Big Data 18CS322

Assignment 1

Analysis of QuickDraw images with MapReduce

Assignment Objectives and Outcomes

a. The objective of this assignment is for the students to install Hadoop in pseudo distributed mode and to become familiar with the Map Reduce programming environment and the HDFS file system.

b. At the end of this assignment, the student will be able to write and debug Map-Reduce code.

Ethical practices

Please submit original code only. You can discuss your approach with your friends but you must write original code. All solutions must be submitted through the portal. We will perform a plagiarism check on the code.

The Dataset:

c. We shall provide a link to the dataset on PESU forums. The schema for the dataset is shared below

Key	**Type**	**Description**
key_id	Numeric string	A unique identifier across all drawings.
word	string	Category the player was prompted to draw.
recognized	boolean	Whether the word was recognized by the game.
timestamp	datetime	When the drawing was created.
countrycode	string	A two letter country code of where the player was located.
drawing	array	A JSON array representing the vector drawing

d. Use this dataset for the tasks given below.

Software/Languages to be used:

e. Python (Version **3.6**) (Use Hadoop Streaming) or Java (Version **1.8** Only)

f. Please make sure to use Hadoop Version **3.2.0** only.

Marks:

g. Task 1 : 2 marks

h. Task 2 : 2 marks

i. Viva : 1 mark

Submission Date:

j. To be announced on the forum

Tasks Overview:

k. Remove bad records
l. Load the data into HDFS.
m. Write the output of the mapper and reducer for each question in the report.
n. Submit one page report based on the template and answer the questions on the report.

Submission Link:

Will be shared with you through PESU Forums along with the report template.

Task Specifications:

a. Task 0:

i. Install Hadoop in Pseudo-Distributed mode.

ii. You can refer the following few links (but not limited to) to help you get started - (Make sure that you install the correct versions of any software - as mentioned above)

iii. https://medium.com/@nidhinmahesh/getting-started-hadoop-mapreduce-hdfs-and-yarn-configuration-and-sample-program-febb1415f945

iv. https://netjs.blogspot.com/2018/02/installing-hadoop-single-node-pseudo-distributed-mode.html

v. https://medium.com/@jayden.chua/installing-hadoop-on-macos-a334ab45bb3
(This link should help MacOS users)

vi. You might need to use StackOverflow extensively to complete this task.

vii. See the setup video shared..

b. Task 1:

i. Problem Statement:

Find object count.

ii. Description:

- Find the number of occurrences of a given word, that has been marked as recognized.
- Find the number of occurrences of the same word that was not marked as recognized and whose timestamp falls on a weekend - {Saturday, Sunday}

iii. Comments:

- Refer subpoint d - to ignore bad records/special characters.
- Make sure the given word is passed as a command line argument to either mapper or reducer - do not hardcode it. [Refer example below (iv)]

Recommended (Not mandatory)

- Python - datetime module
- Java - LocalDate class
- Java - json.org package for parsing

iv. Output Format
Each output for the subtasks should be a new line.
<hadoop cmd> “aircraft carrier”
Eg. Assuming the number of recognized aircraft carriers in the json file is 1945, and the number of unrecognized aircraft carriers whose timestamp lies on either Saturday or Sunday is 348, the output expected is as follows:

```
1945
348
```

c. Task 2:

i. Problem Statement:
Find object count by country.

ii. Description:

Given a specific word, count the number of occurrences of that word per country.

iii. Comments

- Output should be sorted in alphabetical order of countries
- An occurence of the word should only be considered if the Euclidean distance between the 0th coordinates of the first stroke and the origin is greater than a given distance k.
- Both specified word and distance k should be command line arguments. The first argument should be the word, and the second argument should be the distance k.

iv. Output format

- Comma separated values in the form of:
  Country_code, count_of_word

- Each pair must be in a new line, with no space between the values
  <hadoop command> “aircraft carrier” 100
  Specific word : aircraft carrier
  Distance k : 100

  Eg. (The values below are just for representational purposes)
  AE,3
  AU,53
  BR,43
  NY,345

d. Removing bad records

You need to ensure that each of the records (i.e., JSON objects) satisfy the following conditions. If a particular record does not satisfy the same, it is considered as a “bad record” and should not be considered.

JSON property	**Condition to be satisfied**
word	Contains **alphabets and whitespaces** only.
countrycode	Contains **only two uppercase** letters
recognized	Boolean value containing either **"true" or "false"**
key_id	Numeric string containing **16 characters only**
drawing	Array containing n(>=1) strokes. Every stroke has exactly 2 arrays - where each array represents the ‘x’ and ‘y’ pixel coordinates respectively.

```
Eg:-
[
  [  // First stroke
    [x0, x1, x2, x3, ...],
    [y0, y1, y2, y3, ...]
  [  // Second stroke
    [x0, x1, x2, x3, ...],
    [y0, y1, y2, y3, ...]
  ],
  ... // Additional strokes
]
```

