Sentiment Analysis on Open Research Data of COVID 19

Xingqi Zhong, Xiaoqing Yang

1 Introduction

For the sake of the great influence of the Covid-19 pandemic, there are thousands of relevant scholarly articles published. Our group members are curious about the attitude of the articles which imply if this pandemic would become better and better. Given such a background, in order to study the attitude of these articles in the process of the pandemic, our group decided to use sentiment analysis methods in this research. We assumed that the positive, negative, and neutral label of the articles titles and body text in the analysis can reflect the attitude of the scholar author toward the pandemic.

Our study uses a freely available dataset on Kaggle offered by the White House and a coalition of leading research groups named the COVID-19 Open Research Dataset (CORD-19). CORD-19 is a resource of over 1,000,000 scholarly articles, including over 350,000 with full text, about COVID-19, SARS-CoV-2, and related coronaviruses. We aimed to utilize this dataset to analyze sentiment of the articles to have an insight into the attitude of the public towards the COVID.

We will display the remainder of our paper in the following structure. Firstly, we will give an explicit explanation of the structure and complexity of the data we used. Then, we will talk about the methodologies we used to process and manipulate the data and the algorithm we used. And finally, we will cover the conclusion we reached and what we find from the result we obtained.

2 Body

We mainly used Spark dataframe and RDDs to store the data and used both Pandas and Spark SQL for data manipulation. In this part, we will elaborate on the technical details, the challenges we faced and how we resolved in the following sections.

Considering the data storage, data importing and code sharing issues, we conducted part of the computation processes on Google Colab. To monitor the time and condition for job execution, we also ran our code on Jupyter notebook in Docker.

We will unfold our research procedure by starting from the data, the algorithms and methods we used, to the empirical results we obtained on both small samples and the whole data set. Finally, we will conclude our findings and demonstrate a brief result analysis to find out the answer to what we proposed in the initial step.

2.1 Data overview

The data we used consists of two parts. The first piece of data we use is a collection of json files of articles, consisting of information including the paperid, title, author, abstract, citations, references, equations, body text, back matter, table or figure captions, etc. The data takes up 26.99 GB space on disk. The schema of the json files is Figure 1 and Figure 2 in Appendix. The second piece of data is a dataset consisting of big picture landscape information of the COVID research data, including paperid, publish time, publish source, doi, title, abstract, etc. This piece of data takes up 1.54GB on disk. We combined and used these data to find out insightful answers to our questions.

The data our group chose is complex in terms of size, file type. On one hand, these two pieces of data have huge sizes, 28.53 GB in total. The size brought a large challenge to us and motivated us to find a more efficient way to manipulate the data. A good and efficient way can not only be time-saving but also reduce space complexity. On the other hand, these data have different file types. The first one is the type of json file with a complex and nested structure as Figure 1 and Figure 2 in Appendix. And the second one is a csv file.

2.2 Data processing & Data manipulation

2.2.1 Importing the first piece of data

When reading the json file, our group has tried several methods to import the data. First of all, we selected some of the information in the json files, like paper_id, body_tect, etc and imported each json file as a dataframe. And then we concatenated them together. However, because of the huge number and size of json files and the limitation of RAM, this method took us a very long time to run and it is not able to read the whole json files. Although we tried to import the data as a large list to shorten the running time, the limitation of RAM prevented us from reading all of the json files at one time. Therefore, we had to find another way. In order to import data more efficiently, we decided to use Spark to read the json files.

We noticed that the records in the json files are in multiple lines and have a nested structure. If we try to simply read it, we would get an undesirable result. If the version of our spark is prior to 2.2, we would

have to use sc.wholeTextFiles() to get a RDD then convert to DataFrame. But since our spark version is 3.2.1, we can simply set the multiLine option to be True and to read multiple lines in json files.

To extract the nested structure of the files and arrange the files into a tidy and organized structure, we used StructField and StructType in PySpark SQL to create string, integer, array and map columns for the dataframe while specifying the column name, data type, nullable or not, etc. to facilitate the downstream processing of the data. The schema of the data is shown in Figure 4 in Appendix.

