Machine Learning for Language Processing
Lecture 5: Topic Modelling and LDA

Stephen Clark

October 20, 2015

Probabilistic Topic Modelling Probabilistic topic modelling provides a suite of techniques for automatically finding structure in documents. Latent Dirichlet Allocation (LDA) is one prominent example of an approach to topic modelling, based on Bayesian probabilistic modelling. LDA attempts to find themes, or topics, in a set of documents, with the idea that these themes can be used for some downstream searching or browsing task. For example, a scientific search engine could organise its search results by topic (e.g. *genetics, evolution, disease, computation*), rather than simply provide a ranked list.

Crucially, the set of topics is not fixed in advance, nor does LDA rely on any human manual annotation. This is attractive because LDA can be applied to any document set, and the user does not require any knowledge in advance of what topics might be present; nor does the user have to perform any expensive annotation. Essentially we can think of LDA as performing clustering: grouping documents (probabilistically) into clusters, each document cluster corresponding to a topic; and grouping words (probabilistically) into clusters, each word cluster defining a topic.

One downside of LDA is that the topics are not automatically provided with labels – they're simply clusters. Hence how to label, or interpret, clusters after they have been produced is an important research problem.

Key Assumptions behind LDA LDA makes the following modelling assumptions. First, each document is a *mixture* of topics. However, it is typically assumed that each document contains only a few topics. (The latter assumption is controlled by one of the model's parameters.) LDA is a probabilistic model with a corresponding *generative process*, and each document is assumed to be created, or generated, by that process. The structure of the process, defined by a graphical model which encodes some random variables and independence assumptions, is given in advance; hence the inference problem, given a set of documents, is to infer the parameters which determine the various probabilities associated with the graphical model.

A *topic* is defined as a probability distribution over a fixed vocabulary of words. Note that each topic is a distribution over the same vocabulary, so it's not strictly correct to say that LDA groups words into topics (at least hard topics); however, the words which appear "at the top" of the distribution with high probability will be different for each topic. Hence LDA does perform a probabilistic, soft clustering of the vocabulary.

One feature of LDA which is important to grasp is that it's a Bayesian modelling framework, in that the probability distributions themselves are also generated as part of the generative process. Topics are generated first, before the documents. Hence we require distributions over distributions, which is where the Dirichlet distribution comes in. The Dirichlet distribution allows us to encode a prior expectation that each topic will be dominated by a relatively small number of words (in terms of the probability mass assigned to those words); and also that each document will be made up of only a few topics.

The only parameters that require specifying in advance are the number of topics, and the parameters of the Dirichlet distributions (which encode how sparse we'd like the topic and document distributions to be).

Example Topics The clusters on the slide are the 15 most probable words taken from a set of topics (with the corresponding probabilities not shown). So here we've effectively created a hard clustering by applying a cut-off after the top 15 words in each topic. In this example, taken from David Blei, the topics have been obtained from a set of scientific documents. The topics are relatively coherent and meaningful. It is interesting to consider what label you might assign to each topic to best describe it.

Documents and Topics The example on the slide is designed to show how a document can be thought of as a small set of topics, with each word in the document assigned to a single topic. The words in yellow appear to be related to genes; the words in purple to life and evolution; and the words in blue to computational analysis. Other topics may be less meaningful, or contain words with less semantic content (such as function words). How to subsequently use, or interpret, the topics is a separate research question.

The next slide, again from David Blei, shows the generative process applied to a single document.

The Generative Process The first task, before any documents are generated, is to generate the topic distributions (topics shown to the left on the pictorial slide, but not in the pseudocode); how this is done will be explained later. The first task when generating a document is to generate a distribution over topics. Intuitively, we want to say something like: 45% of this document will be made up of the "gene" topic, 20% of the "life" topic, and so on. Now each word is generated. This is done by first choosing a topic (according to the distribution just created), and then choosing a word from that topic (which is a distribution over words, remember).

Note that, as a statistical model of text, the generative process is extremely simple: it's a bag-of-words model where each word is generated independently of every other word. Other than a "set-of-words" model, the bag-of-words model is as simple as it gets when it comes to modelling text (and from a linguistic perspective totally stupid).[1] However, decades of work in NLP, and IR in particular, has shown that it's possible to build effective NLP applications using this model.

The (Formal) Generative Process The joint distribution at the bottom of the slide is the probability of a set of documents, from 1 to D, together with the associated probability distributions. First choose the topics, β_i, from 1 to K, each one independently; these will be chosen according to a Dirichlet distribution, with parameter η. For each document d, choose the topic proportions, θ_d, again independently and again according to a Dirichlet distribution (with parameter α). Then, for each word position n in document d, independently choose a topic $z_{d,n}$. Finally, choose a word $w_{d,n}$ for that position, given the topic (distribution), where each word is conditionally independent given the topic.

