NATURAL LANGUAGE PROCESSING

Topic Modeling with LDA Explained: Applications and How It Works

By Giri • Updated on February 4, 2022

We are surrounded by large volumes of text—emails, messages, documents, reports—and it's a challenge for individuals and businesses alike to monitor, collate, interpret and otherwise make sense of it all. Over recent years, an area of natural language processing called topic modeling has made great strides in meeting this challenge. This article introduces topic modeling—its applications and how it works—through a step-by-step explanation of a popular topic modeling approach called Latent Dirichlet Allocation.

Contents

The volume of text that surrounds us is vast.

And it's growing.

Emails, web pages, tweets, books, journals, reports, articles and more.

And with the growing reach of the internet and web-based services, more and more people are being connected to, and engaging with, digitized text every day.

Accompanying this is the growth of text analytics services. Businesswire, a news and multimedia company, estimates that the market for text analytics will grow by 20% per year to 2024, or by over $8.7 billion.

As text analytics evolves, it is increasingly using artificial intelligence, machine learning and natural

As text analytics evolves, it is increasingly using artificial intelligence, machine learning and natural language processing to explore and analyze text in a variety of ways.

But text analysis isn't always straightforward.

One of the key challenges with machine learning, for instance, is the need for large quantities of labeled data in order to use supervised learning techniques.

Consider classifying spam emails. A supervised learning approach can be used to do this by training a network on a large collection of emails that are pre-labeled as being spam (or not). If such a collection doesn't exist however, it needs to be created, and this takes a lot of time and effort.

Supervised learning can yield good results if labeled data exists, but most of the text that we encounter isn't well structured or labeled.

This is where unsupervised learning approaches like **topic modeling** can help.

What is topic modeling?

Topic modeling is a form of unsupervised learning that identifies hidden relationships in data.

Being unsupervised, topic modeling doesn't need labeled data. It can be applied directly to a set of text documents to extract information.

Topic modeling works in an *exploratory* manner, looking for the themes (or topics) that lie within a set of text data.

There is no prior knowledge about the themes required in order for topic modeling to work.

It discovers topics using a probabilistic framework to *infer* the themes within the data based on the words observed in the documents.

Topic modeling is a versatile way of making sense of an unstructured collection of text documents.

It can be used to automate the process of sifting through large volumes of text data and help to organize and understand it.

Once key topics are discovered, text documents can be grouped for further analysis, to identify trends, for instance, or as a form of classification.

What is topic modeling used for?

What is topic modeling used for?

Topic modeling applications cover a range of use cases—here are a few real-world examples:

Annotation

Topic modeling can 'automatically' label, or annotate, unstructured text documents based on the major themes that run through them.

The labeled data can be further analyzed or can be an input for supervised learning models.

Topic modeling can therefore help to overcome one of the key challenges of supervised learning—it can create the labeled data that supervised learning needs, and it can be done at scale.

eDiscovery

In legal document searches, also called legal discovery, topic modeling can save time and effort and can help to ensure that important information isn't missed.

Legal discovery involves searching through all the documents relevant for a legal matter, and in some cases the volume of documents to be searched is very large. A 100% search of the documents isn't always viable, so it's easy to miss out on relevant facts.

Topic modeling can help with this, by revealing sufficient information about the documents even if all them aren't searched.

Herbert Roitblat, an expert in legal discovery, has successfully used topic modeling to identify all of the relevant themes in a collection of legal documents, even when only 80% of the documents were actually analyzed. This is because there were themes in common between the documents that were analyzed and those that were missed.

Content recommendation

In late 2015, the New York Times (NYT) changed the way it recommends content to its readers, switching from a filtering approach to one that uses topic modeling.

The NYT seeks to personalize content for its readers, placing the most relevant content on each reader's screen.

It used to do this by a simple keyword matching approach for each reader, later changing to a collaborative matching approach for groups of readers with similar interests.

The switch to topic modeling improves on both these approaches.

The NYT uses topic modeling in two ways—firstly to identify topics in articles and secondly to identify topic preferences amongst readers. The two are then compared to find the best match for a reader.

Search engine optimization (SEO)

In 2018, Google described an enhancement to the way it structures data for search—a new layer was added to Google's Knowledge Graph called a Topic Layer.

This is designed to "*deeply understand a topic space and how interests can develop over time as familiarity and expertise grow*"[1].

Google is therefore using topic modeling to improve its search algorithms.

By analyzing topics and developing subtopics, Google is using topic modeling to identify the most relevant content for searches.

Word sense disambiguation (WSD)

WSD relates to understanding the meaning of words in the context in which they are used.

