

Data Analysis with Human-Generated Text using LDA

In the previous document, we looked at using K-means clustering in practice to try to understand DNA as a genetic code. In this video, we'll look beyond K-means and clustering to see how some of the extensions we discussed earlier can be useful in real-life data analysis. In particular, we will analyze human-generated text data.

It's common nowadays to have access to a large text corpus. The text of Wikipedia, for example, is publicly available. We can even look at the text of news releases from MIT. But text data can arise in many other forms. A news website might have the text of all of its articles, and the company running the website might be interested in finding natural ways to present this text to its audience. It might be helpful to present topics in the text for easy navigation to relevant news articles. Or a search engine might be interested in analyzing the text of websites it crawls for convenient presentation of its search results.



For instance, Michael Jordan is the name of both a famous basketball player and a famous statistics researcher. Suppose someone enters Michael Jordan as an input into our Internet Search Engine. Now, if we present the search result by topic instead of solely by popularity, then we can quickly reach both the audience searching for sports information as well as the audience searching for scientific research.

So there are a lot of reasons we might be interested in finding the topics or themes in a collection of text. But as we mentioned in the previous document, clustering by itself can be a little problematic here, because a single document might exhibit multiple topics.

What if Michael Jordan the statistician writes a statistical analysis of the other Michael Jordan's basketball career or what if I look at the Wikipedia page for the sports figure, Michael Jordan? Actually, Wikipedia covers not just his basketball career, but the history of his brief time playing baseball as well as his experience as a businessman and his acting (Remember Space Jam?). So it makes a lot of sense to use a model that lets each data point or each article exhibit multiple topics. Recall that one term for this is a mixed membership model.

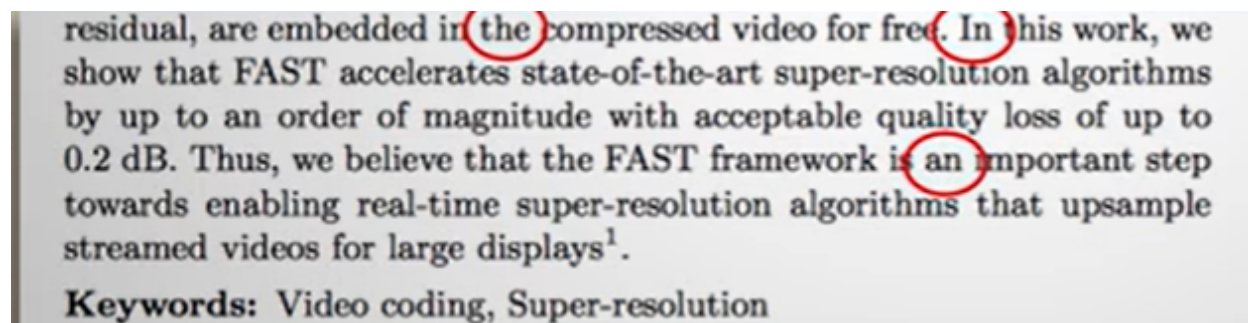
A particularly popular model in this domain is known as **Latent Dirichlet Allocation** or **LDA**.

To see what it looks like to use this model in practice, let's use LDA to analyze what MIT faculty are working on and their research. The code and visualizations for this analysis were all provided by Julia Lye. The Electrical Engineering and Computer Science department and MIT is really big. It's the biggest department on campus with over 130 faculty. And there are a number of major research labs that those faculty belong to. Four of the main labs in EECS are the Computer Science and Artificial Intelligence Laboratory (CSAIL), the Laboratory for Information and Decision Systems (LIDS), the Microsystems Technology Laboratories (MTL), and the Research Laboratory of Electronics (RLE).

If you've never heard of these labs before, it's not necessarily clear what they all do. And even if you're a member of this department, it's hard to keep track of the research of 130 other faculty members. But you might want a high-level summary of what everyone else is working on. For instance, if some area of research looks interesting, you can go talk to the faculty working in that area. This line of thought suggests we should try to find latent groups of research topics among the faculty. But a faculty member can work on multiple research topics at a time. So we're going to try to learn a mixed membership model like LDA.

To create our dataset we assemble abstracts from each professor's published papers for a total of over 900 abstracts. Each abstract is a text document that briefly describes a particular piece of research a professor conducted.

A common preprocessing step for LDA is to remove the most common words, words like 'A', 'The', 'In', etc that don't tell us much about a document. And the least common words, words that are so rare they don't tell us much across documents, are also removed.



