CS 5604 Information Storage and Retrieval
Solr Team Final Presentation

Presenters:
Liuqing Li, Ye Wang, Anusha Pillai, Ke Tian
{liuqing, yewang16, anusha89, ketian} @vt.edu

Instructor:
Dr. Edward A. Fox

Virginia Polytechnic Institute and State University
Blacksburg, VA, 24061
December 6, 2016
Outline

• Background
• Implementation
• Problems Faced
• Lessons Learned
• Future Work
• Acknowledgement
Background — Overview

CMT  CMW  CLA  CTA

HDFS

HBase
- Basic Indexing
- Inc Indexing
  - Morphline
    - Lily HBase Indexer

Solr
- schema.xml
- solrconfig.xml
  - Request Handler
  - Search Component
  - Recommendation
  - Custom Ranking
  - Solr Admin UI

Cloudera

Hue UI

FE  Blacklight

Solr Team Final Presentation
## Background — Updates

<table>
<thead>
<tr>
<th></th>
<th>Spring 2016</th>
<th>Fall 2016</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>schema.xml</strong></td>
<td>Coarse grained</td>
<td>Fine grained</td>
</tr>
<tr>
<td></td>
<td>No copyfields</td>
<td>Copyfields for all fields search</td>
</tr>
<tr>
<td></td>
<td>Create stopwords.txt &amp; profanity.txt</td>
<td>Update the two files</td>
</tr>
<tr>
<td><strong>morphlines.conf</strong></td>
<td>Two field types: string and text</td>
<td>Multiple field types</td>
</tr>
<tr>
<td></td>
<td>Field “time” =&gt; string</td>
<td>Field “time” =&gt; datetime</td>
</tr>
<tr>
<td></td>
<td>No multiple-valued fields</td>
<td>Multiple-valued field parser</td>
</tr>
<tr>
<td><strong>Basic Indexing</strong></td>
<td>Small collection</td>
<td>1.2 billion tweets dataset</td>
</tr>
<tr>
<td><strong>Incremental Indexing</strong></td>
<td>Virtual Cloudera (VC)</td>
<td>VC &amp; Hadoop Cluster (HC)</td>
</tr>
<tr>
<td><strong>Recommendation</strong></td>
<td>Brief description</td>
<td>Implemented in VC &amp; HC</td>
</tr>
<tr>
<td><strong>Custom Ranking</strong></td>
<td>Brief description</td>
<td>Implemented in VC &amp; HC</td>
</tr>
<tr>
<td><strong>Solr Admin UI</strong></td>
<td>Brief description</td>
<td>Detailed description</td>
</tr>
<tr>
<td></td>
<td>Limited faceted search</td>
<td>Detailed faceted search</td>
</tr>
</tbody>
</table>
Implementation — Basic Indexing

• Live Mode
  • Continuous stream of HBase cell updates into live search indexers
  • Simple and efficient
  • Cannot handle big data

• Batch Mode
  • Batch index tables in HBase by using MapReduce jobs
  • Write index files into HDFS (/user/cs5604f16_solr/...)
  • Can handle big data
• schema.xml: fields configuration
  • field (e.g., ideal-cs5604f16-fake)
  • # of fields: 30
  • Types: string (22), text_general (2), int (2), float (2), long (1), date (1)
  • Stored: True (17), False (13)

```xml
<field name="t_month_i" type="int" indexed="true" stored="true"/>
<field name="hashtags_s" type="string" indexed="true" stored="false" multiValued="true"/>
```

• dynamicField: matching multiple fields, using wildcard

```xml
<dynamicField name="*_s" type="string" indexed="true" stored="true"/>
<dynamicField name="*_ss" type="string" indexed="true" stored="true" multiValued="true"/>
```

• copyField

```xml
<copyField source="*_ss" dest="text" maxChars="3000"/>
```
Implementation — Basic Indexing

• **stopword.txt** and **profanity.txt**
  • **stopword.txt**: tf-idf value will not be calculated
  • **profanity.txt**: quick response for such search queries
  • Solr loads the two files while reading schema.xml

```xml
<!-- Case insensitive stop word removal. -->
<filter class="solr.StopFilterFactory"
  ignoreCase="true"
  words="lang/stopwords_en.txt"
/>
<filter class="solr.LowerCaseFilterFactory"/>
<filter class="solr.EnglishPossessiveFilterFactory"/>
<filter class="solr.KeywordMarkerFilterFactory" protected="protwords.txt"/>
```

