Latent Dirichlet Allocation & Topic Modelling

Brian (Nzuki) Mutisyo

Harvard University

bnmutisyo@college.harvard.edu

Spring 2021

Abstract

Topic modelling provides a convenient way to analyze big unclassified text. Manual classification of thousands of documents by humans is a time consuming process and one with limited processing power. Advancement in Machine Learning methods provide tools and algorithms that can be applied to such classification. This paper provides a framework for topic modelling, using the topic model Latent Dirichlet Allocation from Gensim. The intention of this paper is more exploratory and provide an account of what goes into such a complex algorithm and explore how the modelling process happens. At the end, I look at various ways of evaluating the topics that come forth from this process and their limitations.

I. Introduction

Natural language refers to the way we, humans, communicate with each other (speech, text, signs e.t.c). Linguistics on the other hand is the scientific study of language, including its grammar, semantics, and phonetics. Combining these two, through Natural Language Processing (use of computers to process natural language), computers can be used to understand human text.

Humans are generally capable of understanding text and topics in a document. We can pick up context, emotion, similarities and extract meaning from a text. Machines generally have a harder time in understanding text. Latent Dirichlet Allocation (LDA) focuses on solving this problem by topic modelling, by creating a statistical model for discovering the abstract topics in a collection of documents.

To understand the underlying concepts, we can break down the semantics of the word as follows:

- *Latent:* Topic structures in a document are latent meaning they are hidden structures in the text.
- *Dirichlet:* The Dirichlet distribution determines the mixture proportions of the top-

ics in the documents and the words in each topic.

- *Allocation:* Allocation of words to a given topic.

More formally, LDA is a generative probabilistic model for collections of discrete data[1]. LDA suggests that words carry strong semantic information and documents discussing the similar topics will often use a similar group of words. Latent topics are discovered by identifying groups of words that frequently occur together within documents.

LDA also considers the structure of the documents themselves. It suggests that *documents* are probability distributions over latent topics and *topics* are probability distributions over words. This suggests that every document has a topic and each topic has a distribution of words associated with it.

II. Latent Dirichlet Allocation

LDA is a three-level hierarchical Bayesian model, in which each item of a collection is modeled as a finite mixture over an underlying set of topics. Each topic is, in turn, modeled as an infinite mixture over an underlying set of topic probabilities [1].

To better understand how this model works, let us consider the multinomial and the Dirichlet distribution which are the building blocks for this model.

i. Multinomial Distribution

The multinomial distribution is a generalization of the Binomial Distribution. It keeps track of trials whose outcomes can fall into different categories. An object is placed into category j with probability p_j. In our case, a word is placed into topic j with probability p_j. Therefore, if $\mathbf{X} \sim Mult_k(n, p)$, then the probability mass function is defined by:[2]

$$P(X_1 = n_1, \cdots X_k = n_k) = \frac{n!}{n_1!n_2! \cdots n_k!} \cdot p_1^{n_1} p_2^{n_2} \cdots p_k^{n_k} \tag{1}$$

ii. Dirichlet Distribution

The Dirichlet distribution is an important multivariate continuous distribution in probability and statistics. The dirichlet has two main properties that we care about in this paper. First, it is a multivariate generalization of the Beta distribution. Most importantly, in Bayesian statistics, the Dirichlet distribution is a popular conjugate prior for the Multinomial distribution. [1].

The Dirichlet distribution is a distribution with k positive parameters α^k with respect to a k-dimensional space. [3]. Consider the dirichlet distributions with order $k \geq 2$. Suppose:

$$\mathbf{X} = (x_1, \ldots, x_k)$$

$$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_k)$$

Then the Dirichlet Probability Density Function is defined as:

$$P(\mathbf{X}|\alpha_1, \cdots, \alpha_k) = \frac{1}{Beta(\boldsymbol{\alpha})} \prod_{i=1}^{k} x_i^{\alpha_i - 1} \quad (2)$$

The normalizing constant is the multivariate beta function, which can be expressed in terms of the gamma function:[3]

$$\mathrm{Beta}(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{k} \Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{k} \alpha_i\right)}, \qquad \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_k).$$

