Article Recommendation System using Probabilistic Topic Models

Nilay Chakraborty
Rutgers University
New Brunswick, NJ-08901
nilay.chakraborty@rutgers.edu

Sumit Das
Rutgers University
New Brunswick, NJ-08901
sumit.das@rutgers.edu

Saurabh Gokhale
Rutgers University
New Brunswick, NJ-08901
saurabh.gokhale@rutgers.edu

Abstract

The main aim of our project is to develop a web based efficient Article recommendation system using Latent Dirichlet Allocation algorithm (LDA) based Probabilistic Topic Models . The recommendation system will evaluate the similarity between documents based on their thematic information and help suggest relevant articles to the reader for further reading.

1. Introduction

To build a successful web based portal for digital publications, magazines or articles, the most important task is to help the reader effectively navigate through the relevant articles. Having lots of article on the website is not enough if all the contents are not properly sorted or reachable to the users. The difficulty in providing recommendations to the reader is to find articles which are similar in terms of the topics they address to the one being currently read by the user from the huge piles of articles.These recommendations should eventually be verified by a human reader.

To achieve the above requirement, we would be creating topic models from a set of documents. Topic modeling algorithms are statistical methods that analyze the words of the original texts to discover the themes that run through them, how those themes are connected to each other, and how they change over time. There are different types of probabilistic topic modeling algorithms that aim to discover and annotate large archives of documents with thematic information.

Our main goal in this project is to build an efficient recommendation system for digital articles using Latent Dirichlet Allocation algorithm [3] which will provide relevant recommendations to the user for further reading.

2. Prior Work

LDA algorithm that we are using was introduced by Blei, Ng and Jordan [6]. Using this algorithm, each item of a collection is modeled as a finite mixture over an underlying set of topics and in turn, each topic is modeled as an infinite mixture over an underlying set of topic probabilities.

Before LDA, the popular *tf-idf* scheme by Salton and McGill [13] provided a relatively small amount of reduction in description length and reveals little in the way of inter- or intra-document statistical structure. To overcome this shortcoming of *tf-idf*, researchers proposed dimensionality reduction technique of latent semantic indexing (LSI) (Deerwester [12]). Hofmann [7], who presented the probabilistic LSI (pLSI) model, also known as the aspect model, as an alternative to LSI but it is incomplete in that it provides no probabilistic model at the level of documents.

The recommendation systems based on content theme in articles has been researched earlier by different authors.Lang [9] used an approach by combining nearest neighbors and linear regression using a combination of TF-IDF features and features selected with Minimum Description Length (MDL) [1]. Billsus and Pazzani [2] followed another approach by using Nearest Neighbor Classification and Naïve Bayes classification for explicit feedback in an interesting/not interesting classification. Naïve Bayes has also been applied to content-based book recommendation [10]. Another method similar to probabilistic Latent Semantic Indexing [8] has been applied by Cleger-Tamayo et al [5]. Rashid et al. [4] derive information theoretic strategies for how the first recommendations for a new users should be made.

In this project, we have used the Wikipedia dataset to learn and generate a set of cluster of words in english dictionary based on real world occurences using LDA algorithm and then we used consine similarity for checking article closeness. Using these algorithms we have developed an application in the form of article recommendation system using Probabilistic Topic models.

3. Model, assumptions, and requirements

Latent Dirichlet Allocation [6] based Topic Modeling process assumes that

- Each topic k $\in$ {1,....k} is a distribution over words . This distribution comes from a multinomial distribution β_k that itself is drawn from a Dirichlet distribution with parameter λ
- Each document d $\in$ {1,....M} is a distribution over topics . This distribution comes from a multinomial distribution θ_d that itself is drawn from a Dirichlet distribution with parameter α

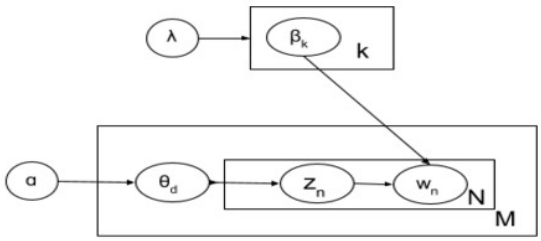


Figure 1. Generative Model Approach

In the LDA topic modeling process, we treat our data arising from a generative process that includes the hidden variables. From the joint probability distribution over both the observed and hidden random variables , we use Gibbs sampling to figure out the latent variables and go from random topics to good topics.

