Yanying Wu
Professor: Dr. Eickhoff
4/12/2021

Report of Design of Organization and Searching Engine of Covid-19 Articles

Introduction

Millions of academic papers get published every year and as a result, the database of academic articles is huge. Therefore, many scientists had a hard time finding useful and relative articles because of the lack of an effective search engine, which potentially leads to the delay of the research process. Since December 2019, Coronavirus has spread around the world and millions of people lost their lives because of the virus. In the COVID-19 database, there are 33,000 academic articles. The need for organization and a good search engine for Covid-19 articles become an urgent need.

Latent Dirichlet Allocation (LDA) is a method in the topic model, which is a statistical model on classifying potential topics among a collection of documents. The reason for using LDA is that, with a large dataset, the LDA algorithm can be used for both dimensionality reduction, which makes data easier to analyze, and classification of topics. Also, LDA is a powerful tool when we perform data visualization. LDA belongs to the BoW method, which means that there is no sequential relationship between words. LDA, an unsupervised learning method in machine learning, contains two important quantities: p(w|t) which is that each document is a multinomial distribution of topics, and p(t|d) which is that each topic is a multinomial distribution of words. During the generative process, p(w|t) and p(t|d) are randomly defined at first, and then after the repetitive processes (iterations) of random assignment of each word for one of the t topics (p'(w|t) = p(w|t) * p(t|d)), the final output, including classification of topics and documents, *etc.*, can be generated.

The BM 25 algorithm, the evaluation of the relevance between the query, list of tokenized search terms, and documents, is used to build the search engine. BM 25 scores' calculation contains three elements: inverse document frequency (IDF), which is used for the weight reduction of common words in all documents, the term frequency, which means higher weight for higher frequency words (up to k), and document weight, which is that shorter documents might have a higher weight. BM25 is used because of the easy implementation in Python and the efficient calculation speed.

Our team organizes the articles by mainly using the LDA method and building the search engine by using the BM25 algorithm with fusion methods. The model needed for the search engine is pre-trained and stored in the tmp file in GitHub. In the following part, more details about methods and algorithms are provided.

Construction Details of the Model and Details' Explanations

There is a total of 33,000 .json files in the database. In the extraction part, since the .json file cannot be directly processed, list (all_papers), a data type used to store multiple collections, is used to store the paper ID, title, authors, abstracts, body text, and body text in each file. In the list,

the replicate papers with the same titles or the same paper ID are removed by using the unique function in toolz library.

For prepossessing the data for LDA and BM25, two corpus dictionaries are built to construct the mapping between the paper ID and title or abstract by using genism and nltk libraries. Inside both dictionaries, the title or the abstract are tokenized, which means all text is split into words with the removal of all punctuations and the lowercase of all words. For improving the accuracy in the LDA and search engine, the words less than two characters or appear only once get removed. Porter steaming is used, meaning that different forms of word contain the same meaning and can be reduced to the stem words. Then, both dictionaries are trained and saved. Following, a bag of words (BoW) corpus is created by passing the tokenized dictionaries (title or abstract and paper ID) to the format of BoW by using the dictionary.doc2bow, an embedding method. The corpus has the ID and frequency of each word in each document. By using the BoW corpus, the model is more easily build by transforming textual information into numerical information. Both BoW corpus is created and saved by sterilizing and pickle function for the comparison of the corpus and query in a later step.

In the organization part, our team uses Latent Dirichlet Allocation (LDA), which function is directly import from the models function in the genism library. 200 iterations are performed for better performance in convergence for our dataset. Our code also uses the LdaMulticore algorism in the genism to parallelize training to improve the speed of the LDA training model and can use a certain amount of memory without expansion. Also, the LDA models are built for both titles' text and abstracts' text separately for a better understanding of the topics' classifications and visualizations.

Furthermore, for the visualization of the LDA data, the pyLDAvis library is used. From the html file gained, the distribution of each topic is shown on the distance map. Each topic shows as a bubble and the sizes of the bubble are relative to the weight of the topic. In some situations, the bubble might overlay with each other, which means the similarity of topics. On the right side, the top 30 salient terms of a specific topic are shown with an adjustable scale bar, which shows a metric for how much information a word provides about a subject.

In the construction of the search engine, the BM 25 scores of both title and abstracts (bm25_ranking_titles and bm25_ranking_abstracts) are calculated separately by importing Gensim. Summarization library with bm25 function and the scores are assorted and arranged from high to low by using NumPy library. Both functions return paper IDs and scores for a query result, and titles for assessment of the accuracy.

To improve the accuracy, the fusion of retrieval model (two BM25 scores) is used. Our team builds fusion functions in two ways: the sum or multiply BM25 scores across title and abstract in the articles. In the multiple functions, if one of the scores returns to 0, the maximum value of the score will be taken instead of multiplication. Next, we use three different query words to gain the result of all models, including the fusion of sum and multiplication model, and the BM25 of title and abstract model independently. In the resulted documents, the title, paper ID, and total scores are printed. The multiplication function is decided to use because, after comparisons, the

multiplication function has the highest matching rate with the Covid-19 collection search engine, and both fusion methods have a much better result, compared to the individual BM25 algorithm of title or abstract.

Further Improvement Discussion

One of the most shortcomings is that the body text did not successfully train the BoW corpus. While running the sterilize function to save the BoW corpus of body text, the random-access memory (Ram) ran out and got killed, which is larger than 8 GB (the memory on JupyterLab). However, the method does not work. advanced computer with more RAM space is needed for training the model for further use in LDA and BM25.

In the organization part, the LDA is used. There are several limitations for LDA. Since LDA uses the model of BoW and did not take the position in a sentence into consideration, and as a result, the decrease of accuracy in matching rate might occur in a longer document, like the code for building the LDA model on abstract. The solution for this problem could be using n-grams in the stemmed text, instead of one-gram.

In the retrieval part, the BM25 is used to calculate the score between the query and extracted information. The fusion of the multiplication of the BM25 model of abstract and title did improve the performance of the output. For pursuing a higher accuracy rate with more relative articles, the coefficient (weight) can be added to the score of the title and abstract of the BM25 model separately. In each article, the BM25 score of title (Xt) and the BM25 score of the abstract (Xa) are different because the lengths of the input are very different. Adding a reasonable coefficient (weight) to both scores (Ct and Ca) and make them equal to each other, as shown: Xt * Ct = Xa *Ca potentially improves the performance of the result. Ct and Ca need to be calculated by using at least three different queries and gain the medium value eventually across all articles. Also, the linear regression could also be runed to find the weight of Ct and Ca.

BM25, as one of the traditional retrieval models, has a disadvantage, which can only handle the case where the Query and Document have overlapping words, but not the semantic relevance among words. One of the potential solutions can be used PageRank algorithm, which is based on the hyperlinks between web pages. PageRank is calculated based on two assumptions: quantity assumption, which is that there will be the more incoming link of website X if X is more important, and quality assumption, which is that the website X is more important if X has higher quality. At first, every web page has the same PageRank value. After each round of calculation, the value will be updated. PageRank is an advanced algorithm in search engines and is used by Google.

Conclusion

In this covid-19 article case, the organization and the visualization of the LDA model are very straightforward and easy to follow. During the construction of the engine, the BM25 model is used to calculate the score of the abstract and title. To improve the performance, the fusion method is used by multiplication of both BM25 scores of abstract and title. With further development, the BoW corpus of body text will be built with the advanced computer. Also, the coefficient, which gives weight to the BM 25 score of the abstract and BM25 score of the title might be needed to be added.