Quick Topic Modeling with Spark NLP and Colab

Thomas Jaensch
3 min read · Mar 13, 2023

> *How to quickly get a sense of topics discussed in a large corpus of New York Times article comments with a few lines of Spark NLP code in a Colab notebook*

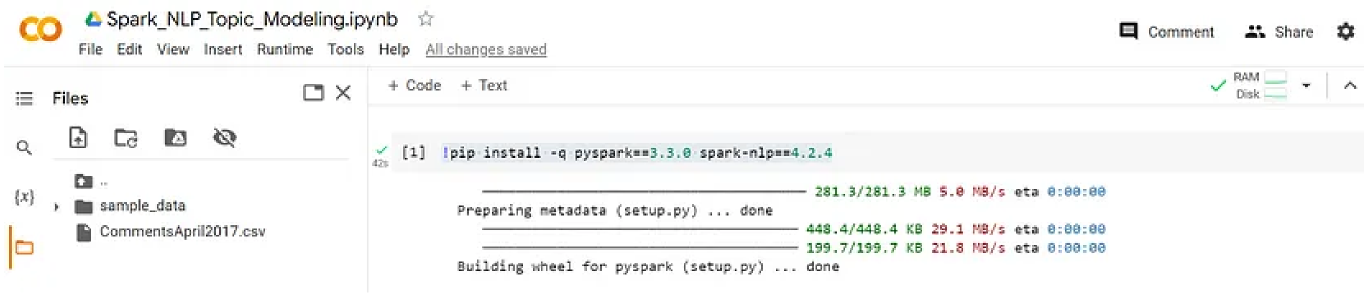
Art my preschool daughter created

Complete notebook code can be found here.

I had a similar use case for a work project and experimented with Databricks widgets to be able to analyze many source documents and a customizable number of topics in one Databricks notebook, but that setup requires access to a Databricks workspace, ML cluster setup, reading files into DBFS, etc. Too much work for a quick demo. Pretty much the same thing can be achieved easily in Colab, and for simplicity's sake with just one data file (in this case the "CommentsApril2017.csv" from this Kaggle dataset).

Let's get started.

Go to https://colab.research.google.com/, create a new notebook, and upload the "CommentsApril2017.csv" file to the Colab session storage (on the left below).


Spark_NLP_Topic_Modeling.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
Files
+ Code + Text
RAM
Disk
sample_data
CommentsApril2017.csv

```
[1] !pip install -q pyspark==3.3.0 spark-nlp==4.2.4
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 281.3/281.3 MB 5.0 MB/s eta 0:00:00
    Preparing metadata (setup.py) ... done
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 448.4/448.4 KB 29.1 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 199.7/199.7 KB 21.8 MB/s eta 0:00:00
    Building wheel for pyspark (setup.py) ... done
```

In the first notebook cell, install PySpark and Spark NLP:

```
!pip install -q pyspark==3.3.0 spark-nlp==4.2.4
```

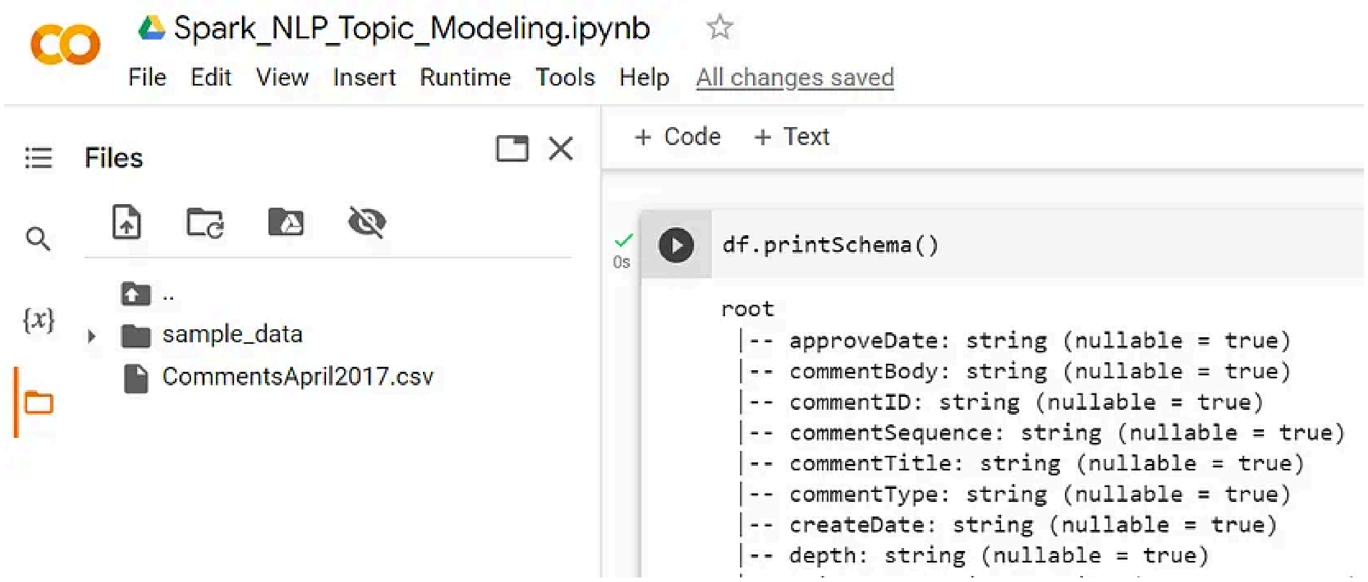
Start the Spark session and import required libraries:

```
from sparknlp.base import *
from sparknlp.annotator import *
from sparknlp.pretrained import PretrainedPipeline
import sparknlp
from pyspark.sql import SparkSession
from pyspark.ml import Pipeline

spark = sparknlp.start()
```