Generally, we denote such a distribution as $Dir(\alpha_1, \ldots, \alpha_k)$. For $k = 2$, the resulting pdf, $f(x_1, x_2)$, is a pdf of the Beta distribution with parameters α_1 and α_2 which is a special case of the dirichlet distribution.

iii. LDA as a Generative Model

LDA assumes the following generative process for each document d in a corpus(group of text):[1]

1. Choose $N \sim Poisson(\lambda)$
2. Choose $\theta \sim Dir(\alpha)$
3. For each of the N words in each document, d_n:
   (a) choose a topic $z_n \sim Multinomial(\theta)$
   (b) choose a word w_n from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic z_n.

In less formal terms, the hierarchy above demands us to:

- Decide on the number of words N the document will have (say, according to a Poisson distribution).
- Choose a topic mixture for the document (according to a Dirichlet distribution over a fixed set of K topics).
- Generate each word in the document by:
  – First picking a topic (according to the multinomial distribution that you sampled above;
  – Then using the topic to generate the word itself (according to the topic's multinomial distribution)

III. Dataset

We will be considering a collection of 50, 000 news articles published by 4 news companies from the past 17 years (2000 - 2017). The data contains:

- Unique ID for each article
- Title of the article
- Publication (News Company)
- Author of the article
- Date of Publication
- Date, year and month of publication
- *Content of the articles*

Content of articles, represents the actual newspaper articles that were written and published during the respective time periods they represent. This will be our main focus.

IV. Implementation & Model Building

i. Data Pre-Processing

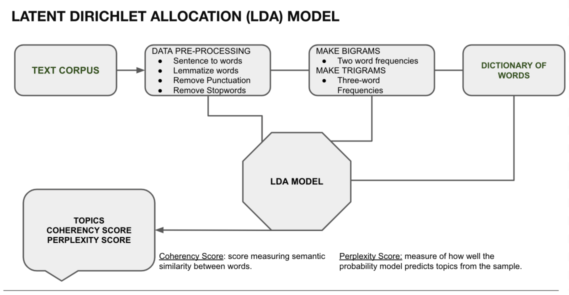
The data that we have is in its crude form, and we need to pre-process it as follows to make it more understandable by computers. My text pre-processing involved:

1. Convert sentences to words
2. Remove Punctuation
3. Remove Stop-words
4. Lemmatize the words
5. Build bigrams
6. Build trigrams
7. Create a dictionary of words

ii. Model Building

It was important for us to pre-process the data like we did in the previous section. This makes it easy for it to be understood by computers. Essentially, we started off with thousands of sentences, and we have converted them into numbers which can easily be interpreted by computers and is now ready to be passed into the model.

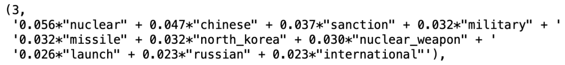
The LDA Model used in this case is from the Python's Library *Gensim*, which has inbuilt models which help in this modelling process. Similarly, other libraries like spaCy and Sci-kit Learn can be used for the same purpose.



Coherency Score: score measuring semantic similarity between words.

Perplexity Score: measure of how well the probability model predicts topics from the sample.

Figure 1: *Representation for LDA Modelling Process.*



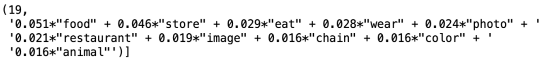
```
(3,
 '0.056*"nuclear" + 0.047*"chinese" + 0.037*"sanction" + 0.032*"military" + '
 '0.032*"missile" + 0.032*"north_korea" + 0.030*"nuclear_weapon" + '
 '0.026*"launch" + 0.023*"russian" + 0.023*"international"'),
```