Algorithm 1 Gibbs Sampling process

1: **procedure** GIBBS(D)
for each: $d \in \mathcal{D}$ ▷ D is list of documents
for each: $w \in d$ ▷ d is single document
▷ z is topic allocated to word w in document d
2: $n_{d,z_{old}} = n_{d,z_{old}} - 1$
3: $v_{d,z_{old},w_{d,n}} = v_{d,z_{old},w_{d,n}} - 1$
4: $p(z\_new_{d,n} = k | z_{-d,n},, w, \alpha, \lambda) = \sigma$ ▷ See below
5: $n_{d,z\_new} = n_{d,z\_new} + 1$
6: $v_{d,z\_new,w_{d,n}} = v_{d,z\_new,w_{d,n}} + 1$

where

$$\sigma = \frac{n_{d,k} + \alpha_k}{\sum_{i=1}^{k} n_{d,i} + \alpha_i} * \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_{i=1}^{k} v_{k,i} + \lambda_i}$$

- After iterating through the above algorithm for a large number of times, we will eventually reach a roughly steady state where the assignments are pretty good. So we will use these assignments to estimate the topic mixtures of each document (by counting the proportion of words assigned to each topic within that document) and the words associated to each topic (by counting the proportion of words assigned to each topic overall).

As an outcome of this algorithm we will have a $M \times K$ matrix where M is number of documents and K is number of topics chosen. Each document is a k-dimensional vector with values along each dimension equal to the corresponding topic probability . Now, considering the current document as a point in the K dimensional space, we use the Cosine similarity [11] to find its neighbors and recommend the nearest neighbors based on a threshold distance value.

4. Implementation

We have used the Wikipedia data set for our project. The data set is very large and has a lot of irrelevant information. So for better learning we need to follow few pre-processing steps to prune the irrelevant data. Some of those processes could be

- Removing all the documents with very few number of words(less than 200-300) from the dataset.
- Removing the documents with meta-data
- From the list of frequent words in all documents, we remove the words which are very common or very rare.
- We can further extend the pruning on the above list by word length, stop lists, lemmatization, parts of speech etc.

The total number of documents remaining was 4.2 million . At the end of this stage, 3 files were generated. i) First file consisted of list of words in all documents and their corresponding IDs. ii) Second file consisted of per document bag-of-words list (word_id, word_frequency) of tuples. iii) Third file consisted of term frequency-inverse document frequency (td-idf) numerical statistic for every word in a document.

We have used the above three files and Gensim framework to run LDA using the data in 3 files. We have set the number of topics to be searched for(K) to 100. After the end of the LDA processing we get a binary file consisting of a matrix representing distribution of words over all 100 topics.

From the main wikipedia data set , we chose the indices of the 100 largest documents based on the number of words. For each document, we first calculate the bag of words statistics and feed this value into the learned LDA */

topic model to get the distribution of topics for that document. Each document was represented as a vector of size 100 where each component, v_i takes the value of the probability of topic i in the document. We used a min-heap to keep the 10 most similar neighbors of a particular document based on the cosine similarity index (normalized dot product) between a pair of documents.

We developed a web-based interface to display each of the documents and its 10 nearest neighbors determined from the above step. The main Wikipedia dataset is provided in XML format. We have implemented an extractor to parse all the 100 topics and their references and generate them in a formatted HTML page.

5. Evaluation

As part of the evaluation process we will have to verify that the algorithm produces efficient recommendations. We will learn from Wikipedia data set. We will be testing it on various types of articles. The evaluation will also include testing the correctness of the algorithm with variety of assumption values for the parameters such as value of K (number of topics), α, β.

We will also try to evaluate and plot the change in time complexity with increasing number of topics i.e. increasing number of dimensions.

5.1. Plan

The first step is to review in more detail the Latent Dirichlet Algorithm. The compactness of the huge Wikipedia data set and deciding how much pre-processing is required on the data for the algorithm to work properly. The second step is to implement the algorithm and learning the data set for creation of the topic models. The next step would be to implement the recommendation system. Then the evaluation process to test the correctness and robustness of the system. Finally the preparation of the final paper and presentation.