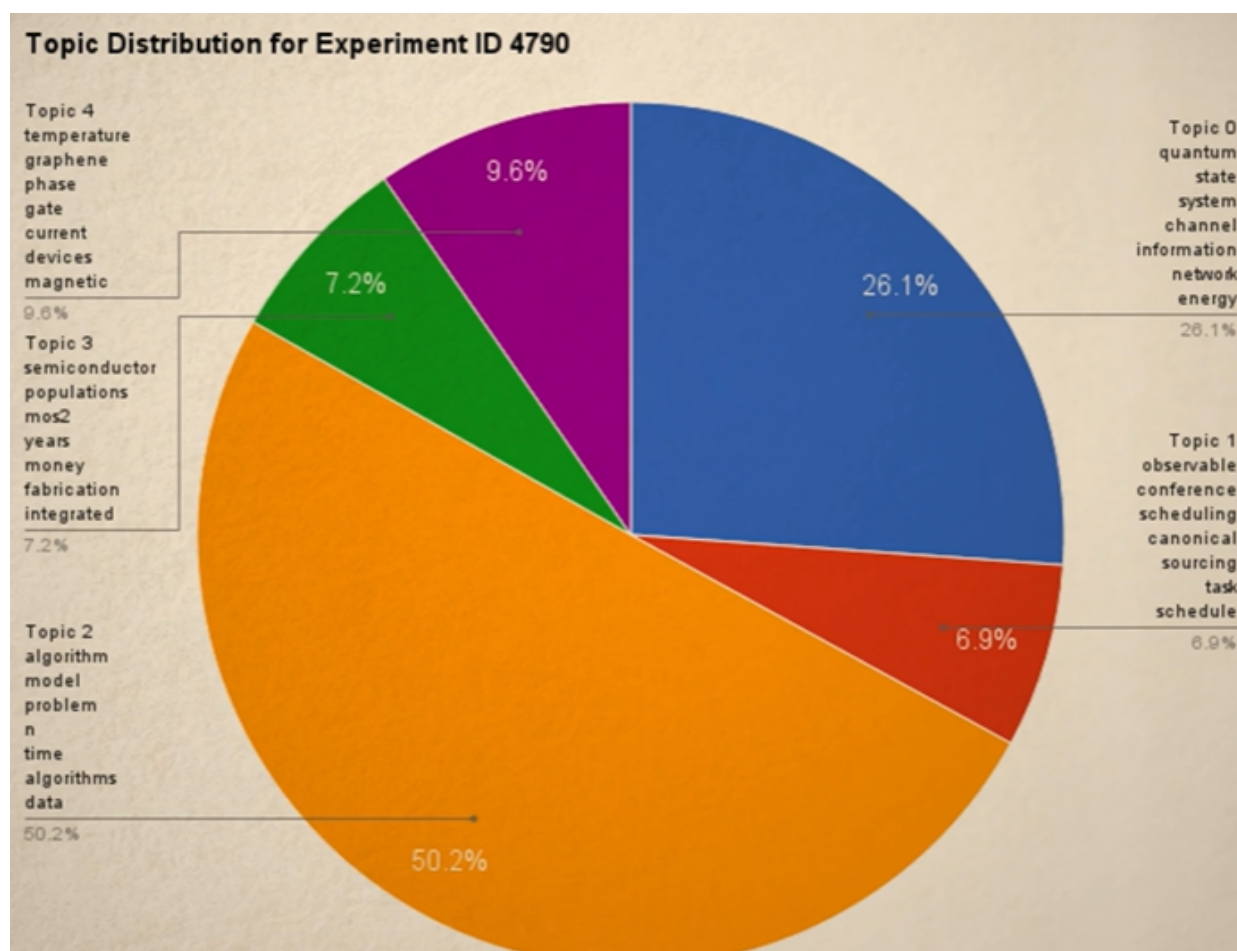
residual, are embedded in the compressed video for free. In this work, we show that FAST accelerates state-of-the-art super-resolution algorithms by up to an order of magnitude with acceptable quality loss of up to 0.2 dB. Thus, we believe that the FAST framework is an important step towards enabling real-time super-resolution algorithms that upsample streamed videos for large displays¹.

Keywords: Video coding, Super-resolution

Just like we need to specify the number of clusters in advance in K-means, here we need to specify the number of topics in advance for LDA. And just like we call the number of clusters K for clustering, here we can call the number of topics = K.

In an earlier video, we talked about heuristics to choose the number of clusters in a clustering problem. Similar heuristics can be used to find the number of topics in topic modeling.