Then we arranged the files into a spark dataframe by applying union to the current records to the existing dataframe. Through this way, we were able to not only import all the information in all json files, but also the running time decreased a lot compared to the two methods we used before.

2.2.2 Importing the second piece of data

We also used two kinds of methods to import the second piece of data. First, we use pandas to read the csv. File and import it as a dataframe. And then we converted it as a spark dataframe. The second method which we decided to use finally is read the .csv file via spark directly. This way is easier and saves more time.

2.2.3 Data processing and manipulating

After importing the data, we would like to know what the two spark dataframes simply looks like as Figure 5 and some statistics information about it. The first dataframe which collects all the information of the all json files has six columns and 286229 rows as Figure 6.

The second dataframe has 19 columns and 902589 rows. We could find that rows of the second piece of data are much larger than the first one and most of them do not have pmcid. We wanted to focus on the information related to pmc in this dataframe and used the where function to find how many rows have pmcid. We extracted all the rows that have pmcid and then saved them as a new .csv file. We used SQL to filter publish time and count the number of rows . There are 273983 rows. In the following figure, We could have a glimpse of the structure and information as Figure 7.

2.3 Methods

2.3.1 Algorithm overview

In order to analyze the attitude of authors toward the Covid-19, we planned to apply sentiment analysis to the body text and title for the articles. It is because this algorithm could quantify the attitude or emotion of the articles and provide intuitive information for us. Also, we could obtain a compound result which we could use for the comparison of attitudes for each year. That's why we choose this method to analyze the data.

2.3.2 Implementation

In the process of sentiment analysis implementation, we used multiple ways to manipulate our data, like Spark, RDD and so on.

We conducted sentiment analysis on the articles related to Covid-19 via the sentimentIntensityAnalyzer (SIA) function. We extracted the title, body text respectively and paper id from the spark dataframe and converted them into RDD. And then we applied the SIA function to the titles and part of body texts respectively and set thresholds to get labels of articles which imply the attitudes of these articles. As a result, we obtained two dataframe with the labels of the articles. We used a small sample of body text and a whole sample of title to analyze. The specific step is the following.

2.3.2.1 Small sample

Originally, we planned to analyze all the body text. However, an error always occurred for not enough space on the Java heap. We tried to increase the memory limit of the spark driver, as is shown in Figure 8. We also enabled the offHeap option in the setup of SparkSession, so as to enable some operations to not compute in Heap.

Considering the unstable environment and limitation of RAM in Google Colab, we chose a small portion of the data to conduct sentiment analysis on.

We noticed that body text is an array-typed column consisting of paragraphs in the article, with both relevant and irrelevant information including text, citation spans, reference spans, equation spans and sections in each paragraph as shown in Figure 9. Owing to this structure, we used the posexplode function in PySpark SQL to separate the paragraphs of body text into multiple rows in the spark dataframe as Figure 10 in Appendix.

Then we used the window function to collect the text into a list for each article and ordered them by their positions. We aggregated the sentences by articles and joined the sentences by blank space. In the meantime, we also counted the number of words and stored the number into a new column named words as Figure 11 in Appendix.

Then we used a self-defined map function to create a new RDD to get the item in body_text column for each row of the spark dataframe and then collected them into a list. List could decrease the RAM. We used the same technique to get a list of paperid. However, instead of map, we used flatMap this time to make sure the function works even if the record in the paper_id column is null.

Since the lists of paper_id and body_text obtained have a one-to-one correspondence to each other, we used the zip function in pyspark to iterate through the items simultaneously in the two lists. In each loop, we used the polarity_scores function in sia to compute the negative, neutral, positive and compounding score for sentences in the articles. We set a threshold of +-0.05 to assign a label for each article, which represents positive sentiment if label equals 1 and negative sentiment if label equals -1 or neutral if label equals 0.

2.3.2.2 Whole data set

One of the challenges in manipulating data was that the title and the author information were included in the metadata column of the dataframe. We tried several ways and finally we used the spark function to deal with it. We manipulated the data with Spark dataframe and extracted the title from the metadata column of the dataframe from all json files. After we got the titles, we could apply the SIA function to them. The sentiment analysis steps of the whole data set were similar to the small sample dataset.

