LDA on Medical Docket CMS-2021-0168

Sambit Sahoo
University of Maryland - College Park
ssahoo@terpmail.umd.edu

ABSTRACT
Latent Dirichlet Allocation (LDA) is a powerful machine-learning algorithm for the purpose of topic modeling. By extracting the topics from the medical docket, we investigate LDA's ability to separate a document into one of two categories: a comment or an attachment. Additionally, we investigate the optimal amount of topics for the documents in the medical docket by evaluating metric values. Through the process of this algorithm, we work towards refining the algorithm to be as accurate as possible. Additionally, we summarize the challenges of topic modeling and using LDA in this instance.

1. INTRODUCTION

The dataset that we use in this research project is made up of texts parsed from the CMS-2021-0168 medical docket. This docket is about the Omnibus COVID-19 Health Care Staff Vaccination (Centers for Medicare & Medicaid Services, 2021). The texts are either comments or attachments. In each post, a person can submit a public comment and attach supporting articles, letters, research papers, and other types of documents. The words that the person types as part of their submission are comments and the supporting documents are the attachments. Comments tend to be more opinionated and bring up religion and other beliefs. The support documents tend to have more factual backing through surveys, reports, and lab testing. The dataset contains the comments and supporting articles from this docket.

There are two main objectives for this research project. First, we want to determine whether or not an LDA model can distinguish between comments and attachments. The model predictions can be verified since each text is labeled. The second objective is to examine the dataset as a whole and use the topic modeling technique from LDA.

2. BACKGROUND

Natural language processing (NLP) enables computers to extract, understand, and interpret human languages, such as English. By developing algorithms and statistical models, computers can analyze and understand human languages. In this scenario, the models should be able to predict and classify texts and categorize them. There are many different NLP models available to use, but we will focus on the utilization of a Latent Dirichlet Allocation model.

Latent Dirichlet Allocation is a natural language processing model that specializes in topic modeling. It is an unsupervised machine-learning algorithm that is used to recognize words from the topics present in the documents by detecting patterns (Tran, 2022). There are a few essential parameters to account for when using LDA: alpha, beta, k. Alpha accounts for the document-topic density, beta is the word density of a topic, and k is the number of components representing the number of topics clustered into parts in a document. LDA is a generative probabilistic model that relies on Bayes Theorem to find the joint probability (Peddireddi, 2021).

Given a dataset of text, there are six essential steps for topic modeling. First, each string will be processed and tokenized. Using *Word2Vec*, all characters in the string are lowercase. Then, all stopwords and articles are removed from the string. Stopwords are words that are common and do not provide useful information, such as “a”, “an”, “the”, and more. The importance of removing these words is to reduce the amount of noise from the text and increase the accuracy of the model (Ganesan, 2019). Additionally, stemming and lemmatization are used to convert words into their base form. In the case of stemming, the suffixes from words are removed. In the case of lemmatization, it accounts for the context and part of speech of the word before reverting to its root form (Saumya, 2022). Next, the remaining words in the string are tokenized into an array. Tokenization is the process of splitting a string into an array of words.

Second, it is beneficial to make intermediate data structures that will be useful during the Latent Dirichlet Allocation modeling process. The corpus stores an array of tuples that contain the token ID and frequency of the token for each text. The dictionary stores the mapping from word IDs to words, which will help determine vocabulary size, debugging, and topic printing. The document-term matrix is a matrix that describes the frequency of terms that occur in the collection of the documents. Now, a Latent Dirichlet Allocation model can be created. LDA is a statistical classification model that makes observations to understand why parts of the data are similar. Given a set number of topics, LDA maps the words in the document into the topics. Each document is a collection of topics and each topic has a collection of words. The LDA model contains a distribution of words for each topic. Each word in the topic has a defined magnitude based on the value it possesses based on the context of the documents (Angad).

Using the created LDA model, we can begin to visualize the data. The Python package, *pyLDAvis*, has beneficial utilization for data visualization. The visualization contains circles, which are the various topics. The size of the circle indicates the relative statistical weight of topics. There is also a section of the visualization that shows the words associated with the topics (*TMO Guide (3) - pyLDAvis*, 2019). If the topics overlap, the topics may be similar. If there is no intersection, the topics are most likely to be distinct. If a topic is engulfed in another topic, the smaller topic could be a subset of the larger topic.