Read the data file into a Spark dataframe and count the number of comments:

```
df = spark.read.format("csv").option("header",True).load("/content/CommentsApril
df.count()
```

When I wrote this, I got 243866 comments for this file. Next, take a look at the schema of the dataframe. There are many columns, but essentially we're only interested in "commentBody" for this NLP task.


Spark_NLP_Topic_Modeling.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Files
+ Code + Text
sample_data
CommentsApril2017.csv
0s

```
df.printSchema()
```

```
root
 |-- approveDate: string (nullable = true)
 |-- commentBody: string (nullable = true)
 |-- commentID: string (nullable = true)
 |-- commentSequence: string (nullable = true)
 |-- commentTitle: string (nullable = true)
 |-- commentType: string (nullable = true)
 |-- createDate: string (nullable = true)
 |-- depth: string (nullable = true)
```

Now we are going to build a Spark NLP pipeline with the goal to fit the dataframe to a Spark NLP pipeline model and extract the tokens that will later on be vectorized and then fed to an unsupervised clustering algorithm to extract relevant topics from the data. Feed "commentBody" into DocumentAssembler below and build the following pipeline:

```
document_assembler = DocumentAssembler() \
    .setInputCol("commentBody") \
    .setOutputCol("document") \
    .setCleanupMode("shrink")
```

```
tokenizer = Tokenizer() \
  .setInputCols(["document"]) \
  .setOutputCol("token")

normalizer = Normalizer() \
    .setInputCols(["token"]) \
    .setOutputCol("normalized")

stopwords_cleaner = StopWordsCleaner()\
      .setInputCols("normalized")\
      .setOutputCol("cleanTokens")\
      .setCaseSensitive(False)

finisher = Finisher() \
    .setInputCols(["cleanTokens"]) \
    .setOutputCols(["tokens"]) \
    .setOutputAsArray(True) \
    .setCleanAnnotations(False)

nlp_pipeline = Pipeline(
    stages=[document_assembler,
            tokenizer,
            normalizer,
            stopwords_cleaner,
            finisher])

nlp_model = nlp_pipeline.fit(df)

processed_df  = nlp_model.transform(df)

tokens_df = processed_df.select('tokens').limit(10000)
```

Convert the tokens to vectors of token counts:

```
from pyspark.ml.feature import CountVectorizer

cv = CountVectorizer(inputCol="tokens", outputCol="features", vocabSize=1000, mi

cv_model = cv.fit(tokens_df)
```

```
vectorized_tokens = cv_model.transform(tokens_df)
```