2.3.3 Empirical Result of Sentiment Analysis

2.3.3.1 Small sample

The results we got from the small sample are shown in Figure 13.

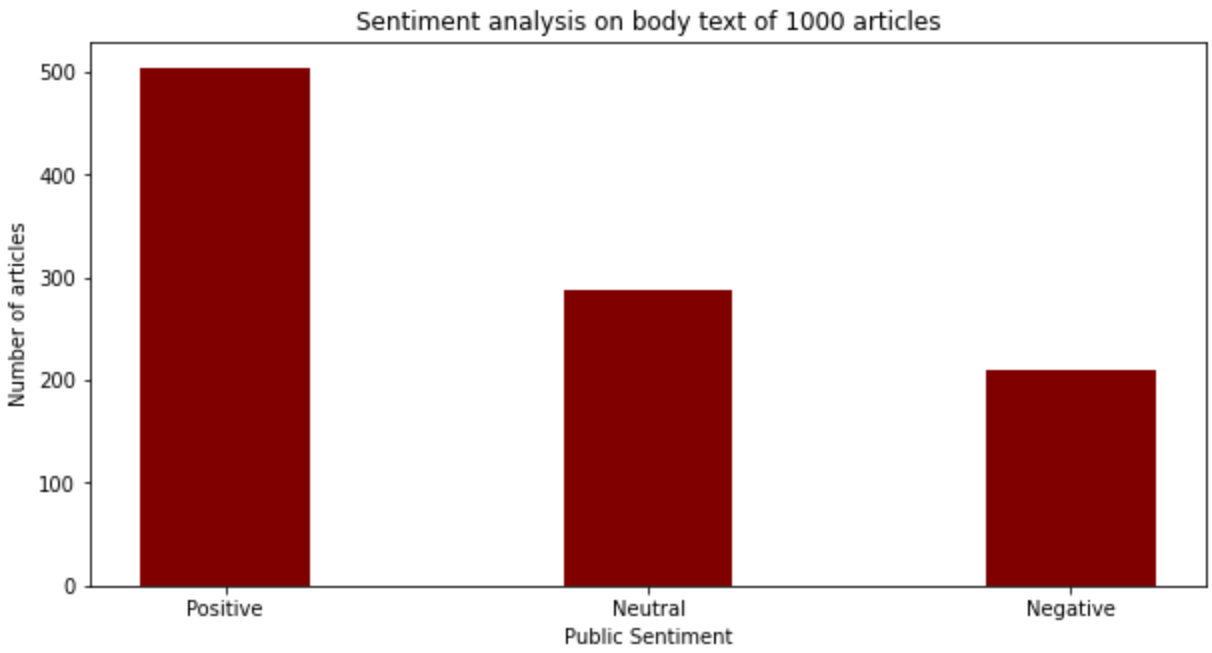

Figure 13

From the results, we can notice that the compounds of most of the articles are extremely high and close to 1 or -1 partly because the articles are too long and contain many sentences. The sentiment analysis combines all attitudes of these sentences. So, although we set a higher threshold, most of the labels are -1 or 1.

2.3.3.2 Whole data set

We counted the labels in the sentiment analysis and got the result as Figure 14 shows.