This can be quite challenging for natural language processing and other text analysis systems to deal with, and is an area of ongoing research.

Recent studies have shown that topic modeling can help with this.

Research at Carnegie Mellon has shown a significant improvement in WSD when using topic modeling.

Traditional approaches evaluate the meaning of a word by using a small window of surrounding words for context. But this becomes very difficult as the size of the window increases.

With topic modeling, a more efficient scaling approach can be used to produce better results.

How topic modeling works

Latent Dirichlet Allocation

To understand how topic modeling works, we'll look at an approach called Latent Dirichlet Allocation (LDA).

This is a popular approach that is widely used for topic modeling across a variety of applications. It has

good implementations in coding languages such as Java and Python and is therefore easy to deploy.

LDA was developed in 2003 by researchers David Blei, Andrew Ng and Michael Jordan. Its simplicity, intuitive appeal and effectiveness have led to strong support for its use.

LDA topic modeling discovers topics that are hidden (latent) in a set of text documents.

It does this by inferring possible topics based on the words in the documents. It uses a generative probabilistic model and Dirichlet distributions to achieve this.

The inference in LDA is based on a Bayesian framework. This allows the model to infer topics based on observed data (words) through the use of conditional probabilities.

A generative probabilistic model works by observing data, then generating data that's similar to it in order to understand the observed data. This is a powerful way to analyze data and goes beyond mere description—by learning how to generate observed data, a generative model learns the essential features that characterize the data.

In LDA, the generative process is defined by a joint distribution of hidden and observed variables.

A Dirichlet distribution can be thought of as a *distribution over distributions*.

In the case of LDA, if we have *K* topics that describe a set of documents, then the mix of topics in each document can be represented by a *K*-nomial distribution, a form of multinomial distribution.

A multinomial distribution is a generalization of the more familiar binomial distribution (which has 2 possible outcomes, such as in tossing a coin). A K-nomial distribution has K possible outcomes (such as in a K-sided dice).

In LDA, the Dirichlet is a probability distribution over the K-nomial distributions of topic mixes.

But there's also another Dirichlet distribution used in LDA—a Dirichlet over the words in each topic.

So, LDA uses two Dirichlet distributions in its algorithm.

Why is all this useful?

By using a generative process and Dirichlet distributions, LDA can better genaralize to new documents after it's been trained on a given set of documents. This is an improvement on predecessor models (such as pLSI).

LDA preliminaries

The first thing to note with LDA is that we need to decide the number of topics, K, in advance.

By choosing K, we are saying that we believe the set of documents we're analyzing can be described by K topics.

If we're not quite sure what K should be, we can use a trial-and-error approach, but clearly the need to set K is an important assumption in LDA.

When analyzing a set of documents, the total set of words contained in all of the documents is referred to as the *vocabulary*.

Besides K, there are other parameters that we can set in advance when using LDA, but we often don't need to do so in practice—popular implementations of LDA assume default values for these parameters if we don't specify them.

We'll look at some of these other parameters later.

The second thing to note with LDA is that once the K topics have been identified, LDA does not tell us anything about the topics other than showing us the distribution of words contained within them (ie. the probability of each word in the vocabulary appearing in the topic).

We therefore need to use our own interpretation of the topics in order to understand what each topic is about and to give each topic a name.

The above two characteristics of LDA suggest that some domain knowledge can be helpful in LDA topic modeling.

Although it's not required for LDA to work, domain knowledge can help us choose a sensible number of topics (K) and interpret the topics in a way that's useful for the analysis being done.

The LDA algorithm—Step by step

Let's now look at the algorithm that makes LDA work—it's basically an iterative process of topic assignments for each word in each document being analyzed.

Recall that LDA identifies the latent topics in a set of documents.

What this means is that for each document, LDA will generate the topic mix, or the distribution of topics for each document.

All documents share the same K topics, but with different proportions (mixes).

Having chosen a value for K, the LDA algorithm works through an iterative process as follows:

Step 1

Initialize the model:

- Randomly assign a topic to *each word in each document*
- Note that after this random assignment, two frequencies can be computed—
    1. the counts (frequency distribution) of topics in each document, call this *topic frequency*
    2. the counts (frequency distribution) of words in each topic, call this *word frequency*

Step 2

Update the topic assignment for *a single word in a single document*:

