CS-GY 6513 Big Data
Assignment 1

Runze Li
rl5083@nyu.edu

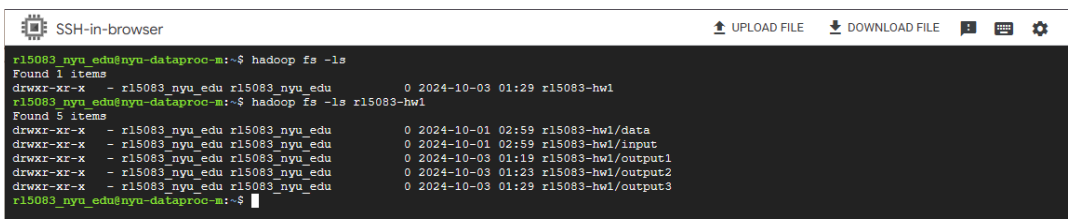
October 6, 2024

1 HDFS

Running any version of Hadoop (Dataproc, HPC, docker or otherwise), submit screen grabs (a picture in jpg, pdf or other suitable format) of the following:

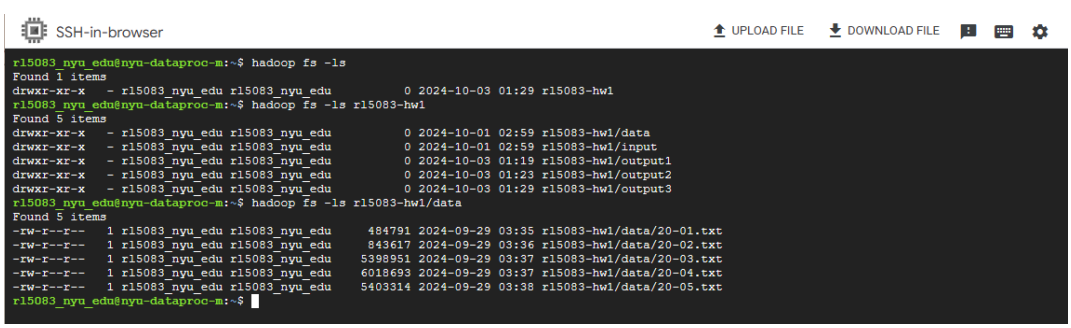
a) Create a directory in HDFS with this format: netid-hw1 (e.g. mine will be 'jcr365-hw1'). Submit a screen grab of the output of a Hadoop file listing showing your home directory and your new directory in it.

```
$ hadoop fs -mkdir rl5083-hw1
```



b) Create a subdirectory in HDFS, 'netid-hw1/data' and extract all input files into it. Submit a picture of directory listings or otherwise show the input files in it.

```
$ hadoop fs -mkdir rl5083-hw1/data
$ hadoop distcp gs://nyu-dataproc-hdfs-ingest/20-01.txt /user/rl5083_nyu_edu/
    rl5083-hw1/data
$ hadoop distcp gs://nyu-dataproc-hdfs-ingest/20-02.txt /user/rl5083_nyu_edu/
    rl5083-hw1/data
$ hadoop distcp gs://nyu-dataproc-hdfs-ingest/20-03.txt /user/rl5083_nyu_edu/
    rl5083-hw1/data
$ hadoop distcp gs://nyu-dataproc-hdfs-ingest/20-04.txt /user/rl5083_nyu_edu/
    rl5083-hw1/data
$ hadoop distcp gs://nyu-dataproc-hdfs-ingest/20-05.txt /user/rl5083_nyu_edu/
    rl5083-hw1/data
```



2 Beginner's Language Models with MapReduce

2.1 10 Most likely words

For this problem, you do not need to compute the unigram (single word) probabilities. Recall from the earlier explanation that the denominator is constant across all words. So, you can just count words, then output the top 10. This is a 2 job problem. The output of job1 can feed into the job2.

You must NOT lowercase or normalize the input. You must tokenize punctuation as a single word/token. For example:

"The cat is purring, so do not wake her." should tokenize as:
"The", "cat", "is", "purring", "so", ",", "do", "not", "wake", "her", "."