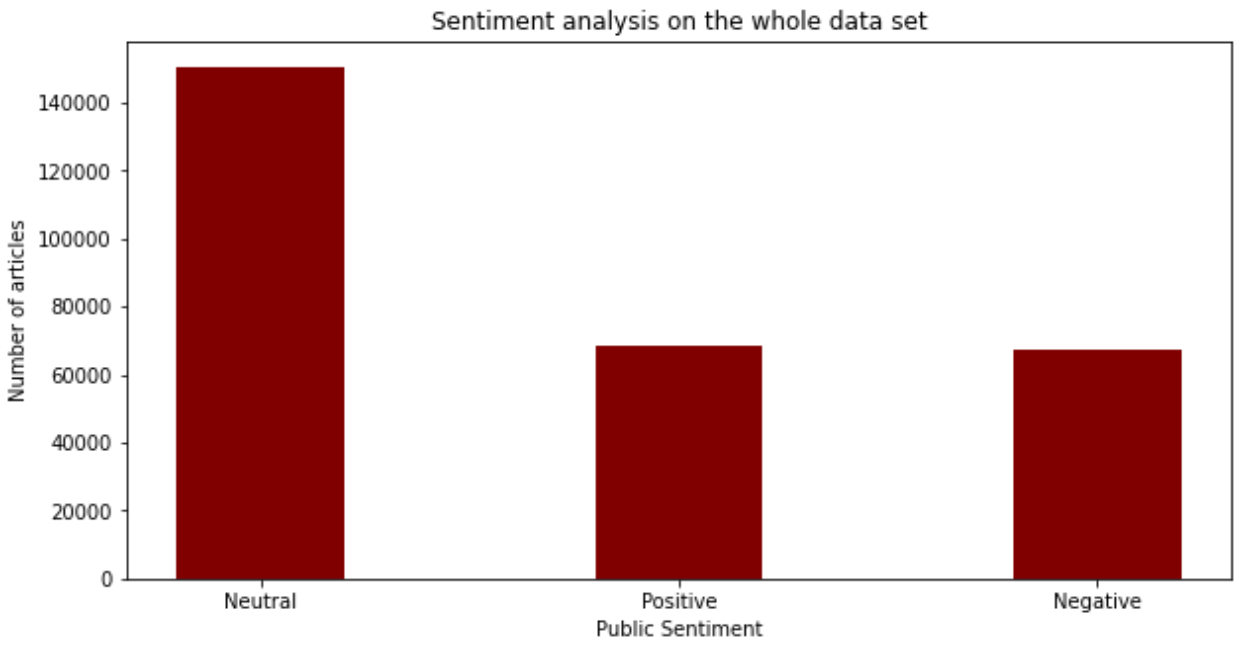

Figure 14

From the result of sentiment analysis, we found that most of the labels are 0. The number of label 0 is up to 150556 and it is more than a half. And the number of the label 1 are close to label -1, which implies that the positive articles and negative articles have similar quantity.

2.4 Empirical Result Analysis

After obtaining the sentiment analysis results, we wanted to go further to get deeper insights into the change of attitude of the public towards COVID. We selected the articles with publish time on or after the outbreak time point of the pandemic and combined the two data sets we have to incorporate the timestamp information into our data. Then we grouped the data according to their published year and calculated the average compounding score for the articles in each group.

We found that in 2020, the average compound is -0.0085. In 2021, the average compound is -0.004. In 2022, the average compound increased to 0.0048. As is shown in Figure 16, the average compounding score for each year is increasing. After researching on the typical thresholds people use and conducting trials by setting different thresholds for our data, we used 0.05 as the classification threshold for positive sentiment and -0.05 for negative sentiment.

Thus, the comparison result we obtained shows that the attitude of the scholars towards the Covid-19 is becoming increasingly positive, which also implies that the improvement of the pandemic on the high level.