Source:
https://pypi.python.org/pypi/many-stop-words
http://www.freewebheaders.com/full-list-of-bad-words-banned-by-google/
Implementation — Basic Indexing

- morphlines.conf: mapping and parsing

Mapping data from HBase to Solr

```
mappings: [
    # tweet : cleantext
    {
        inputColumn: "tweet:cleantext"
        outputField: "raw_cleantext_s"
        type: string
        source: value
    }
]
```

Split multiple values into list

```
split {
    inputField : "topic_label_s"
    outputField : "topic_label_ss"
    separator : ";"
    isRegex : false
    addEmptyStrings : false
    trim : true
}
```

```
"topic_label_s": "twitter;social;media;text"
```

```
"topic_label_ss": [
    "twitter",
    "social",
    "media",
    "text"
],
```
## Implementation — Basic Indexing

- Index the big dataset

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Indexing</th>
<th>ideal-cs5604f16</th>
<th>ideal-cs5604f16-1204</th>
</tr>
</thead>
<tbody>
<tr>
<td># of DataNode</td>
<td>18</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>Space Cost</td>
<td>392.33 GB</td>
<td>399.21 GB</td>
<td></td>
</tr>
<tr>
<td>Indexing</td>
<td>Time Cost</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mapping</td>
<td>1h21m</td>
<td>1h45m</td>
<td></td>
</tr>
<tr>
<td>Reducing</td>
<td>5h11m</td>
<td>5h13m</td>
<td></td>
</tr>
<tr>
<td>Merging</td>
<td>3h18m</td>
<td>3h10m</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>9h50m</td>
<td>10h8m</td>
<td></td>
</tr>
</tbody>
</table>
Implementation — Incremental Indexing

• Purpose
  • Process a continuous stream of HBase cell updates into live search indexes (Near Real-Time, NRT Indexing)
  • Solve the problem of frequent inserts, deletes and updates

• How does it work?
  • Enabling HBase replication (columnfamily)
  • Pointing an NRT Indexer Service at an HBase table
  • Starting an NRT Indexer Service

• Our work

Source:
http://www.cloudera.com/documentation/enterprise/5-6-x/topics/search_config_hbase_indexer_for_search.html
Implementation — Incremental Indexing

Create and check the NRT indexer

```bash
```

```bash
[cs5604f16_solr@node1 ~]$ hbase-indexer list-indexers
ZooKeeper connection string not specified, using default: localhost:2181

Number of indexes: 1

NRTindexer
  + Lifecycle state: ACTIVE
  + Incremental indexing state: SUBSCRIBE_AND_CONSUME
  + Batch indexing state: INACTIVE
  + SEP subscription ID: Indexer_NRTindexer
  + SEP subscription timestamp: 2016-11-24T19:26:45.331-05:00
  + Connection type: solr
  + Connection params:
    + solr.collection = ideal-cs5604f16-fake
    + solr.zk = node1.drlr:2181,node2.drlr:2181,node3.drlr:2181,node4.drlr:2181,solr2.drlr:2181/solr
```
Implementation — Incremental Indexing

Restart the HBase Solr Indexer service

Restart the service in VC

```
[cloudera@quickstart ~]$ sudo service hbase-solr-indexer restart
Stopped HBase Solr Indexer: [ OK ]
Started HBase Solr Indexer (hbase-solr-indexer): [ OK ]
[cloudera@quickstart ~]$ 
```

Restart the service in HC
Implementation — Incremental Indexing

Check the results in HBase and Solr Admin UI

Create and check the NRT indexer
Check the results in HBase and Solr Admin UI
Implementation — Recommendation

• Types
  • Textual similarity based
  • Collaborative filtering

• More Like This Component
  • Identifies similar documents to search result documents.
  • Can be configured as a request handler or search component
  • Uses term vectors to compute similarity.
  • Term vector can be calculated during query runtime or precomputed during indexing
  • Extracts highest matching terms based on tf-idf similarity
Implementation — Recommendation

• schema.xml
  • Set stored = true
  • Set termVectors = true (for calculating tf-idf)
    • After making changes, reindexing is mandatory

• solrconfig.xml
  • Enable mlt

```xml
<requestHandler name="/mlt" class="solr.MoreLikeThisHandler">
  <lst name="defaults">
    <str name="rows">5</str>
    <str name="mlt.fl">text_txt</str>
    <str name="mlt.mintf">1</str>
  </lst>
</requestHandler>
```

• Define other configuration parameters
  • e.g., mlt.fl, mlt.mintf, mlt.mindf, mlt.maxdf, mlt.qf
Implementation — Recommendation