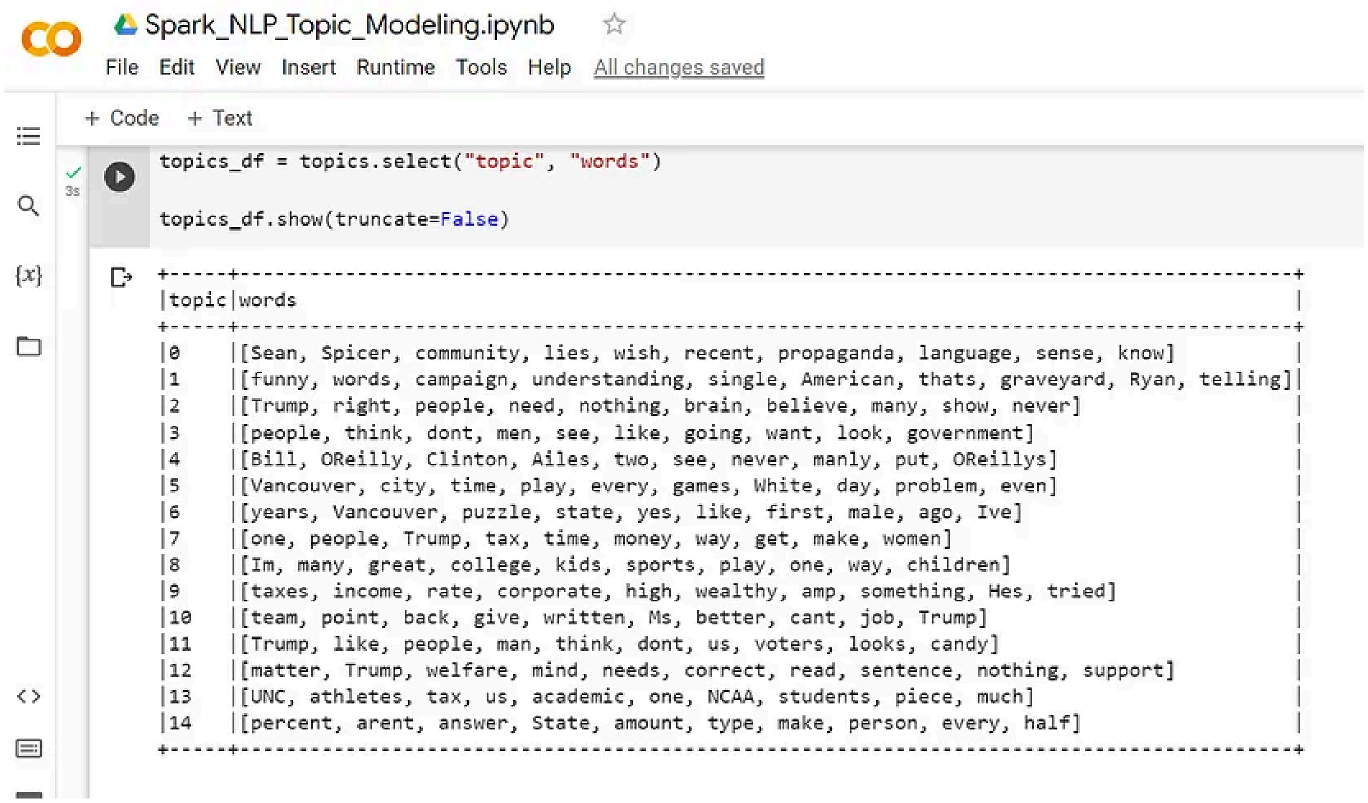
Build the LDA (Latent Dirichlet Allocation) model to generate clusters of a selected number of topics:

```
from pyspark.ml.clustering import LDA

num_topics = 15
lda = LDA(k=num_topics, maxIter=10)
model = lda.fit(vectorized_tokens)
```

Finally, get a dataframe with the extracted topics:

```
from pyspark.sql.functions import udf
from pyspark.sql.types import ArrayType, StringType

vocab = cv_model.vocabulary
topics = model.describeTopics()
topics_rdd = topics.rdd
topics_words = topics_rdd \
       .map(lambda row: row['termIndices']) \
       .map(lambda idx_list: [vocab[idx] for idx in idx_list]) \
       .collect()

def get_words(idx_list):
    return [vocab[idx] for idx in idx_list]

udf_get_words = udf(get_words, ArrayType(StringType()))
topics = topics.withColumn("words", udf_get_words(topics.termIndices))

topics_df = topics.select("topic", "words")

topics_df.show(truncate=False)
```

At the time of writing, this is what I got:


Spark_NLP_Topic_Modeling.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text

```
topics_df = topics.select("topic", "words")

topics_df.show(truncate=False)
```

```
+-----+--------------------------------------------------------------------------------------+
|topic|words                                                                                 |
+-----+--------------------------------------------------------------------------------------+
|0    |[Sean, Spicer, community, lies, wish, recent, propaganda, language, sense, know]      |
|1    |[funny, words, campaign, understanding, single, American, thats, graveyard, Ryan, telling]|
|2    |[Trump, right, people, need, nothing, brain, believe, many, show, never]              |
|3    |[people, think, dont, men, see, like, going, want, look, government]                  |
|4    |[Bill, OReilly, Clinton, Ailes, two, see, never, manly, put, OReillys]                |
|5    |[Vancouver, city, time, play, every, games, White, day, problem, even]                |
|6    |[years, Vancouver, puzzle, state, yes, like, first, male, ago, Ive]                   |
|7    |[one, people, Trump, tax, time, money, way, get, make, women]                         |
|8    |[Im, many, great, college, kids, sports, play, one, way, children]                    |
|9    |[taxes, income, rate, corporate, high, wealthy, amp, something, Hes, tried]           |
|10   |[team, point, back, give, written, Ms, better, cant, job, Trump]                      |
|11   |[Trump, like, people, man, think, dont, us, voters, looks, candy]                     |
|12   |[matter, Trump, welfare, mind, needs, correct, read, sentence, nothing, support]      |
|13   |[UNC, athletes, tax, us, academic, one, NCAA, students, piece, much]                  |
|14   |[percent, arent, answer, State, amount, type, make, person, every, half]              |
+-----+--------------------------------------------------------------------------------------+
```

Remember Sean Spicer?

NLP Machine Learning Topic Modeling Data Science Clustering