Project 1: Big Data Programming Paradigms

Cloud computing and Big Data represent technically different terms, but they are often seen together because of the strong interaction between them. While Big Data simply refers to the capacity to deal with a large amount of data using parallel paradigms, cloud computing usually refers to the processing of anything, including Big Data programs. The cloud, however, provides the compute and storage resources needed to process large amounts of data using parallel paradigms. The cloud provides access to computational resources previously unavailable to many organizations.

In this project, students will get experience with MapReduce, one of the earliest Big Data frameworks that have been adopted for use in cloud systems, and with Spark, one of the most popular successor alternatives to date.

Updates

- 2022-01-18: Changed Task 1 to output the popularity (number of tweets) of each tag instead of its frequency (fraction of tweets). Solutions computing hashtag frequencies will be evaluated with extra 10% credit.
- 2022-01-18: Lifted requirements on printing the CSV from Task 1 sorting by tag popularity. Printing tags in any order will be accepted. Solutions turning in CSV files sorted by popularity will be evaluated with extra 3% credit.

Introduction

In recent years, microblogging and social media platforms such as Twitter and Facebook have become very popular and become a key communication mechanism on the World Wide Web. Every day, users of services like Twitter are capable of generating hundreds of millions of tweets. The analysis of this content is interesting for a large number of applications. The sum of all texts from services like Twitter (tweets) can collectively be considered a source of information about opinions and feelings about products, politics, society and events. As Twitter is currently the microblogging platform with the highest number of active users, several studies have been done using data from the platform. More recently, Twitter has been widely used as a tool for disseminating and collecting information about the Coronavirus (SARS-CoV-2) and the disease it causes (COVID-19). In this sense, several initiatives are seeking to use tweets around the world to try to better understand some aspects of this pandemic.

Dataset Description

The dataset, tools, and frameworks needed to complete the assignment are already installed in the cluster.

The dataset is in HDFS at `hdfs://vcm-23691:9000/datasets/covid19/`. All students have read access to the dataset; you *must not* duplicate the dataset while completing the assignment. Students should use their storage space in HDFS at `hdfs://vcm-23691:9000/user/<netid>` to store the results of different tasks.

The database consists of public tweets extracted between 03/29/2020 and 04/30/2020 from hashtags related to Covid-19 (*#coronavirus, #coronavirusoutbreak, #covid19, #covid_19, #coronavirusPandemic,*

#ihavecorona, #StayHomeStaySafe and *#TestTraceIsolate*). This database is available on the Kaggle platform and consists of approximately 15 million tweets, totaling 7.9 GB.

The database, in JSON files, was processed and contains some fields of the traditional format of a tweet from Twitter's API. More specifically, each JSON file contains one JSON object per line, and each JSON object contains the following fields:

- `status_id`: The identifier of the tweet.
- `user_id`: The identifier of the user.
- `created_at`: The timestamp indicating when the tweet was created.
- `screen_name`: The screen name of the user (e.g., kateperry).
- `text`: The text in the tweet.
- `source`: The application or platform used to create the tweet.
- `reply_to_status_id`: The tweet being replied to.
- `reply_to_user_id`: The user being replied to.
- `reply_to_screen_name`: The screen name of the user being replied to.
- `is_quote`: Whether this tweet is a quote.
- `is_retweet`: Boolean indicating if the tweet is a retweet.
- `favourites_count`: The number of times a tweet was favourited.
- `retweet_count`: The number of times the tweet was retweeted.
- `country_code`: The country code of the user making the tweet.
- `place_full_name`: Name of the place where the tweet was made; depends on location information and may not be available on all tweets.
- `place_type`: Information about the place where the tweet was made.
- `followers_count`: Number of followers of the user.
- `friends_count`: Number of friends of the user.
- `account_lang`: The language used by the user's account.
- `account_created_at`: Timestamp indicating when the user's account was created.
- `verified`: Whether the account is verified.
- `lang`: The language of the tweet.