There are a few measures we can take to test the evaluation of the LDA model. The two common ones are perplexity and topic coherence. Perplexity measures how well the LDA model predicts the test data based on the training data. A lower perplexity score indicates a better predictive performance. In conjunction with the topic coherence, it gives a rough estimate of how well the model is working. Topic coherence measures the interpretability of topics generated by the LDA model. By measuring the degree of semantic similarity between the words with high magnitudes within the topics. A higher coherence score indicates greater topic quality (Giri, 2020). The combination of a low perplexity value and high topic coherence score is an indicator of a well-functioning LDA model.

Since an LDA model has a predefined number of topics, we need to train multiple models with various numbers of topics to find the optimal model. The brute force approach is a solution to finding the optimal LDA model. We can use a loop to create models with an increasing number of topics. Then, we can calculate the coherence score for each model. The model with the highest topic coherence score will be the model we proceed with.

3. METHODOLOGY

We conducted experiments on a dataset that related to the medical docket, CMS-2021-0168. The dataset contained information on both comments and attachments from the docket. The relevant information was the same as the first dataset. It was possible to differentiate between comments and attachments in this dataset by checking the attachment number. If the attachment number was 0, it was a comment, else it was an attachment. We labeled 0 for comments and 1 for attachments.

For the dataset, we began conducting data curation and analysis on it to better understand the data that we have. There were a total of 3496 rows of text. The split is 3215 comments and 281 attachments. Since there was an overwhelming amount of comments, we decided to only include comments that exceeded a character limit of 1500 into the data frame. After this, there were 458 comments and 214 attachments.

In order to create the LDA model, we needed to process and tokenize the text from the data frame. Using helper functions provided in the Jupyter Notebook, we broke down a string of text into one-word tokens. Then, we checked that the token was longer than three characters long and was not a stopword. If it met the previous requirements, it was appended to a list after being stemmed and lemmatized. Once this process was completed on all text in the data frame, we created our intermediate data structures: a dictionary and a corpus.

Using the dictionary and corpus, we created our LDA model. In this instance, we wanted the model to split into two topics, one for comments and one for attachments. Using *pyLDAvis*, we visualized the topics created by the LDA model. The visualization gave us a general understanding of whether the topics were similar or different. If the mapped topics overlap, it is possible that the topics are very similar or potentially one topic is a subset of the other. If the mapped topics have no intersection, it could mean that the topics are distinct. Next, we calculated the log perplexity and topic coherence score of the model. The lower the log perplexity value is, the more certain the model is about the predicted probabilities, meaning that it generalizes testing data well. The higher the topic coherence score is, the higher the degree of semantic similarity between the high-magnitude words within the topics.

Next, we found the dominant topic in each text. For each text, we calculated the topic that had the highest contribution percentage. After checking if it matches the predefined label that we gave each text, we calculated the accuracy of the LDA model by checking whether the dominant topic value matches the label. We repeated this process several times to ensure that the model is working properly rather than getting lucky.

We investigated the optimal amount of topics for the original dataset. Without the character limit, we used the 3496 rows of text. The metrics that we used to define the success of the LDA model were the perplexity and topic coherence scores. We created LDA models with a number of defined topics ranging from 2 to 20.

