Read Fast and Smart: Spark NLP Application on Wikipedia

GROUP: Wonder 3

Zihe Yang (zy151)
Zhiyu Lin (zl281)
Yijun Gan (yg270)

1. EXECUTIVE SUMMARY

Since Wikipedia does not currently have a recommendation system, we created one using Spark NLP and the English Wikipedia (58 GB). Our project recommends related wiki pages based on user inputs, and summarizes those pages into only a few sentences, so that the user can read fast and smart.

2. INTRODUCTION

Have you encountered this scenario? -- You are interested in a topic, but are unsure what article to read, or that the article is so long that you do not have enough time to read through. Our project is built to solve these problems by applying NLP techniques into English Wikipedia.

To parallelize the large dataset, we use AWS EMR with one master (m5.2xlarge) and 6 datanodes (m5.xlarge). In section 5.1, we made some visualizations and used topic modeling to classify page titles. In section 5.2, we described how the pages recommendation is built. We tried the TF-IDF model and Word2Vec, and computed the cosine similarities between a given keyword and the page contents. We tried model methods and decided that TF-IDF works better because it is easier to implement. In section 5.3, word cloud is used to visualize the most important keywords for the recommended articles. In section 5.4, we described steps to summarize the pages content by building an extractive text summarizer with the text rank algorithm. Libraries such as Pyspark, MLlib, Gensim, NLTK and SparkNLP are used in this project.

In the end, given a topic the user is interested in, for example the “human behavior” topic, we were able to return wiki pages most relevant to it, capture keywords and tags most representative of these pages content, and provide a 5-sentence summary of each page content. In our example, we assumed that pages such as “anthropology” and “autism” would show up higher for high relevance with human behavior and that is indeed what we observed.

3. CODE FILES

- *Similarity_spark.ipynb*
  Using Gensim’s TF-IDF to recommend similar articles done in Spark clusters.
- *Similarity_sparkml.ipynb*
  Using Spark MLlib’s TF-IDF packages for article recommendation, done in Spark clusters.

- *Zihe_EDA.ipynb*
  Exploratory data analysis, including summary statistics and topic modeling application.
- *Zihe_Summary.ipynb*
  WordCloud and text summarization model using the text rank algorithm.
- *Wiki_PageInfo.py*
  To explore the whole dataset data with getting the count of words, lines, the frequent words for each page.
- *Word2Vec_Kmeans.ipynb*
  A word2vec model with K-means clustering on a subset of the data. Not included in the final result because it does not directly help to answer the research question.
- *Wikiw2v.model*
  The word2vec model produced by Word2Vec_Kmeans.ipynb

4. DATA

The dataset is English Wikipedia dumps in XML. It is approximately 14 GB compressed and expands to over 58 GB when decompressed. To extract structured articles from Wikipedia database dump, the open-source Python script WikiExtractor.py is used. Results from extracting are saved in the S3 bucket (s3://zihe-public/articles) with 137 sub-folders and 100 files in each. For each file, there are multiple wiki pages containing their IDs, URLs, titles and text contents. In the end, there are a total 6,075,800 wiki pages used in this project.

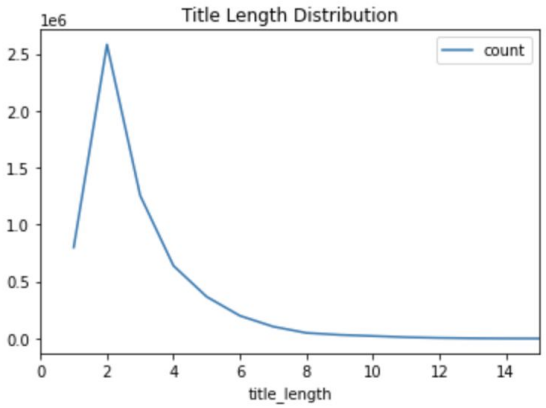
5. METHODS & RESULTS

5.1. Exploratory Data Analysis

Before analyzing contents, we explored the titles of pages first. With the help of AWS EMR, we were able to process this analysis within a few minutes. The longest title we get among all these articles is 43. Through the result below, we could know most pages have 2-word or 3-word titles.

Next, we used topic modeling to conclude all these titles. Results are shown in the script, which is not convincing enough. Titles might not be the good dataset to be modeled with topic modeling.

title_length	count
1	799806
2	2580558
3	1254760
4	639450
5	367086
6	200223
7	105165
8	49728
9	32808
10	22452



5.2. Pages Recommendation by given Topics

Here we will take an example for the following methods. We are interested in the topic “human behavior”. Therefore, we fed this phrase into the TF-IDF model and asked that the model compare each word in this phrase to the entire dictionary, and assign articles similarity scores based on all of the words that it contains. We observe that “human behavior” returned the results shown below. Note that due to credit and time limit, we were only able to run the model on a portion of the dataset.

```
+---+----------------------------------------+----------+
|id |title                                   |similarity|
+---+----------------------------------------+----------+
|20 |Anthropology                            |0.0512    |
|14 |Altruism                                |0.0476    |
|1  |Autism                                  |0.0429    |
|15 |Ayn Rand                                |0.0107    |
|18 |Algeria                                 |0.0084    |
|7  |Aristotle                               |0.0081    |
|22 |Alchemy                                 |0.0068    |
|0  |Anarchism                               |0.0047    |
|19 |List of Atlas Shrugged characters       |0.0042    |
|32 |Andorra                                 |0.0029    |
|28 |Apollo                                  |0.0022    |
|35 |Animal Farm                             |0.0019    |
|6  |Abraham Lincoln                         |0.0019    |
|17 |Allan Dwan                              |0.0       |
|4  |Alabama                                 |0.0       |
|8  |An American in Paris                    |0.0       |
|9  |Academy Award for Best Production Design|0.0       |
|11 |Actrius                                 |0.0       |
|13 |International Atomic Time               |0.0       |
|10 |Academy Awards                          |0.0       |
+---+----------------------------------------+----------+
only showing top 20 rows
```