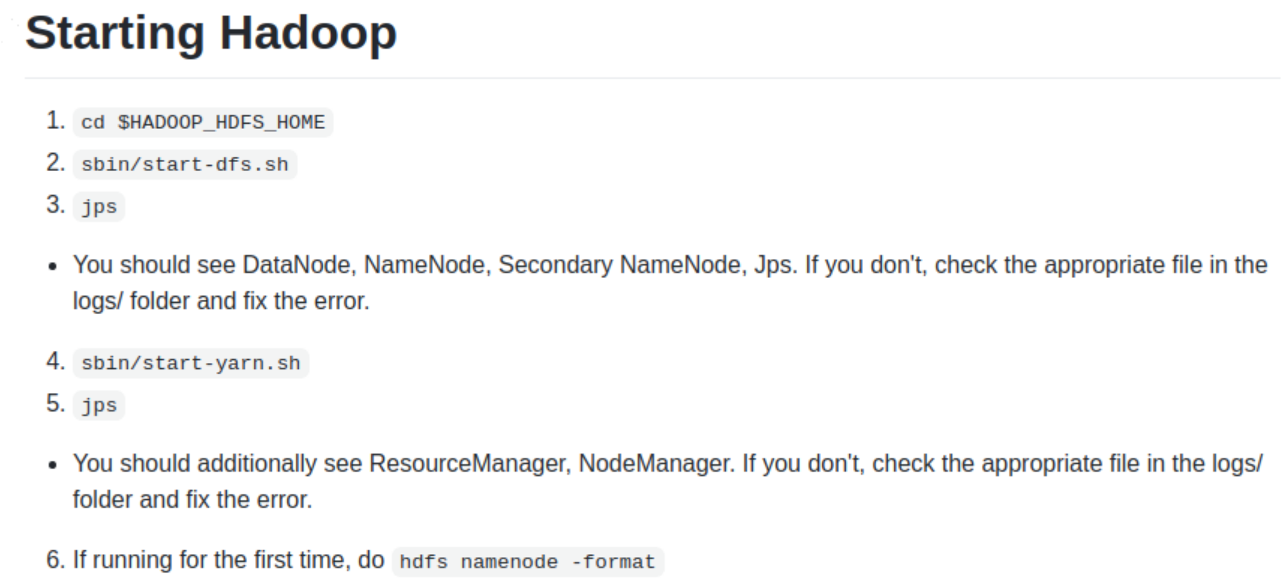
Eg:- (Representational purposes only)

Valid Record:
{"word": "airplane", "countrycode": "US", "timestamp": "2017-03-08 21:12:07.26604 UTC", "recognized": true, "key_id": "5152802093400064", "drawing": [[[167, 109, 80, 69, 58, 31, 57, 117, 99, 52, 30, 6, 1, 2, 66, 98, 253, 254, 246, 182, 165], [140, 194, 227, 232, 229, 229, 206, 124, 123, 149, 157, 159, 153, 110, 82, 77, 74, 109, 121, 127, 120]], [[207, 207, 210, 221, 238], [74, 103, 114, 128, 135]], [[119, 107, 76, 70, 49, 39, 60, 93], [72, 41, 3, 0, 1, 5, 38, 70]]]}

Bad Record:
{"word": "aircraft+carrier", "countrycode": "US", "timestamp": "2017-03-08 21:12:07.26604 UTC", "recognized": true, "key_id": "515280209300064", "drawing": [[[167, 109, 80, 69, 58, 31, 57, 117, 99, 52, 30, 6, 1, 2, 66, 98, 253, 254, 246, 182, 165], [140, 194, 227, 232, 229, 229, 206, 124, 123, 149, 157, 159, 153, 110, 82, 77, 74, 109, 121, 127, 120]], [[207, 207, 210, 221, 238], [74, 103, 114, 128, 135]], [[119, 107, 76, 70, 49, 39, 60, 93], [72, 41, 3, 0, 1, 5, 38, 70]]]}

This is a bad record because ‘word’ property contains a ‘+’ character and ‘key_id’ property contains only 15 characters.

Helpful Hadoop Commands



Starting Hadoop

1. `cd $HADOOP_HDFS_HOME`
2. `sbin/start-dfs.sh`
3. `jps`

- You should see DataNode, NameNode, Secondary NameNode, Jps. If you don't, check the appropriate file in the logs/ folder and fix the error.

4. `sbin/start-yarn.sh`
5. `jps`

- You should additionally see ResourceManager, NodeManager. If you don't, check the appropriate file in the logs/ folder and fix the error.

6. If running for the first time, do `hdfs namenode -format`

Loading a file into HDFS

```
hdfs dfs -put full-local-file-path/ /hdfs-dir-name
```

Running Python MR:

```
hadoop jar path-to-streaming-jar-file \
-input path_to_input_folder_on_hdfs \
-output path_to_output_folder_on_hdfs \
-mapper absolute_path_to_mapper.py \
-reducer absolute_path_to_reducer.py
```

Running Java MR:

```
bin/hadoop jar path_to_MR_jar_file.jar /path_to_input_folder_on_hdfs
/path_to_output_folder_on_hdfs
```