NGram Generation

168

In the first part of the project, you will work with a plain-text corpus of 6000 books from Project Gutenberg and generate a list of n-grams from it.

An n-gram is phrase with n-words in it. For example a 1-gram is a single word such as "this" or "where", and 2-grams are phrases with two words, such as "this is" or "where is".

This text corpus has been processed and stored in the following S3 location:

`s3://15-319-s13/book-dataset/pg_00` to **`s3://15-319-s13/book-dataset/pg_38`**

We recommend using **`hadoop distcp`** command to transfer the entire text corpus directly to an HDFS store for your MapReduce job.

Note: For this checkpoint, assign the tag with Key: **`Project`** and Value: **`4.2`** for all resources

Task 1:

Process the entire text corpus using a MapReduce job to output every phrase in the corpus, along with the number of times the phrase appeared. These n-grams must be in the following plain-text format:

```
<phrase><\t><count>
```

For Example:

```
this    1000
this is 500
this is a   250
```

Please note the following instructions/assumptions:

- Run your MapReduce job on a cluster provisioned using Amazon EMR.
- Use spot instances when possible.
- Do not exceed more than 2 USD per hour for your EMR cluster
- Generate 1-gram, 2-gram, 3-gram, 4-gram, 5-gram output in the same MapReduce job.
- You need to generate raw n-grams and not worry about punctuation or phrase semantics.
- You must limit the words in the phrase to be purely alphabetical **`[A-Za-z]`** and strip all punctuation and numbers. Non-alphabetical characters can be replaced with a space, generating additional words in the line.
- Treat your words as case-insensitive, store only lower-case words.
- You do not need to consider n-grams and phrases that span multiple lines in the text corpus. You can process each line independently.
- Treat underscore '_' as a non-word character (some regular expressions do otherwise).
- Do not output any empty characters "" or double spaces " " between words.
- Treat apostrophe as punctuation - so the word don't would generate the ngrams **`don ; don t ; t ;`**
- To obtain the output from HDFS to local disk, we advise running the **`hadoop dfs -get`** commands from the Master Node of your provisioned cluster.
- Transfer the output of your program to an S3 bucket for the next part of the project. You can do this by using **`s3cmd`** or by directly writing to S3 from your MapReduce program.

We recommend that you code and test and debug your program on a small file first before attempting to generate the n-grams for an entire text corpus. Once you have generated the n-grams from the corpus, transfer it to an S3 bucket. Upload your MapReduce Java code to S3 (in a different bucket or folder from your output) and set the permissions appropriately.

When you are ready, please complete the following checkpoint quiz:

Note: Please destroy your cluster in EMR after you have completed the checkpoint quiz below

checkpoint

NGram Generation