A Simple Index for Wikipedia Page Retrieval

Murari Mani

Objectives

- Build a reverse index for quick lookup for wikipedia
- Rank documents according to query and document contents
- Retrieve best document matching query

Data Source

- Wikimedia data dump of current articles pages-articles-multistream.xml.bz2
  - Current revisions only, no talk or user pages
- http://itorrents.org/torrent/3A8C87DE09C85193CFBCB10DC64B7A64C2CEE7FC.torrent Size: ~15 GB compressed
- Tested by downloading and extracting the simple english wikipedia which is 179 MB uncompressed
  - simplewiki-20170820-pages-meta-current.xml.bz2 (179.8 MiB) on itorrents.org
- Parser : https://github.com/attardi/wikiextractor

Structure of Parsed Data

Split into xml pages of 100MB each (configurable) with individual pages delineated by <doc> and </doc> tags

```
<doc id="1" url="https://simple.wikipedia.org/wiki?curid=1" title="April">

April

April is the fourth month of the year, and comes between March and May. It is one of four months to have 30 days.

...

</doc>
```

Strategy for Storage & Document Retrieval

Step 1: Read data into HDFS

Natural choice since the data is unstructured and we will be performing a bunch of map-reduce tasks

Step 2: Compute the tf-idf* score to get the most relevant meta-document i.e wiki_00

Step 3: Within wiki_00 compute the tf-idf* for individual html pages to get the best matching webpage

*https://nlp.stanford.edu/IR-book/pdf/06vect.pdf

Strategy for Storage & Document Retrieval

Step 1: Read data into HDFS

Natural choice since the data is unstructured and we will be performing a bunch of map-reduce tasks

Step 2: Compute the tf-idf score to get the most relevant meta-document i.e wiki_00

Step 3: Within wiki_00 compute the tf-idf for individual html pages to get the best matching webpage

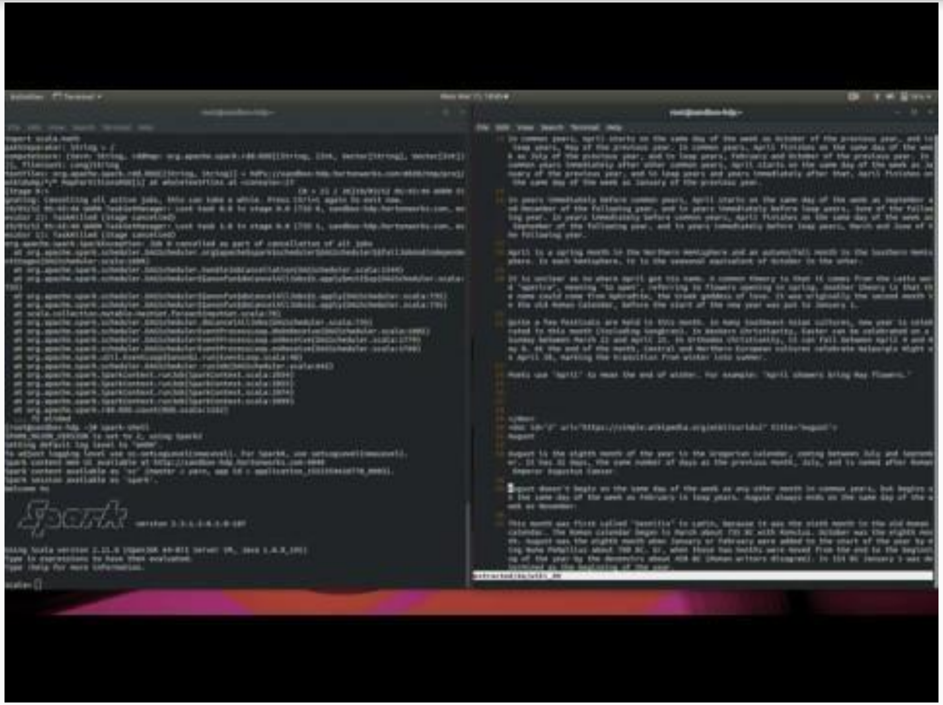
Shortcomings

1. Performs well for uncommon queries such as “blah”
2. Does not work so well for common queries such as “president”
3. Reason: All fields of the document are equally weighted
4. We should assign more weight to the title/summary so that the score is adjusted accordingly
5. **Errors when I try to run the large data set on the SandBox**

Future Extensions

1. Support more complex queries
2. Support zone based weighting
3. Improve the ranking by incorporating more sophisticated ML-based ranking
4. Deploy on a real HDFS cluster

Demo at: https://youtu.be/leZVQ9L9j-g


[illegible]

Thanks for listening :) !!