Objective

Create a generic framework for evaluation of alternate retrieval methods/softwares for the given set of documents.

Background

I have a business need in my current work to choose either Oracle Free Text Search or Solr to index text data. This indexed data will be integrated and used for relevancy search. Thanks to the solid understanding I gained from this course "CS-410 Text Information Systems", I was able to propose and got approval for this initiative.

Note: Also at my work place a third-party software Elsevier is currently used to index scientific grant related text like abstract, specific aims. The index score output is used to categorize research projects under various disease categories. In my original project proposal, I was considering to perform analysis of softwares that can be considered for a potential replacement for vendor provided software. But I didn't get approval for this original proposal from team management. So I have evaluated information retrieval softwares.

1 Overview of Functions

1.1 Oracle Free Text Search Creation

- Create a table with the required fields/columns.
- Identify the column that will be used in the next for text search.
- Create a context index on the column that will be used for text search.

1.2 Solr Index Creation

- Create a schema with the required fields/columns.
- Configure the query parser Lucene/Edismax.
- Index the data.

1.3 Evaluation

- Select query terms, phrases and other factors for evaluation.
- Execute query and get results from evaluation software 1 – Oracle text search
- Execute query and get results from evaluation software 2 – Solr
- Get explicit feedback from users (Gold copy)
- Calculate precision, recall and F-scores

2 Implementation

2.1 Oracle Free Text Search Creation

Oracle 12c is used for this evaluation. The assumption is the user who is creating the table already as required privileges to create tables and context index.
Reference: https://docs.oracle.com/database/121/CCAPP/toc.htm

```
create table extractions_t(
  extraction_id      NUMBER(10) not null,
```

```
 appl_id              NUMBER(10) not null,
 extracted_text   CLOB,
 template_section_code   VARCHAR2(3),
 fy                   NUMBER
);
create index ARCH_EXTRACTED_TEXT on EXTRACTIONS_T (EXTRACTED_TEXT)
 indextype is CTXSYS.CONTEXT;
```

2.2 Solr Index Creation

- Schema Creation (partial managed-schema.xml is given below)
  Note extractText is configured as Text English general (text_en). This means Porter stemmer and stopwords will be applied to index data.

```
<field name="applId" type="long" indexed="true" required="true" stored="true"/>
  <field name="fy" type="long" indexed="true" required="true" stored="true"/>
  <field name="extractText" type="text_en" indexed="true" stored="true"/>
  <field name="id" type="string" multiValued="false" indexed="true" required="true"
stored="true"/>
  <field name="templateSectionCode" type="string" indexed="true" required="true"
stored="true"/>
```

- Method that creates and add data to Index using SolrJ client

```
private  void addIndex() throws IOException, SolrServerException {
        SolrClient client = new
HttpSolrClient.Builder("http://localhost:8983/solr/archived_extractions").build();

        Collection<ArchivedExtraction> docs = new ArrayList<ArchivedExtraction>();
        int i = 0;
        List<ExtractionRecord> extRecords = null;
        List<Integer> fys = new ArrayList<>();
        fys.add(2013);
        fys.add(2014);
        fys.add(2015);
        fys.add(2016);
        fys.add(2017);

        for (Integer fy :fys) {
                List<Long> applIDs =  textSearchService.getApplIDsByFy(fy);
                        BatchedList<Long> batchedList = new BatchedList<Long>(new
ArrayList<Long>(applIDs), BatchedList.ORACLE_IN_CLAUSE_SIZE);
            while(batchedList.hasNextBatch())
            {
                List<Long> applIdList = batchedList.nextBatch();
                extRecords = textSearchService.getFreeTextSearchResults(applIdList);
```

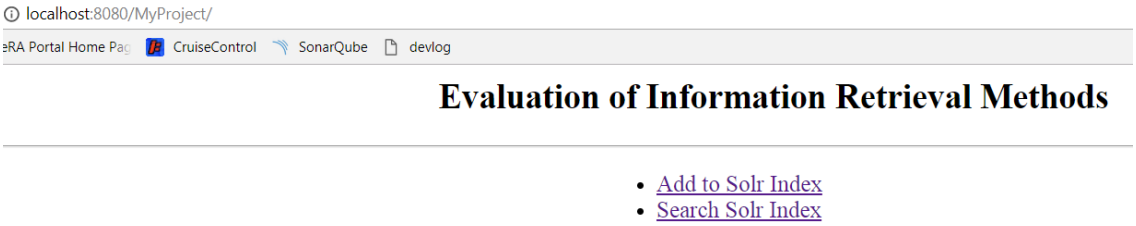
```
                for (ExtractionRecord extRecord : extRecords) {
                        ArchivedExtraction doc = new ArchivedExtraction();
                        doc.setApplId(extRecord.getApplId().toString());
                          doc.setId(extRecord.getExtractionId()+"-"+extRecord.getApplId());
                          //doc.setDocCreatedDate(LocalDate.now());
                          doc.setExtractText(extRecord.getExtractedText());
                          doc.setTemplateSectionCode(extRecord.getTemplateSectionCode());
                          doc.setFy(extRecord.getFy());
                          client.addBean(doc);
                          i++;
                          if(i%100==0) client.commit();
                                }
                        }
          client.commit();
        }
    }
```