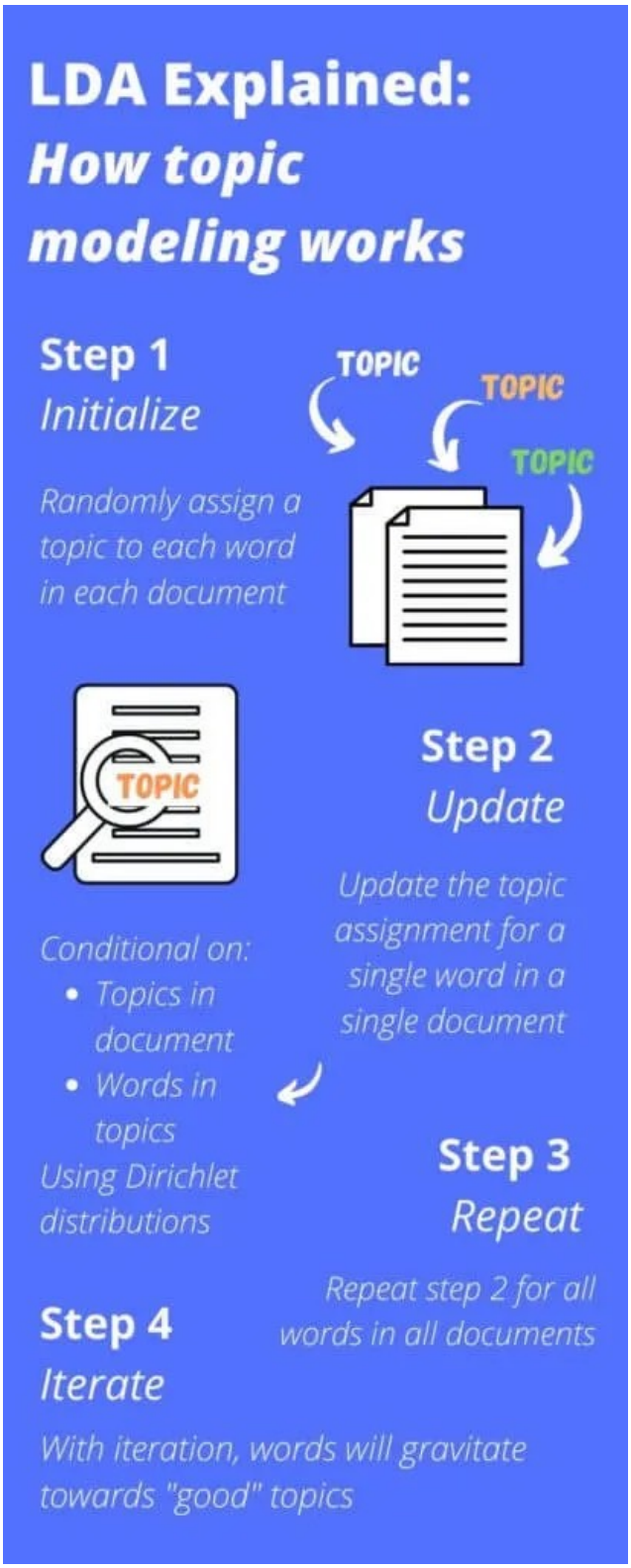
- Choose a word in a document
    - Un-assign its assigned topic (ie. un-assign the topic that was randomly assigned during the initialization step)
    - Re-assign a topic to the word, given (ie. conditional upon) all other topic assignments for all other words in all documents, by considering—
        1. the popularity of each topic in the document, ie. how many times the document uses each topic, measured by the frequency counts calculated during initialization (topic frequency) *and* a Dirichlet-generated multinomial distribution over topics for each document, and
        2. the popularity of the word in each topic, ie. how many times each topic uses the word, measured by the frequency counts calculated during initialization (word frequency), *and* a Dirichlet-generated multinomial distribution over words for each topic
    - Mulitply 1. and 2. to get the conditional probability that the word takes on each topic
    - Re-assign the word to the topic with the largest conditional probability

Step 3

Repeat Step 2 for all words in all documents.

Step 4

Iterate.



How LDA topic modeling works: A 4-step iterative algorithm using Dirichlet distributions

In Step 2 of the algorithm, you'll notice the use of two Dirichlets—what role do they serve?

As mentioned, by including Dirichlets in the model it can better generalize to new documents.

To understand why Dirichlets help with better generalization, consider the case where the frequency count for a given topic in a document is zero, eg. if the topic does not appear in a given document after the random initialization.

But the topic may actually have relevance for the document.

By including a Dirichlet, which is a probability distribution over the K-nomial topic distribution, a non-zero probability for the topic is generated.

Hence, the topic may be included in subsequent updates of topic assignments for the word (Step 2 of the algorithm).

This additional variability is important in giving all topics a chance of being considered in the generative process, which can lead to better representation of new (unseen) documents.

Why does the LDA algorithm work?