LDA as a Graphical Model The *plate diagram* on the slide is a diagrammatic notation for representing graphical models. The arrows represent conditional independence, as in standard graphical model notation, and the boxes represent *repetition*. For example, the box with the D in the bottom corner means that the θ_d distribution has to be generated D times, once for each document; and the box with the N in the bottom corner means that the θ_d distribution has to be sampled from N times, once for each word position (to give the $z_{d,n}$ distibutions). Note that the N box is within the D box, meaning that N seperate draws have to be made for each d. Finally, the $z_{d,n}$ distribution (topic) has to be sampled from N times to generate each word, $w_{d,n}$.

There is a crucial difference between the unshaded white nodes and the nodes shaded grey. The grey nodes are *observed* in the data; i.e. their values are known. The white nodes are unobserved, and their values have to be inferred. The plate diagram neatly expresses the significant amount of information we're attempting to infer, given only the words in the documents.

The Dirichlet Distribution The key point to remember about the Dirichlet distribution is that it's a *distribution over distributions*. Hence the "observations" $\boldsymbol{x}$ on the slide are vectors of values, with the values x_i summing to 1. The parameter $\boldsymbol{\alpha}$ is also a vector, with one parameter α_i for each x_i.

[1] There are many extensions to the basic LDA model, for example one which takes some word order into account.

The formula for the Dirichlet distribution is provided for completeness on the slide, with the comment that the Dirichlet is *conjugate prior* to the multinomial (and categorical) distribution. If the binomial distribution provides the probability of the number of heads given a series of coin tosses, the multinomial is the extension of this to the many-sided die case. The idea of the conjugate prior is that the form of the posterior distribution (after we've seen the data) is the same as the form of the prior, which greatly simplifies the problem of statistical inference.

The key intuition underlying the use of the Dirichlet distribution is provided by the picture. Here we have "observations" which are probability distributions over 3 values, x, y and z. The plots show how likely a particular distribution is, given the $\boldsymbol{\alpha}$ values shown. So with $\boldsymbol{\alpha} = (6, 2, 2)$, for example, the chances of getting a distribution with most of the mass concentrated on x is high. In practice for topic modelling, it is typical to use the same value for all the α_is, so that there is a single α parameter. The effect of the α parameter is to determine how sparse the resulting distribution is likely to be. For an α value less than 1, there is pressure to choose distributions over topics which favour just a few of the topics.

Parameter Estimation Our main concern is to estimate the topic distributions β_k, i.e. the probability of each word given the topic. We are also concerned with the distributions $\theta_{d,k}$, i.e. the probability of each topic for each document in the collection. Attempts have been made to estimate these distributions directly, e.g. using EM, but a more common, and successful, approach has been to get at these distributions indirectly, via the probability of each word *token* in the collection belonging to a particular topic. Technically this is achieved by marginalising out, i.e. summing over, the distributions β and θ. Here we'll gloss over this aspect of the mathematics, and proceed directly to the estimates provided by *Gibbs sampling*.

Estimates using Gibbs Sampling The application of Gibbs sampling to a collection of documents is straightforward [1]. For each word token in turn, estimate the probability of that word token being assigned each topic, keeping fixed the topic assignments for all the other word tokens. Then using this distribution over topics, sample a topic, and assign it to the word token.

The Gibbs sampling algorithm stores count matrices C^{WT} and C^{DT}. C^{WT} has the counts for each word *type* and topic, and C^{DT} has the counts for each document and topic. Initially each word token is assigned a random topic, and the count matrices are calculated. Then, each word token is considered in turn. First, the count matrices are decremented by one for the entries that correspond to the current topic assignment to the token under consideration (since we want to calculate probabilities based on all the *other* topic-token assignments). Then the probability of this token being assigned each topic is given by the formula on the slide, which is used to generate a sample and the new assignment is made.

The two relative frequencies on the slide have intuitive interpretations. The one on the left measures how often w_i has been assigned topic j across all documents; and the one on the right measures how often topic j has been assigned to words in document d_i. Increasing either of these counts will increase the chance that the ith token will be assigned topic j. Notice also the effect of η and α, which are the parameters of the two Dirichlet distributions. Here they can be seen as acting as smoothing parameters in the relative frequency estimates (much like add-one smoothing).

Finally, note that the formula given on the slides is an unnormalised probability estimate. These estimates need to be divided by a normalising constant, which is a sum of the unnormalised values across all topics.

The Final Estimates Once the Gibbs sampler has been run for a number of iterations, the final sample can be used for calculating the final count matrices, which can then be used to calculate the (smoothed) relative frequency estimates on the slide.

Why does topic modelling work? Behind all the fancy mathematics lies a simple intuition: *words which tend to appear in the same documents get clustered in the same topics.*

Readings for Today's Lecture David Blei's topic modelling website has a host of useful material: http://www.cs.columbia.edu/∼blei/topicmodeling.html. A number of pictures on today's slides were stolen from David's 2012 ICML tutorial. A good place to start is his general introduction to topic modeling.

References

[1] Mark Steyvers and Tom Griffiths. Probabilistic topic models. In T. Landauer, D. Mcnamara, S. Dennis, and W. Kintsch, editors, *Latent Semantic Analysis: A Road to Meaning.* Laurence Erlbaum, 2006.