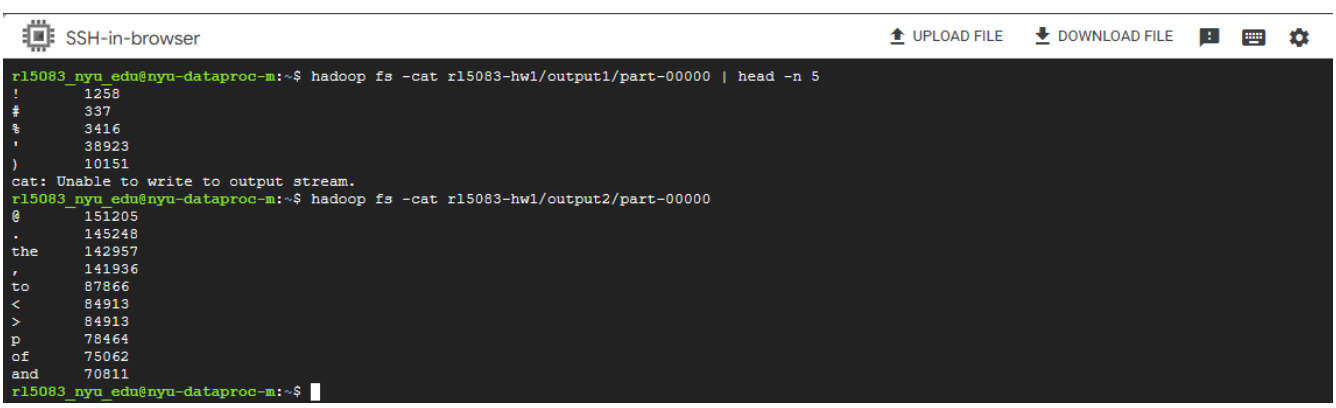
- The input is lines of text: TextInputFormat is your input data class.
- In Hadoop MapReduce, you can pass it individual files, whole directories (which are recursed) or simple wildcard patterns to match multiple files.
- Remember: Your mapper is guaranteed to exist only for a single input split. Your reducer is guaranteed to exist only for a single key.
- You have full control to define what is a key and what is a value.
- Replace all multiple spaces to a single space before.
- Ignore lines with less than 1 word.
- Mappers and Reducers cannot hold global state.
- You must use multiple jobs to solve this problem.
- Reducer parameters are one-way, one-shot iterators. You cannot double loop on the iterator.

Input for this problem: hw1text.zip (provided in class website)

After writing the code for WordCount (**mapper1.py, reducer1.py**) and Top-10 Words (**mapper2.py, reducer2.py**), we run the following command to get the result for WordCount and Top-10 Words:

```
$ hadoop jar $HADOOP_HOME/hadoop-streaming-3.3.6.jar -input /user/
   rl5083_nyu_edu/rl5083-hw1/data -output /user/rl5083_nyu_edu/rl5083-hw1/
   output1 -file mapper1.py -file reducer1.py -mapper "python mapper1.py" -
   reducer "python reducer1.py"

$ hadoop jar $HADOOP_HOME/hadoop-streaming-3.3.6.jar -D mapreduce.job.reduces=1
    -input /user/rl5083_nyu_edu/rl5083-hw1/output1 -output /user/rl5083_nyu_edu
   /rl5083-hw1/output2 -file mapper2.py -file reducer2.py -mapper "python
   mapper2.py" -reducer "python reducer2.py"
```



2.2 Extra credit – Simple ID Tokenizer

Use the word count given in class, or the output of Problem 2.1, and assign an increasing integer ID to each word (word) in order of decreasing count. ID's start at 1.

For example, if the word counts in the input are like this:

The, 123
house, 99
is, 88
cat, 76
...

Your output should be a table like this: 1, The
2, house
3, is
4, house
....

Input for this problem: hw1text.zip (provided in class website)

Here we use the result of WordCount as the input of ID-Tokenizer (**mapper3.py, reducer3.py**). Run the following command to get the result:

```
$ hadoop jar $HADOOP_HOME/hadoop-streaming-3.3.6.jar -D mapreduce.job.reduces=1
    -input /user/rl5083_nyu_edu/rl5083-hw1/output1 -output /user/rl5083_nyu_edu
   /rl5083-hw1/output3 -file mapper3.py -file reducer3.py -mapper "python
   mapper3.py" -reducer "python reducer3.py"
```