It is common in scenarios with large datasets to have missing or corrupt elements, internal inconsistensies, and generally poor formatting. The dataset in this exercise also has these problems. For example, because tweets can come from different `source`s like the Twitter Web App, Twitter for Android or even collector services, many fields may have been rendered with empty information. Familiarize yourself with the dataset, look for common errors or anomalies, verify your assumptions, and try try to handle the malformed data items while in your analysis to make your results more faithful to reality. Review your answers carefully.

Tasks

The use of hashtags (`#`) on Twitter allows followers to post discussions around specific topics, including public health topics or events. In this project, you are tasked with answering some questions about events related to Coronavirus during the pandemic using a Big Data environment. More specifically, you will perform the following analyses.

1. Hashtag Popularities and Relations

Using MapReduce, in a single application, compute the popularity of each hashtag *h* and create a list of its correlated hashtags. The popularity is simply the number of tweets in which a hashtag *h* appears. A hashtag *c* is correlated with another hashtag *h* if *c* and *h* appear in the same tweet. Prepare a CSV file where the first column is a hashtag, the second column is the popularity of the hashtag, and the last column represents a list of all hashtags correlated with *h*. Sort tags in decreasing popularity, and break ties by sorting tags lexicographically.

> Study the WordCount example in the `cluster-warmup` document. A key aspect of this task is splitting the computation into the Map and Reduce instances. Think about how Map instances will receive data from HDFS, what they need to compute, and what they will send to Reduce instances. Complementarily, think about what information Reduce instances need from Map instances to compute the result.

> This task will be implemented in Java for submission in Hadoop. Because the input is in JSON format, we strongly suggest you use a JSON parser to interpret each record in the Map stage into a Java `Object`. This can be achieved, for instance, by using the org.json package. Download the jar in the Maven repository and add it as a dependency when compiling the application, using a similar command:
>
> ```
> export HADOOP_CP=$(hadoop classpath)
> javac -cp "$HADOOP_CP:./json-20211205.jar" \
>   -d classesDirectory YourClassName.java
> jar -cvf YourJarName.jar -C classesDirectory/ .
> ```

> Output files generated by Hadoop have `tab`-separated fields by default. We can change this behavior changing the `apred.textoutputformat.separator` configuration parameter. You can do this inside your program through the Configuration instance:
>
> ```
> Configuration conf = getConf();
> conf.set("mapred.textoutputformat.separator", ",");
> ```
>
> Or you can also set the configuration as an environment variable when submitting your job to Hadoop:
>
> ```
> hadoop jar <file.jar> <module> -D
> mapred.textoutputformat.separator="," <params>
> ```

1.1. Word Cloud in Popular Hashtags

Using the CSV above, identify the 50 most popular hashtags, then show these hashtags in a word cloud chart.

Prepare a PNG or PDF file containing the word cloud. Set the size of words in the cloud proportional to their popularity in the tweets.

> For tasks that only require graphing results, you are welcome to use Jupyter notebooks or another framework of your choice. Jupyter is one of the most popular web-based interactive development environments. The basics to get started are shown in this tutorial. The `cluster-warmup` document provices information about how to use Jupyter on the cluster.

> The word cloud is a popular text analysis tool that provides a visualization of the frequency of words in the source text, making the most frequently occurring words more prominent. For those who will use the Jupyter notebook/Python language, a suggestion for creating the word cloud from a frequency list is the WordCloud library, which is already installed on the cluster.

1.2. Related Hashtags

The relationship between hashtags can be seen as a graph where the vertices are the hashtags and the edges are the presence of relationships between them. The generation of a histogram of degrees can help researchers in the study of subject topics. (The degree of a vertex is the number of edges connected to it.) Vertices with high degrees (that is, with many related hashtags) indicate that the topics discussed on Twitter tend to be more general, while vertices with low degrees can indicate isolated or specific subjects.