5.2. Evaluation Results

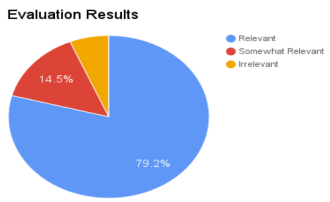
We evaluated the algorithm correctness by randomly choosing 100 documents and providing 10 recommendations for each of these 100 documents.

Below are the execution times required during the process:

- Preprocessing (for 5M documents): $\sim$ 13 hours.
- LDA processing: $\sim$ 12 hours.
- Generating per document topic distribution (for 2M documents): $\sim$ 24 hours.
- Recommendation process: $\sim$ 6 hours

For the evaluation of similarity between the recommended topics, we manually tested and grouped the documents into 3 categories: **Relevant**, **Somewhat Relevant** and **Irrelevant**.

Below is the pie-chart that we observed after our evaluation.



The results of the evaluation can be viewed here: http://spanky.rutgers.edu/reduced_files/

Main document	Reco 1	Reco 2	Reco 3	Reco 4	Reco 5
John French (Army General)	Mahmood Khan (Army General)	Albert Guérisse (Army General)	Terry Peck (Army General)	SS Californian (Steamship)	Étaples mutiny
Economic history of US	Manufacturing in Vietnam	Economy of Malta	Economy of Denmark	Economy of Germany	Legal working age

Figure 2. **Evaluation Phase:** 1) Green - Relevant, 2) Yellow - Somewhat Relevant, 3) Red - Irrelevant

5.3. Discussion of Results

From the above pie-chart of relevance, it is observed that out of 100% documents, approximately 79.2% documents were relevant, 14.5% were somewhat relevant and 6.3% were irrelevant recommendations.

6. Conclusion

After observing our evaluation results, we conclude that LDA topic modelling algorithm works optimally in finding thematic information within the articles. It does a good job in identifying both abstract and general topics in documents and can efficiently identify the dominant theme in documents even when there are irrelevant details pertaining to other topics . Also the cosine similarity index used in our project gives very efficient results(80% accuracy) in identifying the most similar documents based on their underlying themes and hence can be considered a good measure in future documents analysis.

7. Future Work

- Applying our component on a dataset from different article websites such as NY Times and evaluate the results.
- Applying our component on user tweets from twitter to gain thematic information from a tweet.

- Choosing the optimal number of topics (k) through iterations and convergence rather than specifying as a parameter to the algorithm.

- Use clustering on labelled topic datasets for a better validation of our system.

References

[1] J. R. Barron, A.R. and B. Yu. The minimum description length principle in coding and modeling. *Special Commemorative Issue: Information Theory. IEEE Transactions on Information Theory, 44(6)*, 1998. 1

[2] D. Billsus and M. J. Pazzani. User modeling for adaptive news access, 2000. User Modeling and User-Adapted Interaction, 10:147–180. 1

[3] D. M. Blei. "probabilistic topic models." - communications of the acm. 55. 1

[4] N.-A. Y. Blei, D. M. and M. I. Jordan. Latent dirichlet allocation. j, 2003. Journal of Machine Learning Research, 3:993–1022. 1

[5] F.-L. J. M. Cleger-Tamayo, S. and J. F. Huete. Top-n news recommendations in digital newspapers., 2012. . Knowledge-Based Systems, 27(0):180 – 189. 1

[6] A. Y. N. David M. Blei and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(2):993–1022, 2003. 1, 2

[7] T. Hofmann. Probabilistic latent semantic indexing. *Proceedings of the Twenty-Second Annual International SIGIR Conference*. 1

[8] T. Hofmann. Probabilistic latent semantic indexing., 1999. In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '99, pages 50–57, New York, NY, USA. ACM. 1

[9] K. Lang. Newsweeder: Learning to filter netnews, 1995. In Proceedings of the 12th International Machine Learning Conference (ML95). 1

[10] R. J. Mooney and L. . Roy. Content-based book recommending using learning for text categorization., 2000. In Proceedings of the Fifth ACM Conference on Digital Libraries, pages 195–204. ACM Press. 1

[11] G. E. C. r. On measures of entropy and information. 9(3):10, 2016. 2

[12] T. L. G. F. S. Deerwester, S. Dumais and R. Harshman. Indexing by latent semantic analysis. 1

[13] G. Salton and e. M. McGill. *Introduction to Modern Information Retrieval McGraw-Hill*, 1983. 1