Step 2 of the LDA algorithm calculates a conditional probability in two components—one relating to the distribution of topics in a document and the other relating to the distribution of words in a topic.

Hence, each word's topic assignment depends on *both* the probability of the topic in the document and the probability of the word in the topic.

In this way, words will move together within a topic based on the *suitability* of the word for the topic and also the *suitability* of the topic for the document (which considers all other topic assignments for all other words in all documents).

Note that *suitability* in this sense is determined solely by frequency counts and Dirichlet distributions and not by semantic information. In this way, the observed structure of the document informs the discovery of latent relationships, and hence the discovery of latent topic structure.

In the words of Jordan Boyd-Graber, a leading researcher in topic modeling:

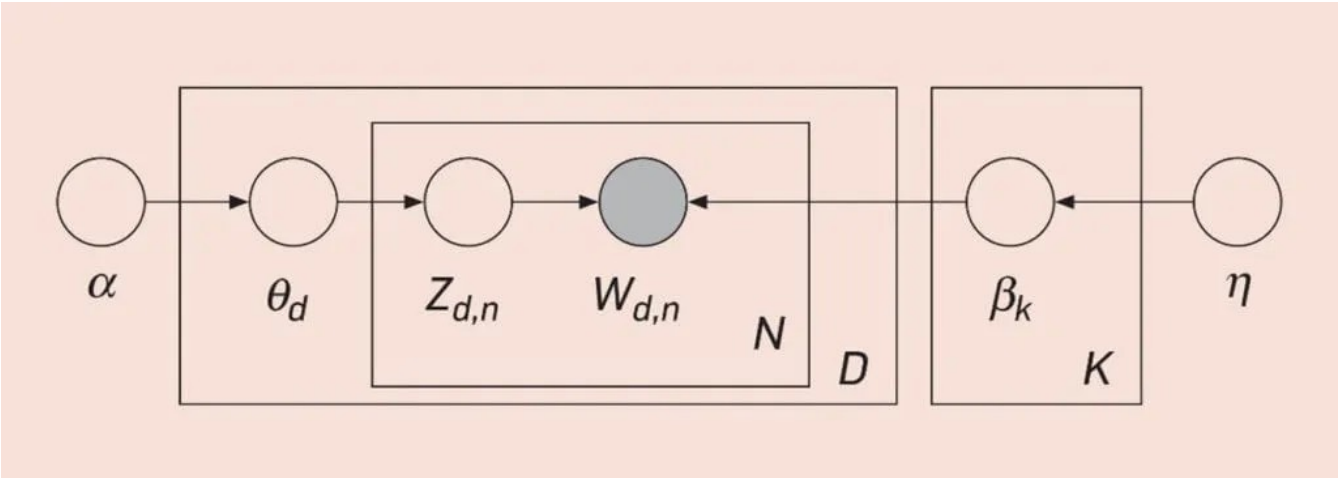
> "The initial [topic] assignments will be really bad, but all equally so. The words that appear together in documents will gradually gravitate towards each other and lead to good topics."
>
> Jordan Boyd-Graber, Associate Professor at the University of Maryland

Other parameters of LDA

Earlier we mentioned other parameters in LDA besides K.

Two of these are the *Alpha* and *Eta* parameters, associated with the two Dirichlet distributions.

To get a sense of how the LDA model comes together and the role of these parameters, consider the following graph of the LDA algorithm.



LDA graph model. Source: http://www.cs.columbia.edu/~blei/papers/Blei2012.pdf

In the above graph:

- θ_d is the topic mix for document d
- $Z_{d,n}$ is topic assignment for word n in document d
- $W_{d,n}$ is word n in document d
- N is the total number of words in all documents (the size of the vocabulary)
- D is the number of documents
- K is the number of topics
- β_k is the distribution of words for topic k

Alpha (α) and Eta (η) act as 'concentration' parameters. They correspond to the two Dirichlet distributions —Alpha relates to the distribution of topics in documents (topic mixes) and Eta relates to the distribution of words in topics.

As mentioned, popular LDA implementations set default values for these parameters. You can also set them manually if you wish.

The values of Alpha and Eta will influence the way the Dirichlets generate multinomial distributions. Higher values will lead to distributions that center around averages for the multinomials, while lower values will lead to distributions that are more dispersed.

To illustrate, consider an example topic mix where the multinomial distribution averages [0.2, 0.3, 0.5] for a 3-topic document. There are three topic proportions here corresponding to the three topics.

Note that the topic proportions sum to 1. Since this is a topic mix, the associated parameter is Alpha.

When a Dirichlet with a large value of Alpha is used, you may get generated values like [0.3, 0.2, 0.5] or [0.1, 0.3, 0.6] etc.