Consider each hashtag as a vertex in the graph, and correlations computed in Task 1 as edges. Create a histogram graph that expresses the distribution of hashtag degrees.

> You can also use Jupyter or other frameworks for this task. If using Python, the matplotlib library supports building rich graphs. This page provides instructions on how to plot bar graphs.

2. User Popularity Increase

There are many celebrities and telecommunications services that use Twitter as a way to spread their information more quickly and attract more followers. Twitter currently uses a verification seal to validate the authenticity of these profiles. In this exercise, we'll use the dataset to evaluate if verified users have benefited from the pandemic to increase their number of followers.

In this sense, using Spark, identify the 1000 verified users (marked in column `verified`) that grew the most in the number of followers in the period covered by the dataset. Among these users, are there users who are also in the top 1000 most active users, that is, the 1000 users with the highest number of tweets?

Prepare a CSV file with three columns: a user's `screen_name`, that user's *relative* increase in the number of followers during the period, and a boolean (either `0` or `1`) indicating whether that user is in the top 1000 most active users. Use the data in the CSV file to write a short paragraph discussing the evolution of the follower base of verified users during the pandemic.

> Because the data is well structured, Spark's abstraction of DataFrame is a good and a well-documented interface, with a set of operators to handle Big Data in Spark. The DataFrame abstraction offers operators like `read.json`, `filter`, `groupby`, and `join` that can be used in this exercise. Spark parallelizes the execution of these operators, making computation significantly faster than on a single-thread program. This tutorial provides an introduction to PySpark DataFrames and contains several examples.

3. Temporal Analysis

The moving average, also called rolling average, has been one of the main indicators used in the pandemic for several metrics, for example, the number of new cases per week and hospital occupation.

Define a metric to capture one aspect of the pandemic. Calculate in Spark how the metric evolved over the weeks in the United States, the country most affected until November 2020.

> You are free to define any metric of your choice. For example, one example metric is to estimate the trend in number of cases by tracking the number of *users* (not tweets) self-reporting they are infected using the `#ihavecorona` hashtag (but try to use your imagination and familiarity with the dataset to define a different metric).

A moving average is a statistic that captures the average change in a data series over time. Each point in a moving average depends on a range of records. For example, computing a moving average over a week requires data from the previous 7 days.

> In Spark, analysis that depends on a range of previous records can be computed by Window functions.

Prepare a CSV file where each line should capture the value of your metric for each day in the period covered by the dataset. Each line should have two columns: a date following the ISO 8601 standard indicating a day, and the value of your metric for the week prior to that day. Also prepare a time series graph in PNG or PDF format showing the data in the CSV.

> You can use Jupyter or other frameworks for the plotting. This tutorial shows how to plot time series using matplotlib.

Grading

All tasks will be graded with equal weight, that is, each task is worth *1/3* of the total grade. CSV files should follow the format described in each task's description. Graphs should be understandable given only a short caption: Graphs should include a title and label on all axes. Graphs should also include a legend when appropriate (for example, when there are multiple lines in a graph).

Although not a strict requirement, students should strive to make efficient use of computing resources on the cluster. Unecessary and inefficient use of resources (e.g., duplicating the dataset) may have a negative impact on the final grade. Remember that the cluster is shared for all students, be careful not to extrapolate its resources.

What to Submit

You should submit on Sakai:

- All code you developed in this project. Organize the code of each task in a separate directory and submit a zip file containing code for all tasks.
- Three CSV files containing results, one CSV for each task.
- Three graphs: one word cloud graph (Task 1.1), one histogram graph (Task 1.2), and one time series graph (Task 3).
- A PDF file discussing whether verified users grew their follower base using the results from Task 2.

Extra Credit

Solving Task 1 in Spark and solving Task 2 in MapReduce are worth 10% extra credit each (for a maximum of 20% extra credit). Suggestions for other extensions as extra credits are welcome.