**Uploading the relevant files onto your S3 Bucket**

For all uploads, you can either directly upload the file into the S3 Bucket via the management console or connect to your EC2 Instance and send any files that have been locally stored in your instance to S3.

1. **The dataset (Project Gutenberg data, ~10K books / 4.5 GB)**

A screenshot of a computer

AI-generated content may be incorrect.I have attached it with this file. Easiest way to upload it without dealing with storage issues is to just open your chosen S3 Bucket via the management console and select the “Upload” option. This will take some time, but you should be able to do other stuff will the file uploads.

When you reach the “Upload” page, just drag the entire unzipped folder into the box and you should see something like the image above. Just upload it and you should be good.

1. **The .jar files for the Java EMR Cluster**

At this point, you should already have a folder called “/mywordcountjar/” on your S3 Bucket. This file should be provided in the attached GitHub link and can be uploaded from there.

1. **The .jar files for the Python EMR Cluster**

You should have two Python files located in /home/hadoop/workspace/python

They should be called mapper.py and reducer.py

Run the two commands below and replace the S3 URI with your preferred location. You will reference this location later on in the EMR Task.

aws s3 cp mapper.py s3://bucket21047881/pyStreaming/mapper.py

aws s3 cp reducer.py s3://bucket21047881/pyStreaming/reducer.py

**Setting up the Java EMR Cluster**

For this, set up a usual EMR cluster with the following modifications provided in the list below.

1. A screenshot of a computer screen

   AI-generated content may be incorrect.Under Cluster configuration, change the EC2 Instance types to be the m8g.xlarge instead.
2. A screenshot of a computer

   AI-generated content may be incorrect. Under Cluster scaling and provisioning, change the Instance(s) size to 4.
3. A screenshot of a computer

   AI-generated content may be incorrect.Under steps, create or edit the step listed in the cluster. Feel free to change the Name if you want. The details are below, but you will need to edit the JAR location and Arguments based on where your files are stored on your S3 Bucket. The order of the arguments matter, but as long as each argument is separated by a whitespace, it does not matter how exactly you write them.

s3://bucket21047881/mywordcountjar/wc.jar

stubs.WordCount

s3://bucket21047881/pg\_text/

s3://bucket21047881/wordCountsTest/

1. Once done, Save the step and Clone the cluster. After a bit of waiting, it should start running. If it terminates though, then something may have gone wrong.

**Setting up the Python EMR Cluster**

For this, you can actually just follow all the steps as in setting up the Java EMR Cluster up until Step 3.

A screenshot of a computer

AI-generated content may be incorrect.The details of Step 3 look like this for the Python version. The arguments are a bit of a mess due to auto formatting on AWS, but I have arranged them below a bit more clearly.

command-runner.jar

hadoop-streaming

-input s3://bucket21047881/pg\_text

-output s3://bucket21047881/streamingWordCounts

-cacheFile s3://bucket21047881/pyStreaming/mapper.py#mapper.py

-cacheFile s3://bucket21047881/pyStreaming/reducer.py#reducer.py

-mapper mapper.py

-reducer reducer.py

As before, once down just save the step and clone the cluster. If all goes well, you will get your output in about 2 hours.