Figure 2: *sample output 1*

V. Implementation Results

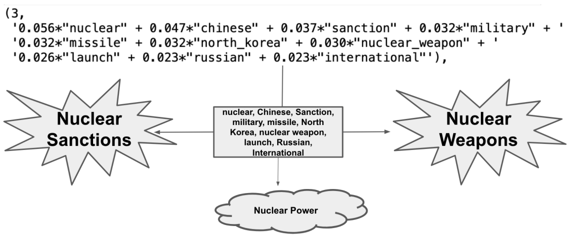
The first output of the model is the topics themselves. After approximating LDA's posterior distribution, the K topics are represented as multinomial distributions over *N* (Number of Words). Each topic distribution contains every word but assigns a different probability to each of the words. The words within topics with high probability are words that tend to co-occur more frequently. These high-probability words, usually the top 10 or top 15, are used to interpret and semantically label the topics as shown in *figures 2 and 3.* [4]

Notice the *weights* on each of the words making up the topics. The weights represent how important the corresponding key word was in coming up with the topic.



```
(19,
 '0.051*"food" + 0.046*"store" + 0.029*"eat" + 0.028*"wear" + 0.024*"photo" + '
 '0.021*"restaurant" + 0.019*"image" + 0.016*"chain" + 0.016*"color" + '
 '0.016*"animal"')]
```

Figure 3: *sample output 2*



```
(3,
 '0.056*"nuclear" + 0.047*"chinese" + 0.037*"sanction" + 0.032*"military" + '
 '0.032*"missile" + 0.032*"north_korea" + 0.030*"nuclear_weapon" + '
 '0.026*"launch" + 0.023*"russian" + 0.023*"international"'),
```

Nuclear Sanctions
nuclear, Chinese, Sanction, military, missile, North Korea, nuclear weapon, launch, Russian, International
Nuclear Weapons
Nuclear Power

Figure 4

i. Evaluating the topic Models

Now that we have built the topic models, we need to evaluate how good they are. Evaluating a topic model can help you decide if the model has captured the internal structure of a corpus (a collection of text documents). There are a couple of ways to evaluate topic models.

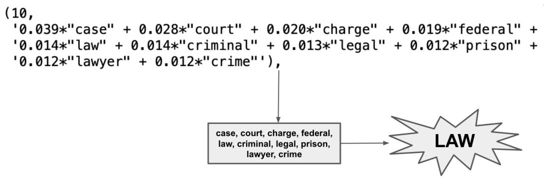
i.1 Human Judgement

Human Judgement involves using our own interpretations to classify the topics from the words making up a particular topic. Human judgement can take two forms:

- Observation-based e.g. on observing the top 'N' words in a topic and inferring a topic from those words.
- Interpretation-based, e.g. using 'word intrusion' and 'topic intrusion' to identify the words or topics that "don't belong" in a topic or document.

For instance, from results we obtained from our trained model, the following classification in *figure 4* can be made:

Here, we used the topic words (in the rectangular box) to make inference and assign labels to three topics that those words would most likely describe (nuclear power, nuclear weapons, nuclear sanctions).



```
(10,
 '0.039*"case" + 0.028*"court" + 0.020*"charge" + 0.019*"federal" + '
 '0.014*"law" + 0.014*"criminal" + 0.013*"legal" + 0.012*"prison" + '
 '0.012*"lawyer" + 0.012*"crime"'),
```

case, court, charge, federal, law, criminal, legal, prison, lawyer, crime
LAW

Figure 5

Same as above, using human judgement, the output above can most likely be linked to the topic 'Law'.

i.2 Using Quantitative Metrics

The next way of evaluating our topics would be to use quantitative metrics. This involves using Coherence scores and Perplexity (held out likelihood) calculations.

1. *Coherence Score*

   Topic Coherence measures score a single topic by measuring the degree of semantic similarity between high scoring words in the topic. These measurements help distinguish between topics that are semantically interpretable topics and topics that are artifacts of statistical inference.[5]

   The coherence score of a topic is computed as the sum of pairwise distributional similarity word scores over the set of topic words, V. This is generalized as:

$$coherence(V) = \sum_{(v_i, v_j) \in V} score(v_i, v_j, \epsilon) \tag{3}$$

where V is a set of word describing the topic and ϵ indicates a smoothing factor which guarantees that score returns real numbers. [5]