You can see that these topic mixes center around the average mix.

When a small value of Alpha is used, you may get values like [0.6, 0.1, 0.3] or [0.1, 0.1, 0.8]. Here, you can see that the generated topic mixes are more dispersed and may gravitate towards one of the topics in the mix.

Eta works in an analogous way for the multinomial distribution of words in topics.

The choice of the Alpha and Eta parameters can therefore play an important role in the topic modeling algorithm.

Text pre-processing and representation

It is important to remember that any documents analyzed using LDA need to be pre-processed, just as for any other natural language processing (NLP) project.

Pre-processing text prepares it for use in modeling and analysis. It is an essential part of the NLP workflow.

Some key pre-processing steps include:

- Tokenization, which breaks up text into useful units for analysis
- Normalization, which transforms words into their base form using lemmatization techniques (eg. the lemma for the word “studies” is “study”)
- Part-of-speech tagging, which identifies the function of words in sentences (eg. adjective, noun, adverb)

Also essential in the NLP workflow is text representation. This involves the conversion of text to numbers (typically vectors) for use in quantitative modeling (such as topic modeling).

There are a range of text representation techniques available.

You can learn more about text pre-processing, representation and the NLP workflow in this article.

Evaluation

Evaluation

Once you've successfully applied topic modeling to a collection of documents, how do you measure its success? How do you know if a useful set of topics has been identified?

To answer these questions you need to evaluate the model.

There are various ways to do this, including:

- Human testing, such as identifying which topics "don't belong" in a document or which words "don't belong" in a topic based on human observation
- Quantitative metrics, including cosine similarity and word and topic distance measurements
- Other approaches, which are typically a mix of quantitative and frequency counting measures

While these approaches are useful, often the best test of the usefulness of topic modeling is through interpretation and judgment based on domain knowledge.

This will of course depend on circumstances and use cases, but usually serves as a good form of evaluation for natural language analysis tasks such as topic modeling.

To learn more about the considerations and challenges of topic model evaluation, see this article.

Hands-on examples

To get a better sense of how topic modeling works in practice, here are two examples that step you through the process of using LDA.

Both examples use Python to implement topic models using the gensim package.

The *first example* applies topic modeling to US company earnings calls—it covers sourcing the transcripts, text pre-processing, LDA model setup and training, evaluation and fine-tuning, and applying the model to new unseen transcripts. You can check it out here.

The *second example* looks at topic trends over time, applied to the minutes of FOMC meetings. Here, after identifying topic mixes using LDA, the trends in topics over time are extracted and observed. You can check it out here.

Conclusion

We are surrounded by *large and growing* volumes of text that store a wealth of information.

In order to analyze this, many modern approaches require the text to be *well structured or annotated*. This is *difficult and expensive* to do.

Topic modeling is an area of natural language processing that can analyze text *without the need for annotation*—this makes it *versatile and effective for analysis at scale*.

A popular approach to topic modeling is **Latent Dirichlet Allocation** (LDA).

Topic modeling with LDA is an *exploratory* process—it identifies the *hidden topic structures* in text documents through a *generative probabilistic* process. These identified topics can help with understanding the text and provide inputs for further analysis.

LDA uses *Bayesian statistics* and *Dirichlet distributions* through an iterative process to model topics.

The essence of LDA lies in its *joint exploration* of *topic distributions* within documents and *word distributions* within topics, which leads to the identification of coherent topics through an iterative process.

LDA is a widely used approach with good reason—it has *intuitive* appeal, it's *easy* to deploy and it produces *good results*.

It also helps to solve a major shortcoming of supervised learning, which is the need for labeled data.

Topic modeling is an evolving area of NLP research that promises many more *versatile use cases* in the years ahead.

FAQs

What is a* topic *in topic modeling?
Topic modeling is a form of **unsupervised learning** that can be applied to unstructured data. When topic modeling is applied to a set of text documents, the words in the documents that are **most likely to co-occur** are collected together into **groups**. These groups of words are referred to as **topics**.

How does LDA work?
LDA is *Latent Dirichlet Allocation* which uses **Dirichlet distributions** to discover hidden, or **latent**, topics in a set of documents. It works through a **4-step iterative process** of (1) *Initializing* topic assignments for each word in the documents, (2) *Updating* the topic assignment for a given word based on the probabilities of co-occurrence with other words and topics and Dirichlet variability, (3) *Repeating* for all words in all documents, and (4) *Iterating* the process.

What are common topic modeling applications?
Being unsupervised, topic modeling is useful for **analyzing unstructured text data** that has no labels.