5.3. Keywords in Recommendations

Given top recommended pages in 5.2, we are going to take all the pages whose similarity rate is higher than 1%, which are "Anthropology", "Altruism", "Autism", "Ayn Rand" these four pages in this case. With WordCloud techniques, we are able to see these keywords in these recommended pages. Top keywords relating to the topic “Human Behavior” are “study”, “social” and “work”.

5.4. Summary of Recommendations

Here we process the text summarization of recommended pages with the help of the Python NLTK library. The idea is to identify the important sentences from the original text and extract only those from the text. Those extracted sentences would be our summary.

First, we preprocessed the data with tokenization, normalization, lemmatization and removing stopwords. Next, we need to convert the text into sentences. Then, to find the frequency of occurrence of each word. Then, it is the time to calculate the scores for each sentence by adding weighted frequencies of the words that occur in that particular sentence. When we have the sentence scores dictionary that contains sentences with their corresponding score, we could take the top 5 sentences with the highest scores as the summary of the page.

Below is the summary of the page - “Autism” with 5 sentences and 188 words. Compared to the original 286 sentences and 6457 words in this page, text summarization saves a bunch of time for us to understand this page.

```
Autism
The terminology of autism can be bewildering, with autism, Asperger syndrome and PDD-NOS often cal
led the "autism spectrum disorders" (ASD) or sometimes the "autistic disorders", whereas autism it
self is often called "autistic disorder", "childhood autism", or "infantile autism".

Of the five PDD forms, Asperger syndrome is closest to autism in signs and likely causes; Rett syn
drome and childhood disintegrative disorder share several signs with autism, but may have unrelate
d causes; PDD not otherwise specified (PDD-NOS; also called "atypical autism") is diagnosed when t
he criteria are not met for a more specific disorder.

Organizations dedicated to promoting awareness of autism include Autism Speaks, Autism National Co
mmittee, and Autism Society of America.

Two are commonly used in autism research: the Autism Diagnostic Interview-Revised (ADI-R) is a sem
istructured parent interview, and the Autism Diagnostic Observation Schedule (ADOS) uses observati
on and interaction with the child.

All known teratogens (agents that cause birth defects) related to the risk of autism appear to act
during the first eight weeks from conception, and though this does not exclude the possibility tha
t autism can be initiated or affected later, there is strong evidence that autism arises very earl
y in development
```

6. CONCLUSION & FUTURE WORK

6.1. Findings and Learnings

Our project started with the idea to build a recommendation system by performing text similarity analysis on a sufficiently large dataset. We learned lots of big data methods and NLP tools during this project, such as how to use pyspark to analyze data and how to use TF-IDF with cosine similarity. We successfully preprocessed the dataset with RDDs and Spark Data Frames, but the runtime of several steps still took too long, such as the text similarity calculations. This project has taught us that we need to make an extra long time budget for data preprocessing in the future, and try to write code with parallelization and pipelines in mind to optimize the computation process.

6.2. Results Validation

Since the objective is to help users read faster and smarter, we did not incorporate supervised learning techniques, so we cannot provide an accuracy number for the recommendation system. Here we manually checked the model results for a few input examples, and have received reasonable and convincing output texts. In the future, we will try to use methods with more accurate validation criteria to solve this problem.

6.3. Future Work

Generating the similarity matrix is very expensive. In the future, we would look for another algorithm alternative. We could also fix this bottleneck by gaining access to more nodes with more memory. This would reduce runtimes and possibly make our results cost efficient.

REFERENCES

1) WikiMedia Data dump torrents.
Available: https://meta.wikimedia.org/wiki/Data_dump_torrents#English_Wikipedia
2) DataBricks spark-xml.
Available: https://github.com/databricks/spark-xml
3) Wikipedia download page.
Available: https://en.m.wikipedia.org/wiki/Wikipedia:Database_download
4) Wikipedia article extraction and cleaning.
Available: https://github.com/attardi/wikiextractor
5) Document recommendation with Spark.
Available:
https://www.kaggle.com/kdziedzic66/simple-solution-using-spark-and-gensim
6) Gensim for text similarity.
Available:https://medium.com/better-programming/introduction-to-gensim-calculating-text-similarity-9e8b55de342d
7) SparkML TFIDF example.
Available:https://medium.com/@rezandry/find-most-relevance-text-data-using-pyspark-with-tf-idf-a4269a13e59

DIVISION OF LABOR

Zihe Yang: Extracted dataset, made exploratory data analysis and built text summarization models.
Zhiyu Lin: Built text similarity models with TF-IDF and Word2Vec, visualized data with Kmeans.
Yijun Gan: Explored the whole dataset data by summarizing word counts, line counts and high frequency words for each page.