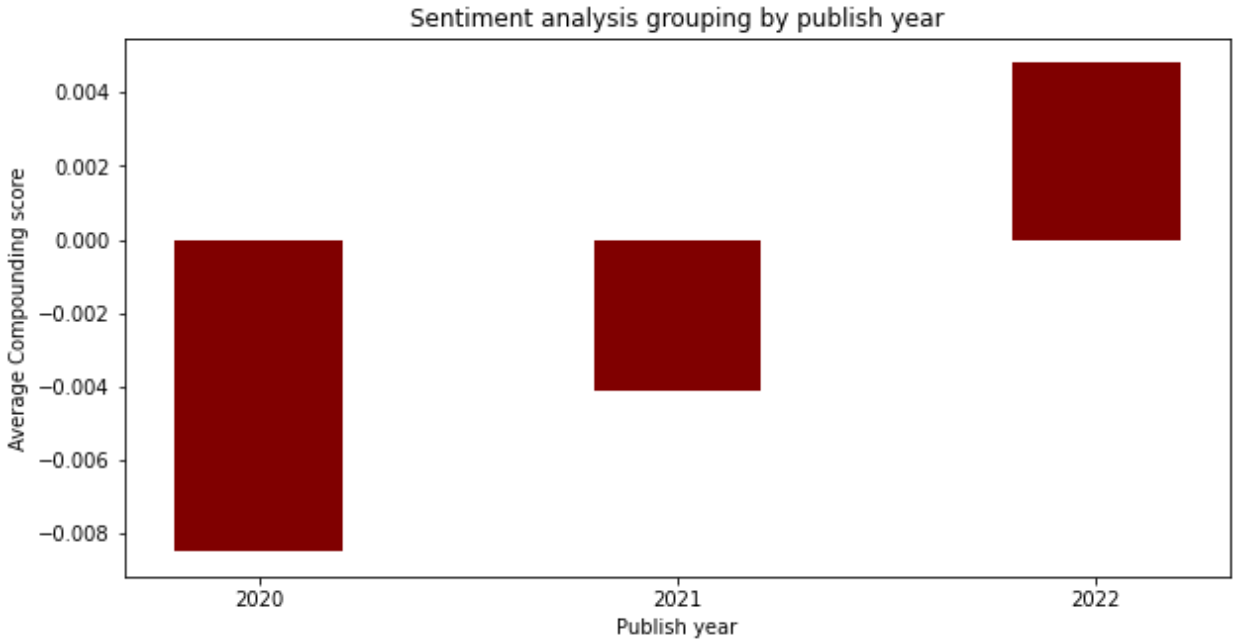

Figure 16

3 Conclusion

By using the methods that have been mentioned in the Body section, we find that for all articles about COVID-19, around 53% of them are neural, while 24% of them are positive and 24% of them are negative. This result shows a relatively objective attitude of the scholars towards COVID-19, which is quite reasonable.

We also analyzed the change of sentiment across time passage and have found the improving trend of the sentiment from 2020 to 2022. This shows that people are becoming more resilient and optimistic, which is good news to learn.

4 Appendix

Data Schema and structure demonstrations

```
# JSON schema of full text documents

{
    "paper_id": <str>,                      # 40-character sha1 of the PDF
    "metadata": {
        "title": <str>,
        "authors": [                        # list of author dicts, in order
            {
                "first": <str>,
                "middle": <list of str>,
                "last": <str>,
                "suffix": <str>,
                "affiliation": <dict>,
                "email": <str>
            },
            ...
        ],
        "abstract": [                       # list of paragraphs in the abstract
            {
                "text": <str>,
                "cite_spans": [             # list of character indices of inline citations
                                            # e.g. citation "[7]" occurs at positions 151-154 in "text"
                                            #      linked to bibliography entry BIBREF3
                    {
                        "start": 151,
                        "end": 154,
                        "text": "[7]",
                        "ref_id": "BIBREF3"
                    },
                    ...
                ],
                "ref_spans": <list of dicts similar to cite_spans>,     # e.g. inline reference to "Table 1"
                "section": "Abstract"
            },
            ...
        ],
        "body_text": [                      # list of paragraphs in full body
                                            # paragraph dicts look the same as above
            {
                "text": <str>,
                "cite_spans": [],
                "ref_spans": [],
                "eq_spans": [],
                "section": "Introduction"
            },
            ...
            {
                ...,
                "section": "Conclusion"
            }
        ],
```

Figure 1

```
        "bib_entries": {
            "BIBREF0": {
                "ref_id": <str>,
                "title": <str>,
                "authors": <list of dict>       # same structure as earlier,
                                                # but without `affiliation` or `email`
                "year": <int>,
                "venue": <str>,
                "volume": <str>,
                "issn": <str>,
                "pages": <str>,
                "other_ids": {
                    "DOI": [
                        <str>
                    ]
                }
            },
            "BIBREF1": {},
            ...
            "BIBREF25": {}
        },
        "ref_entries":
            "FIGREF0": {
                "text": <str>,                  # figure caption text
                "type": "figure"
            },
            ...
            "TABREF13": {
                "text": <str>,                  # table caption text
                "type": "table"
            }
        },
        "back_matter": <list of dict>           # same structure as body_text
    }
}
```

