Blunsom. *Neural Variational Inference for Text Processing*. 2016. arXiv: 1511.06038 [cs.CL].

[9] Xinyi Wang and Yi Yang. "Neural Topic Model with Attention for Supervised Learning". In: *Proceedings of AISTATS*. 2020.