4. RESULTS

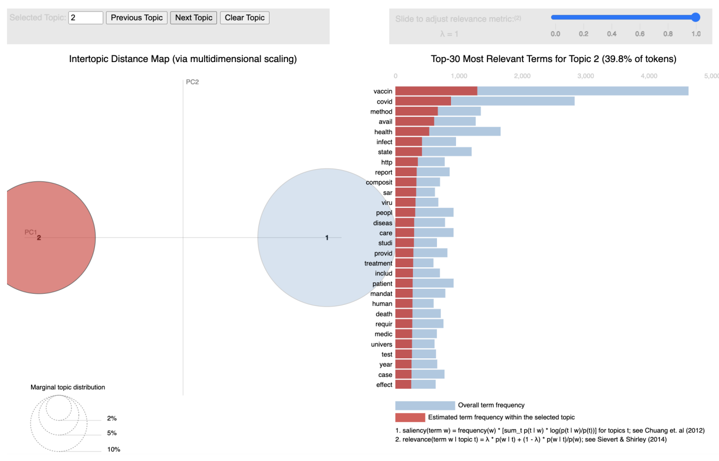
After we finished the data collection, curation, and analysis, we created our preprocessed comments, dictionary, and corpus. With these intermediate data structures, we created an LDA model with two topics. These were the top ten words from each topic:

```
Topic: 0
Words: 0.013*"vaccin" + 0.009*"covid" + 0.007*"method" + 0.006
*"avail" + 0.005*"health" + 0.004*"infect" + 0.004*"state" +
0.004*"http" + 0.003*"report" + 0.003*"composit"

Topic: 1
Words: 0.023*"vaccin" + 0.013*"covid" + 0.008*"health" + 0.005
*"state" + 0.005*"method" + 0.004*"avail" + 0.004*"patient" +
0.004*"care" + 0.004*"peopl" + 0.004*"provid"
```

We deduced that Topic 0 is the topic designated for attachments. The token, “http”, is the foundation of the World Wide Net. It is likely that parts of the attachment might have links to other sites as references and sources.

Additionally, the LDA model was visualized using *pyLDAvis*:


Selected Topic: 2
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:
λ = 1
Intertopic Distance Map (via multidimensional scaling)
Top-30 Most Relevant Terms for Topic 2 (39.8% of tokens)
PC2
PC1
vaccin
covid
method
avail
health
infect
state
http
report
composit
sar
viru
peopl
diseas
care
studi
provid
treatment
includ
patient
mandat
human
death
requir
medic
univers
test
year
case
effect
Marginal topic distribution
2%
5%
10%
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

From this visualization, we found that 60.2% of tokens were in Topic 0 and 39.8% of tokens were in Topic 1. There was a big gap separating the two topics. This indicated that there was a clear distinction between the two topics that the LDA model picked up on.

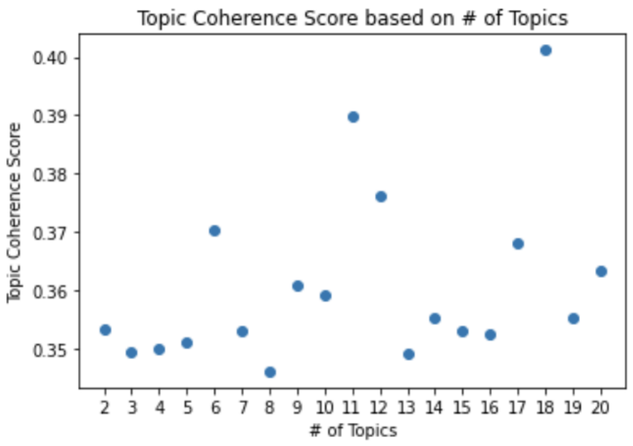
The log perplexity value from the LDA model was low as expected. The score of -8.003 suggested that the LDA model predicted the data well. The topic coherence score was low with a value of 0.336.

Next, we found out which topic was dominant for each text in the data frame and labeled them. Then, we matched the dominant topic label with the predefined label for each text, keeping track of the count for correct and incorrect matches. After dividing the two values by 672, we calculated our correct and incorrect percentages. We repeated this process a total of 5 times:

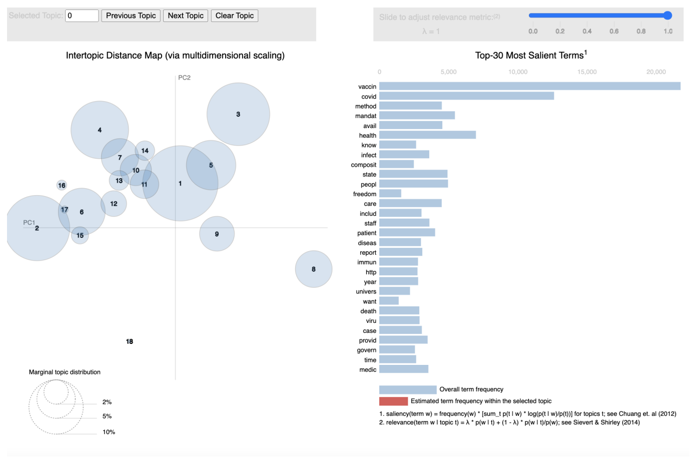
	Correct	% Correct	Incorrect	% Incorrect
Run 1	482	71.73	190	28.27
Run 2	344	51.19	328	48.81
Run 3	481	71.58	191	28.42
Run 4	470	69.94	202	30.06
Run 5	538	80.06	134	19.94
Average	463	68.90	209	31.10

After 5 trials, we conclude that the LDA model with two topics has an accuracy rate of 68.90. This is a fairly accurate model considering the low topic coherence score and the number of topics.