Figure 2

```
root
 |-- paper_id: string (nullable = true)
 |-- metadata: struct (nullable = true)
 |    |-- title: string (nullable = true)
 |    |-- authors: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- first: string (nullable = true)
 |    |    |    |-- middle: array (nullable = true)
 |    |    |    |    |-- element: string (containsNull = true)
 |    |    |    |-- last: string (nullable = true)
 |    |    |    |-- suffix: string (nullable = true)
 |    |    |    |-- affiliation: struct (nullable = true)
 |    |    |    |    |-- laboratory: string (nullable = true)
 |    |    |    |    |-- institution: string (nullable = true)
 |    |    |    |    |-- location: struct (nullable = true)
 |    |    |    |    |    |-- settlement: string (nullable = true)
 |    |    |    |    |    |-- country: string (nullable = true)
 |    |    |    |-- email: string (nullable = true)
 |-- body_text: array (nullable = true)
 |    |-- element: struct (containsNull = true)
 |    |    |-- text: string (nullable = true)
 |    |    |-- cite_spans: array (nullable = true)
 |    |    |    |-- element: struct (containsNull = true)
 |    |    |    |    |-- start: integer (nullable = true)
 |    |    |    |    |-- end: integer (nullable = true)
 |    |    |    |    |-- text: string (nullable = true)
 |    |    |    |    |-- ref_id: string (nullable = true)
 |    |    |-- ref_spans: array (nullable = true)
 |    |    |    |-- element: struct (containsNull = true)
 |    |    |    |    |-- start: integer (nullable = true)
 |    |    |    |    |-- end: integer (nullable = true)
 |    |    |    |    |-- text: string (nullable = true)
 |    |    |    |    |-- ref_id: string (nullable = true)
 |    |    |-- eq_spans: array (nullable = true)
 |    |    |    |-- element: struct (containsNull = true)
```

Figure 4

```
[ ] df.show(3)
```

```
+----------+--------------------+--------------------+--------------------+--------------------+------------+
|  paper_id|            metadata|           body_text|         bib_entries|         ref_entries|back_matter|
+----------+--------------------+--------------------+--------------------+--------------------+------------+
|PMC8206995|{Timing of surger...|[{Patients with p...|{BIBREF0 -> {null...|{TABREF0 -> {Tabl...|         []|
|PMC7111423|{Poster Sessions,...|[{\nJ. Åhman*, E....|{BIBREF0 -> {null...|                  {}|         []|
|PMC7122603|{Cardiovascular A...|[{\nα\n1-adrenoce...|{BIBREF0 -> {null...|                  {}|         []|
+----------+--------------------+--------------------+--------------------+--------------------+------------+
only showing top 3 rows
```

Figure 5

```
df.count()
```

```
286229
```

Figure 6

```
spark.sql("""

select pmcid, publish_time, title, authors, abstract from kag where pmcid like 'PMC%' and publish_time>='2019-12-01'
order by publish_time

""").show()
```

```
+----------+------------+--------------------+--------------------+--------------------+
|     pmcid|publish_time|               title|             authors|            abstract|
+----------+------------+--------------------+--------------------+--------------------+
|PMC7327228|  2019-12-01|Implementing the ...|Seal, Hayley E.; ...|Children with con...|
|PMC8422197|  2019-12-01|Message from Depu...|    Leung, Gabriel M|                null|
|PMC8422199|  2019-12-01|One Hundred Years...|Zhang, Ran; Dong,...|Almost 100 years ...|
```

Figure 7

Spark driver setup

```
from pyspark.conf import SparkConf
conf = SparkConf()
conf.set("spark.driver.memory","15g")
```

```
<pyspark.conf.SparkConf at 0xffff4c457d00>
```

