A BRIEF MATHEMATICAL INTRODUCTION TO TOPIC MODELING

BRANDEN CIRANNI

1. Introduction. In the digital age, we sometimes have so much information that it could be difficult to make sense of it all. Consider a motivating example: Suppose you are a Data Engineer at Amazon and your manager has received complaints that it is hard to identify the pros and cons of products that have thousands of reviews. There is too much information for a consumer to sift through, leading to frustration and lower conversion rates. You have been tasked with creating a model to solve this problem that allows the user to filter reviews by topic. So if someone were buying a computer, they could filter to the reviews about the Processor, RAM, Storage, etc. However, your model isn't given a pre-made list of topics; It needs to create its own *novel topics* so it can pick up on unique discussions like "overheating issue" or "sticky shift key". This is the goal of **Topic Modeling**: To decompose this large quantity of information into a much smaller set of novel topics that are shared across the reviews, and allow any future reviews to be similarly classified.

To generalize, Topic Modeling is a type of dimensionality reduction that seeks to express any text document in k dimensional space where k is some chosen number of topics. For a *corpus* of documents $\mathbf{D}$ with m documents, and n unique words used across them, express each document as a n-dimensional vector, with the entry $\mathbf{D_{i,j}}$ corresponding to the *count* of word j in document i. A row of $\mathbf{D}$ looks like

$$\begin{array}{rc} & \begin{array}{ccccc} Word_1 & Word_2 & Word_3 & \dots & Word_n \end{array} \\ D_i = & \begin{pmatrix} 1 & 0 & 2 & \dots & 0 \end{pmatrix} \end{array}$$

The goal is then to reduce $\mathbf{D}$ to the much smaller $m \times k$ matrix of document-topic weights so it can be understood as a distribution of topics.

We consider three approaches for this: *Latent Semantic Indexing (LSI)* which hinges heavily on Singular Value Decomposition, *Non-negative Matrix Factorization (NMF)* which finds a low-rank approximation based on a constrained optimization problem, and *Latent Dirichlet Allocation (LDA)*, a probabalistic approach supposing each document to be a mixture model over latent topics.

2. Singular Value Decomposition. The backbone of the above-mentioned LSI model is Singular Value Decomposition. To understand the importance of SVD, it helps to examine the meaning of the matrices $\mathbf{U}$ and $\mathbf{V}$. In finding the SVD, we factor $\mathbf{D}$ as $\mathbf{D} = \mathbf{U\Sigma V^T}$ with $\mathbf{U}$ and $\mathbf{V}$ being orthogonal matrices containing the eigenvectors of $\mathbf{DD^T}$ and $\mathbf{D^TD}$ respectively. To consider what $\mathbf{DD^T}$ means in this context, you can verify to yourself that each cell $\mathbf{DD^T_{i,j}} = \langle D_i, D_j \rangle$, represents some measure of similarity between two documents. We interpret this as a *document-to-document correlation* matrix. We can see similarly, $\mathbf{D^TD}$ gives us a *term-to-term correlation* matrix [1] [2]. The eigenvectors that make up the columns of $\mathbf{U}$ and $\mathbf{V}$ are bases for the correlation matrices, and the square root of the eigenvalues are the singular values in $\mathbf{\Sigma}$, with the eigenvalues and eigenvectors sorted by the magnitude of the eigenvalues by convention. The magnitude of these eigenvalues signifies the relative *importance* of a correlation factor [1].

Now the problem is to shift from correlations between documents and terms, to correlations between documents and the k *latent topics* that we wish to find. To do this, we reconstruct the original matrix from the decomposition, only keeping the k

largest singular values, setting the rest to zero [2]. That is, we only consider the most important correlation factors between documents. These k most important factors become our *latent topics*. With the final $m-k$ rows of $\mathbf{\Sigma}$ set to the zero vector, observe that the last $m-k$ columns of $\mathbf{U}$ multiply to 0 during matrix multiplication, as will the last $n-k$ rows of $\mathbf{V^T}$. Thus, under these conditions $\mathbf{D} \approx \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^{\mathbf{T}}$, where $\hat{\mathbf{U}}$ is the $m \times k$ document-topic correlation matrix, the desired k dimensional topic-model.

3. Constrained Optimization. For the next type of Topic Modeling, I introduce a conceptually simple matrix method purely based on a constrained optimization problem that can be more efficient in compute time compared to SVD [3]. We define the problem as finding matrices $\mathbf{W}$ and $\mathbf{H}$ such that $\mathbf{D} \approx \mathbf{WH}$ with the constraint $\mathbf{W}, \mathbf{H} \geq 0$ [3]. The understanding for this, is that we are trying to factor $\mathbf{D}$ into the *Document-Topic* matrix $\mathbf{W}$ and the *Topic-Term* matrix $\mathbf{H}$, so we iteratively consider possible candidates for each, take their matrix product, and adjust the terms in each until a close approximation is obtained. We can define a squared error loss function $||\mathbf{D} - \mathbf{WH}||^2$ and optimize this according to update rules which guarantee a non-increasing euclidean distance $||\mathbf{D} - \mathbf{WH}||$. This is not only a possible improvement in terms of speed, but the cost function is guaranteed to be convergent to at least a local minima under the specified constraints, and thus provides a high quality of topics, given my our matrix $\mathbf{W}$.[3].

4. The Dirichlet Distribution. LDA takes an interesting approach - it chooses topics which allow it to *generate* the most similar documents compared to the actual ones [4]. To understand this, we first look at the Dirichlet Distribution. The k-dimensional Dirichlet Distribution, denoted $Dir(\alpha)$ is a distribution over the $(k-1)$ simplex, parameterized by a *concentration* α. For simplicity, consider the $k = 3$ case, as the 2-simplex is the equilateral triangle. Consider each corner of the triangle to be a topic, and say document d is a point on the triangle. For small α, d is likely to be closer to the corners and vice versa, and d's distance from each corner is it's distribution over the topics. Our goal is to choose topics that optimize the placement of documents so the most correlated ones are closest together.

The reason the Dirichlet Distibution is special is that for any document, it produces a probability vector that adds up to 1 for input into the Multinomial Distribution! The combination of these two distributions gives us a generator. Input a document, and the generator yields a Multinomial over topics. Then, consider another $Dir(\beta)$ over the $(n-1)$ simplex of words. Input a topic, and it yields a Multinomial distribution over words. Chaining the two, we can input a document and generate a set of words, a new document [4]! We repeat this for all documents, compare the generated documents to the actuals, and through optimization, choose the set of topics which minimizes the reconstruction error (the optimal arrangement of documents on the $(k-1)$ simplex relative to the latent topics). This is our k-dimensional topic-model.

REFERENCES

[1] Steven L. Brunton and J. Nathan Kutz. *Data Driven Science and Engineering*. Brunton & Kutz, Seattle, WA, USA, 2017.

[2] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard. Harshman. Indexing by latent semantic analysis. *Journal of The American Society for Information Science*, 1990.

[3] Daniel D. Lee and Sebastian H. Seung. Algorithms for non-negative matrix factorization. *Neural Information Processing Systems Foundation*, 2000.

[4] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 2003.