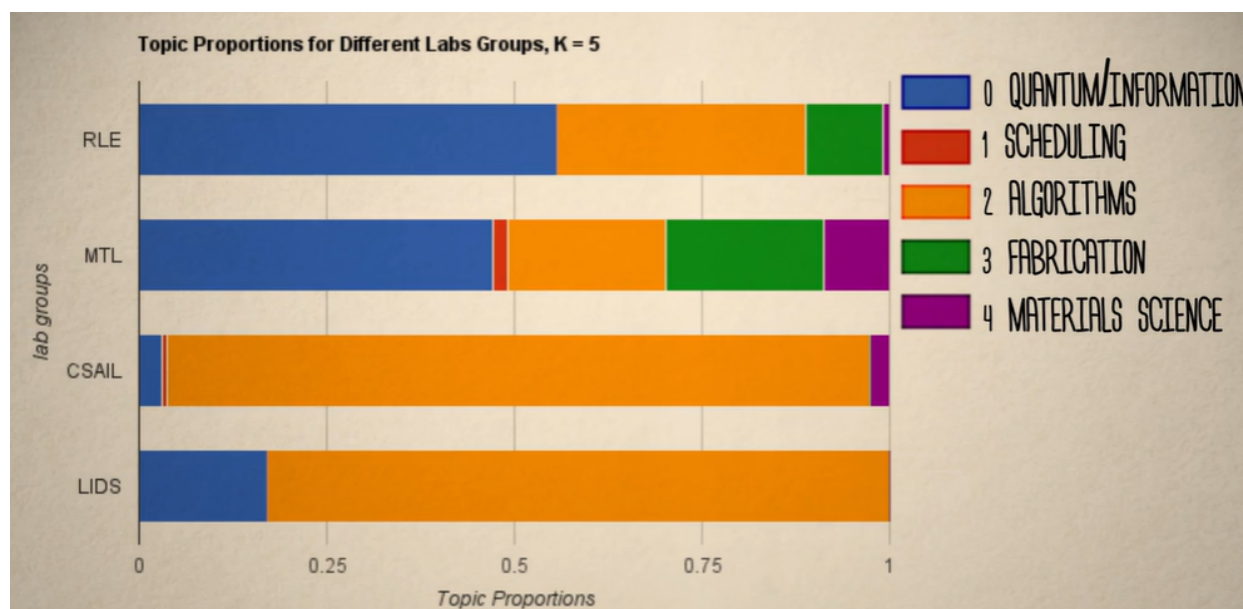
Heuristics for this problem suggest that a smaller K yields more meaningful results and hence we choose $K = 5$ topics. At this point, we can run a stochastic variational inference algorithm for LDA and look at the results. Just like we discover clusters in clustering, in topic modeling, we discover topics.



So, the first thing to do is to look at the topics. We can visualize topics by looking at the most common words in each topic. In this case, one of the topics has words like quantum, state, system, channel, and information. This topic seems to be capturing a line of research on quantum computation, communication, and information theory. Another topic has words like algorithm, model, problem, n, time. This is a topic on traditional computer science themes, like algorithms and complexity theory. 'n' is a symbol typically used to express the number of data points in a data problem. Yet another topic has words like temperature, graphene, phase, and gate. These are words related to material science.

So one of the first things we notice here is that research in the department is pretty diverse, which makes sense since the department contains both electrical engineering and computer science.

We can also look at the topics of research in each lab.



To generate this figure we consider all the research abstracts within each class and show what proportion of the research text falls in each topic category.

First, we notice that CSAIL and LIDS are both dominated by algorithm topics. This makes sense as CSAIL essentially contains all the computer science professors within the EECS Department. And the faculty in LIDS work on similar problems but perhaps a little more focused on information theory. We see this reflected in the higher proportion of the topic on quantum and information-related themes. RLE and MTL both contain words related to circuits. Again, this makes sense since these labs form the more traditional lab-based side of electrical engineering. Finally, MTL is distinguished by a significant proportion of research tasks related to material science.

So the nice thing about this analysis is that we learned all of these topics and the breakdown of labs by topic automatically. Few people, including professors at MIT, would be able to describe exactly what all these different MIT professors work on. But we were able to automatically pick up what words go together and describe coherent themes of this research. And we were able to distinguish different parts of the department, by the different types of research that goes on there. You could imagine doing a similar analysis for a company or for the user base of an app. What are the different themes users talk about, or post on a forum? How do different forums differ in what they focus on?

This type of analysis can be very useful for understanding a user base or for generating a quick summary of a lot of text documents that might otherwise be difficult to navigate.

In this document, we've seen how extensions of clustering can be useful in practice. In particular, we investigated a data analysis problem with human-generated text using Latent Dirichlet Allocation or LDA. We've come a long way in this module and sub-module. We started by introducing unsupervised learning. And then we focused in-depth on clustering and K-means clustering, in particular. We saw all the ins and outs of using K-means clustering in practice. And this sort of detail and care is necessary for any machine learning or statistics algorithm. **No algorithm is entirely a black box.** You want to understand what assumptions are going into your algorithm. And there are always assumptions. Once you understand these assumptions, you can design if your algorithm is the right approach to a problem or if you need an extension or something totally new.