Figure 8

```
|-- body_text: array (nullable = true)
|    |-- element: struct (containsNull = true)
|    |    |-- text: string (nullable = true)
|    |    |-- cite_spans: array (nullable = true)
|    |    |    |-- element: struct (containsNull = true)
|    |    |    |    |-- start: integer (nullable = true)
|    |    |    |    |-- end: integer (nullable = true)
|    |    |    |    |-- text: string (nullable = true)
|    |    |    |    |-- ref_id: string (nullable = true)
|    |    |-- ref_spans: array (nullable = true)
|    |    |    |-- element: struct (containsNull = true)
|    |    |    |    |-- start: integer (nullable = true)
|    |    |    |    |-- end: integer (nullable = true)
|    |    |    |    |-- text: string (nullable = true)
|    |    |    |    |-- ref_id: string (nullable = true)
|    |    |-- eq_spans: array (nullable = true)
|    |    |    |-- element: struct (containsNull = true)
|    |    |    |    |-- start: integer (nullable = true)
|    |    |    |    |-- end: integer (nullable = true)
|    |    |    |    |-- text: string (nullable = true)
|    |    |    |    |-- ref_id: string (nullable = true)
|    |    |-- section: string (nullable = true)
```

Figure 9

```
df.select("paper_id", F.posexplode("body_text").alias("pos", "value")).show(5)
```

```
+----------+---+--------------------+
|  paper_id|pos|               value|
+----------+---+--------------------+
|PMC8206995|  0|{Patients with pe...|
|PMC8206995|  1|{Pre-pandemic stu...|
|PMC8206995|  2|{More granular da...|
|PMC8206995|  3|{This was an inte...|
|PMC8206995|  4|{Participating ho...|
+----------+---+--------------------+
only showing top 5 rows
```

Figure 10

```
body_text.show(n=5)
```

```
+----------+--------------------+-----+
|  paper_id|           body_text|words|
+----------+--------------------+-----+
|PMC1110908|Severe acute resp...| 6129|
|PMC1215526|Virus-host intera...|12937|
|PMC1481559|The control of di...| 4380|
|PMC1609170|SARS is a new inf...| 4497|
|PMC1852297|Hepatitis C virus...| 3254|
+----------+--------------------+-----+
only showing top 5 rows
```

Figure 11

```
df_result
```

	neg	neu	pos	compound	body_text	paper_id	label
0	0.017	0.922	0.061	0.9997	Severe acute respiratory syndrome (SARS) first...	PMC7152143	1.0
1	0.053	0.893	0.054	-0.9862	Virus-host interactions are crucial for the ou...	PMC7130089	0.0
2	0.036	0.897	0.066	0.9992	The control of diseases associated with highly...	PMC8652887	1.0
3	0.020	0.857	0.124	0.9999	SARS is a new infectious disease that emerged ...	PMC7995808	1.0
4	0.043	0.872	0.085	0.9991	Hepatitis C virus (HCV, a member of the Flaviv...	PMC7158344	1.0
...	...	...	...	...	...	...	...
995	0.036	0.921	0.044	0.9615	Acute appendicitis (AA) is the most common gen...	PMC7978675	0.0
996	0.090	0.818	0.092	0.9258	The emergence of novel coronavirus disease 201...	PMC6357155	0.0
997	0.111	0.857	0.033	-0.9998	In December 2019, an outbreak of a novel coron...	PMC8307643	-1.0
998	0.029	0.889	0.083	0.9985	The World Health Organization (WHO) has been e...	PMC8398366	1.0
999	0.028	0.927	0.045	0.9992	Design automation and safety of autonomous sys...	PMC4252826	1.0