• Request Handler

Link:
https://drive.google.com/open?id=0B2iasHDgHqGyYUk0R3RkVktkM2M
• Search Component

Link:
https://drive.google.com/open?id=0B2iasHDgHqGyU0doVEpidlh3c2c
Implementation — Custom Ranking

• Purpose
  • Customize and optimize the ranked results

• How does it work?
  • Search Component
    • `prepare()`: pre-processing, invoked before query is executed
    • `processing()`: post-processing, invoked after all the results are fetched
  • Custom Scoring

\[
\text{Score} = \text{Doc}_\text{score, Solr} + \text{Doc}_\text{importance} + W_{\text{topic}} \times \text{Doc}_\text{score, topic} + W_{\text{cluster}} \times \text{Doc}_\text{score, cluster}
\]

• Re-ranking
Implementation — Custom Ranking

Build and copy jar file into Hadoop Cluster
Implementation — Custom Ranking

Modify the solrconfig.xml

```xml
<requestHandler name="/custom" class="solr.SearchHandler">
  <!-- default values for query parameters can be specified, these will be overridden by parameters in the request -->
  <lst name="defaults">
    <str name="echoParams">explicit</str>
    <int name="rows">10</int>
    <str name="df">text</str>
    <str name="fl">*, score</str>
  </lst>

  <arr name="last-components">
    <str>Customranking</str>
  </arr>

</requestHandler>

<searchComponent name="Customranking" class="cs5604f16.solr.Customranking">
</searchComponent>
```

Build and copy jar file into Hadoop Cluster

Modify the solrconfig.xml
Implementation — Custom Ranking

Update the instanceDir
Reload the collection
Check the results in Solr Admin UI

```
start, rows
0, 10

fl
t_importance_f, score
df

Raw Query Parameters
key1=val1&key2=val2

wt
json

"response": {
  "numFound": 21,
  "start": 0,
  "maxScore": 2,
  "docs": [
    {
      "t_importance_f": 0.5,
      "score": 1.5
    },
    {
      "t_importance_f": 0.42,
      "score": 1.42
    },
    {
      "t_importance_f": 0.0206,
      "score": 1.0206
    },
    {
      "t_importance_f": 0.02,
      "score": 1.02
    }
  ]
}
```
Implementation — Solr Admin UI

DashBoard: provide basic functions for users to choose. (Logging to check Solr logs for debugging)

Core Selector: select the core (dataset) for queries

Solr instance Information: current versions, JVM information

Choose ideal-cs5604f16-fake for querying
Implementation — Solr Admin UI

1. The request-handler: /select

2. The query event: q

3. Parameters for query: fq (filter queries), sort (descending or ascending)

4. Execute query

5. Results outputs: json format

---

Solr Team Final Presentation
The faceted search query: range
Faceted search field: t_month_i
Parameters, true when enabled
Search Results: counts
Search Results: details
# Problem Faced

## Cloudera and OS
- Virtual Cloudera seems slow and often crashes due to the memory
- Not familiar with the whole architecture at the beginning
- Versions of Cloudera and Solr

## Data
- Consistency check
- Not enough real data available to perform tests
- Not much information available regarding logs to perform collaborative filtering

## Collaboration
- Communication and modification
Lessons Learned

- Solr
- HBase
- HDFS
- Patience
- Carefulness
- Team Collaboration
Future Work

Search
- Customize more request handlers
- Deal with the profanity issue

Custom Ranking
- Customize more search components

Recommendation
- Create a custom recommendation component (Probabilities – CTA team)
- Implement the collaborative filtering (Log files – FE team)

Solr
- Figure out SolrCloud, multiple Solr nodes in Cloudera Search
## Acknowledgement

### Projects

<table>
<thead>
<tr>
<th>Projects</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSF IIS - 1319578</td>
<td>III: Small: Integrated Digital Event Archiving and Library (IDEAL)</td>
</tr>
<tr>
<td>NSF IIS - 1619028</td>
<td>III: Small: Collaborative Research: Global Event and Trend Archive Research (GETAR)</td>
</tr>
</tbody>
</table>

### Teams

- CMT, CMW, CLA, CTA, FE teams

### Persons

<table>
<thead>
<tr>
<th>Role</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instructor</td>
<td>Dr. Edward A. Fox</td>
</tr>
<tr>
<td>GRA</td>
<td>Sunshin Lee</td>
</tr>
</tbody>
</table>
Thank you!

Questions?