Next, we examined the optimal number of topics for an LDA model given the original dataset. We created LDA models with a various amount of topics. The model with the highest topic coherence score had 18 topics with a coherence score of 0.4012. Additionally, the log perplexity value was This was the scatterplot of the different coherence scores in relation to the number of topics:



This was the *pyLDAvis* visualization for the LDA model with 18 topics:



1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

This visualization showed that most of these topics have some relation to another topic. The only topics that don't overlap with others were topics 8, 9, and 16.

5. DISCUSSION

The research problem of distinguishing texts as either comments or attachments might not be easy to solve using an LDA model. In order to create an optimal LDA model, the perplexity value must be low and the topic coherence score must be high. In this case, the topic coherence score was low, but the number of topics was preset to 2, so it would split on either being a comment or an attachment. Using an LDA model for this problem is probably not the best approach.

There were difficulties with labeling the dataset as a comment or an attachment. Using the *attachment_num* column in the data frame was accurate, but not perfect. In the medical docket, some people would leave a comment saying to look at the attached files. This means that the comment is not relevant to the medical rule change and should not be included as text for the LDA model. Additionally, people would submit their comments as a PDF file, making it classified as an attachment when it should be a comment. Also, there were some attachments that were posters or infographics that could not be easily parsed, so there would be no text pulled from the attachment. It would be difficult to pull a large amount of data with 100% accuracy for labeling because it would be tedious to verify all the comments and attachments.

5. CONCLUSION

LDA modeling has many applications and advantages. Unfortunately, for the objective of this project, it did not meet the expected standard. The accuracy of the model for separating between comments and attachments was 68.9%. This is decent, but the topic coherence scores were regularly low and not a great sign of a well-functioning LDA model.

For future work, we can expand the dataset to account for medical dockets. Using an LDA model, we can investigate whether it can differentiate between the different medical dockets. Also, the current research problem of seeing whether or not the LDA model can understand the difference between a comment and an attachment can be investigated further. By reducing the amount of data that existed in the data frame, the model might not have sufficient data to accurately split the data. By populating the data frame with more attachments, the LDA model could have been more accurate.

6. REFERENCES

Angad. (n.d.). *Angad_TopicModels_Presentation_Slides-062421.pdf*. Nextcloud. Retrieved May 11, 2023, from https://nextcloud.mind.cs.umd.edu/index.php/s/GnLXPs63rSzMQaT

Centers for Medicare & Medicaid Services. (2021). *Regulations.gov*. Www.regulations.gov. https://www.regulations.gov/docket/CMS-2021-0168

Ganesan, K. (2019, April). *What are Stop Words?* Opinosis Analytics. https://www.opinosis-analytics.com/knowledge-base/stop-words-explained/

Giri. (2020, November). *Topic Model Evaluation*. HDS. https://highdemandskills.com/topic-model-evaluation/

Peddireddi, Y. (2021, May). *Topic Modelling in Natural Language Processing*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/05/topic-modelling-in-natural-language-processing/

Saumya. (2022, June). *Stemming vs Lemmatization in NLP: Must-Know Differences*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/06/stemming-vs-lemmatization-in-nlp-must-know-differences/#:~:text=Stemming%20is%20a%20process%20that

TMO Guide (3) – pyLDAvis. (2019, May). WE1S. https://we1s.ucsb.edu/research/we1s-tools-and-software/topic-model-observatory/tmo-guide/tmo-guide-pyldavis/

Tran, K. (2022, July). *pyLDAvis: Topic Modelling Exploration Tool That Every NLP Data Scientist Should Know*. Neptune.ai. https://neptune.ai/blog/pyldavis-topic-modelling-exploration-tool-that-every-nlp-data-scientist-should-know