1000 rows × 7 columns

Figure 12

Spark SQL manipulation

We created temporary tables for the two datasets and respectively joined them with the second piece of dataframe which include publish time on the common column, i.e.paperid and pmcid, using Spark SQL. We used Spark dataframe to store the data in this step.

```
spark.sql("""
with merged_df as (
    select pmcid, publish_time, title as md_title, authors, abstract
    from kag
    where pmcid like 'PMC%' and publish_time>='2019-12-01'
    order by kag.publish_time),

 df as (
  select pmcid, publish_time, md_title, authors, abstract, neg, neu, pos, compound, label
from merged_df
INNER JOIN result ON merged_df.pmcid=result.paper_id)

select  pmcid, EXTRACT(YEAR FROM publish_time) as publish_year, md_title, authors, abstract, neg, neu, pos, compound, label
from df

""").show()
```

```
+----------+------------+--------------------+--------------------+--------------------+-----+-----+-----+--------+-----+
|     pmcid|publish_year|            md_title|             authors|            abstract|  neg|  neu|  pos|compound|label|
+----------+------------+--------------------+--------------------+--------------------+-----+-----+-----+--------+-----+
|PMC6874235|        2019|Middle East Respi...|Zheng, Jian; Hass...|A high percentage...|  0.0|  1.0|  0.0|     0.0|  0.0|
|PMC6890879|        2019|Evaluation of ant...|Peng, Ju-Yi; Horn...|Bacillus lichenif...|  0.0|  1.0|  0.0|     0.0|  0.0|
|PMC6898929|        2019|Risk factors for ...|Gupta, Ena; Hosse...|BACKGROUND: Clini...|0.244|0.756|  0.0| -0.6705| -1.0|
|PMC6899506|        2019|A systematic revi...|Dighe, Amy; Jomba...|Human infection w...|  0.0|  1.0|  0.0|     0.0|  0.0|
|PMC6901795|        2019|Zebrafish TRIM25 ...|Jin, Yilin; Jia, ...|RIG-I-like recept...|  0.0|0.796|0.204|  0.5574|  1.0|
|PMC6902615|        2019|Spontaneous breat...|Xia, Jingen; Gu, ...|BACKGROUND: The u...|0.316|0.684|  0.0| -0.7184| -1.0|
|PMC6905523|        2019|IL-4/IL-13 polari...|Rogers, Kai J.; B...|BACKGROUND: Ebola...|  0.0|  1.0|  0.0|     0.0|  0.0|
|PMC6909074|        2019|Effect of carboni...|Esfandiari, Neda;...|BACKGROUND: Prist...|  0.0|  1.0|  0.0|     0.0|  0.0|
|PMC6910952|        2019|Structural insigh...|Sato, Yusuke; Tsu...|Npl4 is likely to...|  0.0|  1.0|  0.0|     0.0|  0.0|
|PMC6912106|        2019|Recombinant Rotav...|Papa, Guido; Vend...|Rotavirus (RV) re...|  0.0|0.859|0.141|  0.4215|  1.0|
|PMC6917592|        2019|Interferon-Indepe...|Ashley, Caroline ...|The critical role...|  0.0|  1.0|  0.0|     0.0|  0.0|
|PMC6924143|        2019|Prediction of nov...|Khanna, Varun; Li...|BACKGROUND: Toll-...|  0.0|0.796|0.204|  0.3182|  1.0|
|PMC6925858|        2019|A review on the e...|Matei, Ioana A.; ...|Anaplasma phagocy...|  0.0|  1.0|  0.0|     0.0|  0.0|
|PMC6928167|        2019|Herpes simplex vi...|Hraiech, Sami; Bo...|BACKGROUND: Herpe...|0.178|0.822|  0.0| -0.3818| -1.0|
|PMC6934550|        2019|In Vivo Activity ...|DeWald, Lisa Evan...|During the Ebola ...|  0.0|  1.0|  0.0|     0.0|  0.0|
|PMC6934710|        2019|A natural polymor...|Ávila-Pérez, Gine...|Zika virus (ZIKV)...|  0.0|0.676|0.324|  0.5859|  1.0|
|PMC6935106|        2019|Prevalence and ph...|Zhang, Fanfan; Lu...|BACKGROUND: In Ch...|  0.0|  1.0|  0.0|     0.0|  0.0|
|PMC6936066|        2019|Roles of transfor...|Song, Dongli; Tan...|BACKGROUND: Teloc...|  0.0| 0.86| 0.14|  0.3818|  1.0|
|PMC6939335|        2020|Dexmedetomidine i...|Nakashima, Tsuyos...|BACKGROUND: Dexme...|0.132|0.711|0.157|   0.128|  1.0|
|PMC6941262|        2020|Whole genome sequ...|Kamau, Everlyn; O...|BACKGROUND: Human...|0.145|0.855|  0.0|  -0.296| -1.0|
+----------+------------+--------------------+--------------------+--------------------+-----+-----+-----+--------+-----+
only showing top 20 rows
```

Figure 15