Distributed Computing and Storage Architectures: Project

Contents

1 Project Overview

We want to deploy an end-to-end big data harvesting and analysis chain. This includes mining, processing, and analysis of big data. More specifically, we will use Twitter and its public Application Programming Interface (API) to have access to an exceptionally valuable source of data, namely, tweets, which represent millions of voluntarily expressed opinions on any imaginable topic. In this project you will:

- Deploy a Hadoop cluster, capable of running distributed MapReduce tasks;
- Access the Twitter API using Python;
- Use the data to count the number of tweets that contain a given hashtag or a specific word.

2 Tasks

The tasks can be split into two main parts.

2.1 Hadoop

The first part is the deployment of the Hadoop cluster. You will be using the Cloudera Distribution of Apache Hadoop (CDH) and `mrjob` as the python MapReduce library.

The steps are as follows:

1. You can evaluate CDH by installing the Apache Hadoop and CDH components on a single Linux node in pseudo-distributed mode[1]. In pseudo-distributed mode, Hadoop processing is distributed over all of the cores/processors on a single machine. Alternatively (simpler option), you can use Cloudera QuickStart virtual machines (VMs)[2], which include everything you need to try CDH, Cloudera Manager, Cloudera Impala, and Cloudera Search. (follow the documentation available on the Cloudera website).
2. Install `mrjob` to run MapReduce using Python. Even though Hadoop was written to work primarily with Java code, it supports other languages using Hadoop Streaming (follow the documentation available on the `github mrjob` website).

[1] https://www.cloudera.com/documentation/enterprise/5-6-x/topics/cdh_qs_cdh5_pseudo.html

[2] http://www.cloudera.com/downloads/quickstart_vms/5-8.html

2.2 Twitter API

Twitter provides a rich API for querying the system and accessing data. More information can be obtained on the `dev.twitter.com` website. In order to obtain the tweets, these are the steps to be followed:

1. Make sure you have the `oauth2` library installed for authentication.

2. Create a twitter application—requires a twitter account; use a dummy website account when filling the forms, and obtain API keys and access-token keys.

3. Use your credentials in the `twitterstream.py` file—available in the `./goodies/Twitter Collect` folder—to start harvesting Twitter data (always try to understand code that is already provided). Alternatively, you can use any Python library that supports Twitter APIs, such as Twython[3] or Tweepy[4].

4. Harvest tweets originated from the United States of America by using as location query the following geocoordinates

$$[-122.995004, 32.323198, -67.799695, 49.893813].$$

   These geocoordinates in Twitter format define a bounding box that covers the geographical area of the USA. Make sure that you harvest only tweets in English. You will see that each tweet contains a lot of extra information apart from the actual message. You will need to obtain the text message from the initial tweet. Save the text into a file. Let the script run a considerable amount of time, to gather sufficient data, namely, 20.000 tweets.

5. An essential part of text analysis is to have clean text. Tokenization is used to split a raw text into useful meaningful components. You can use built-in tokenizers from `nltk`[5], namely, the `TweetTokenizer`. Alternatively, You can use Brendan O'Connor's twokenizer; namely, use the files `emoticons.py` and `twokenizer.py` available in the `./goodies/tokenizer` folder.

6. Retrieve the hash-tags `#hstg` from the harvested tweets. You do not need to parse the text messages, this information is part of the Tweet object. Keep the most popular 10 hash-tags (this means that in MapReduce you need to count the number of times each hashtag appears).

[3] https://twython.readthedocs.io/en/latest/
[4] http://tweepy.readthedocs.io/en/v3.5.0/
[5] http://www.nltk.org/api/nltk.tokenize.html

7. Retrieve the list of 10 most popular English words in your data. Namely, use the MapReduce framework to count the number of occurrences of each word in your dataset.

3 Comments

All the code should be written in Python. The final version of the code should be written to run in a distributed fashion, using MapReduce. But initial versions, for debugging purposes, can run locally.

4 Project Requirements

In this project, you will work in teams of **2 people**. You can form teams by yourselves, and inform us by the **30th November 2017**.

Deliverables

Upon finishing the project, you need to submit:

- A report describing your work and results for each task in the project
- The full source code of your project, with clear comments
- A detailed guideline to run your code, including the dependencies, the commands to run, etc.
- A description of the contributions of each team member in the project

Deadline

The deadline for the submission of the project report and material is **23:59 CET, Sunday 14th January 2018**. Any submission after this deadline is considered invalid. You will have to combine the deliverables in a .zip folder and submit them through email to ndeligia@vub.be

Evaluation

For this project, you will be evaluated based on following criteria:

- Your written report (explicitly mentioning the contribution of each member of the team) and your source code.