2. *Perplexity Score*

   The perplexity metric is a predictive one. It assesses a topic model's ability to predict a test set after having been trained on a training set. Perplexity is calculated by splitting a data set into two parts—a training set and a test set (i.e. held out documents) Likelihood is usually calculated as a logarithm, so this metric is sometimes referred to as the 'held out log-likelihood'.[6]

   The above can be generalized as: [1]

   $$perplexity(D_{test}) = \exp\left\{-\frac{\sum_{d=1}^{M}\log p(\omega_d)}{\sum_{d=1}^{M} N_d}\right\}$$

   where M is the number of documents (in the test sample), ω_d represents the words in document d, N_d the number of words in document d.

   Although the perplexity metric is a natural choice for topic models from a technical standpoint, it does not provide good results for human interpretation. [6] (i.e. perplexity does not capture the relationship between words in a topic or topics in a document).

i.3 Discussion

Throughout my modelling, I obtained the following table of results (*Table 1*).

Table 1

Table of Scores	
Coherence Score:	0.5242
Perplexity Score:	-10.4627

The coherence score is actually a good score. It shows that the model correctly identifies semantic similarity between words more than 50% of the time. The perplexity score, as it turns out, does not provide us with any useful interpretation of the text like we saw in the previous section.

One key hyper parameter that may improve the coherency score is the value of *number of topics*. For a coherency score of 0.524, I used $k = 20$ topics. To improve on the model, one can perform some hyper parameter tuning using grid search to find the optimal number of topics that maximizes the coherency score. However, $k = 20$ was more optimal for me to use because using more topics creates overlaps between the topics. For instance, using $k = 50$ produced irrelevant topics with less information in them.

This particular implementation faces several limitations. The first one is the choice of the python library for modelling. The *Gensim* library which I used lacks a well robust predictive element. *Scikitlearn* provides a different mode of topic modelling that can be used to predict topics of unseen documents once the model had been trained. This was a challenge for the project, because the library that I used provides a platform for classification, rather than one for both classification and prediction. To help mitigate this, using *Scikitlearn* rather than *gensim* would have yielded better results. However, Gensim has a more robust method of text pre-processing, due to the inbuilt preprocessing pipelines that it has.

VI. Conclusions

To summarize the conclusions above, we looked at how we can perform Topic Modelling on a given text corpus. We used the LDA algorithm to help us achieve this, though there are other tools like Multiple Linear Regression, Latent Semantic Analysis (LSA), Probabilistic Latent Semantic Analysis (PLSA), Latent and Correlated Topic Model (CTM) which can be used for topic modelling too.

We also showed how topic models can be evaluated - through human judgement, coherence and perpexity scores. However, there's no straight forward or reliable way to evaluate topic models to a high standard of human interpretability. In terms of quantitative approaches, coherence is a versatile and scalable way to evaluate topic models. But it has limitations. Topic modeling provides no guarantees about the topics that are identified (hence the need for evaluation) and sometimes produces meaningless, or "junk", topics. These can distort the results of coherence calculations.[6] There are also problems associated with the *reference corpus*. In cases where the probability estimates are based on the reference corpus, then a smaller or domain-specific corpus can produce misleading results when applied to a set of documents that are quite different to the reference corpus.

Overall, topic model evaluation isn't easy. There's no straight forward or reliable way to evaluate topic models to a high standard of human interpretability. Nevertheless, the most reliable way to evaluate topic models is by using human judgment. But this takes time and is expensive.

References

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[2] J. K. Blitzstein and J. Hwang, *Introduction to probability*. Crc Press, 2019.

[3] Wikipedia, "Dirichlet distribution."

[4] S. Syed and M. Spruit, "Full-text or abstract? examining topic coherence scores using latent dirichlet allocation," in *2017 IEEE International conference on data science and advanced analytics (DSAA)*, pp. 165–174, IEEE, 2017.

[5] K. Stevens, P. Kegelmeyer, D. Andrzejewski, and D. Buttler, "Exploring topic coherence over many models and many topics," in *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pp. 952–961, 2012.

[6] Giri, “Topic model evaluation.”