2.3 Installing MyProject Web application

- Copy the war file MyProject.war to \Tomcat\8\webapps
- Start the tomcat server by \Tomcat\8\bin\startup.bat
- Edit the \Tomcat\8\wtpwebapps\MyProject\WEB-INF\MyProject-servlet.xml to configure datasource. Highlighted the values that need to be added for data access layer configuration.

```
<bean id="datasource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
      <property name="driverClassName" value="oracle.jdbc.driver.OracleDriver" />
      <property name="url"
value="jdbc:oracle:thin:@(DESCRIPTION=(ENABLE=BROKEN)(ADDRESS_LIST=(ADDRE
SS=(PROTOCOL=TCP)(HOST=localhost)(PORT=1530)))(CONNECT_DATA=(SERVICE_N
AME=orcl)(SERVER=DEDICATED)))" />
      <property name="username" value="scott" />
      <property name="password" value="tiger" />
  </bean>
```

- Stop the server by running by \Tomcat\8\bin\shutdown.bat
- Restart the tomcat server by \Tomcat\8\bin\startup.bat
- Access the application using URL http://localhost:8080/MyProject/



2.4 Evaluation

Table to store raw Statistics

Table to store appl_id which is unique identifier for a research project and its relevance to the given query search_term. If the given project is identified as relevant for a search term per explicit user feedback then the value of gold_standard_flag will be set to 1 else the value will be 0. Similarly if oracle free text search (fts) identifies the project as relevant to a search term the value of oracle_fts will be set to 1 else 0. If solr query identifies the project as relevant to a search term the value of will be set to 1 else 0.

```
CREATE TABLE stats_calc (search_term VARCHAR2(100),
                         appl_id number,
                         gold_standard_flag number default 0,
                         oracle_fts_flag number default 0,
                         Solr_idx_flag  number default 0
                         );

ALTER TABLE  stats_calc ADD CONSTRAINT stats_calc_pk PRIMARY KEY
(appl_id,search_Term);
```

Oracle Query For Searching a Given Query Term

```
 select appl_id from
   ( select distinct arch.appl_id appl_id
   from archived_extractions_t arch
   where contains(arch.extracted_text, 'Knee Osteoarthritis' , 1) > 0
     and latest_code = 'Y'
     and arch.template_section_code is null
     and arch.fy= 2017
   );
```

Java Method to Use Solr Query For Searching a Given Query Term

```
private  Set<Long> selectData( String queryTerm) {
                    SolrClient client = new
HttpSolrClient.Builder("http://localhost:8983/solr/archived_extractions").build();
                    HashSet<Long> applSet = new HashSet();
            try {
                    SolrQuery query = new SolrQuery();
                    Long fy = (long) 2017;
                    String searchquery ="fy:"+fy;
                    query.setQuery("extractText:"+queryTerm );

                 query.addFilterQuery(searchquery);
                 QueryResponse response = client.query(query);
                 SolrDocumentList results = response.getResults();
                 for (int i = 0; i < results.size(); ++i) {
                    applSet.add((Long) results.get(i).getFieldValue("applId"));
```

```
        }
            } catch (SolrServerException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
            } catch (IOException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
            }
    return applSet;
}
```

3. Usage

Step 1: Store explicit feedback

Creating Gold Standard(user feedback) Upload for query term Knee Osteoarthritis provided by users

```
insert into stats_calc(search_term,appl_id,gold_standard_flag) values ('Knee Osteoarthritis',9435379,1);

insert into stats_calc(search_term,appl_id,gold_standard_flag) values ('Knee Osteoarthritis',9413126,1);

insert into stats_calc(search_term,appl_id,gold_standard_flag) values ('Knee Osteoarthritis',9386212,1);

insert into stats_calc(search_term,appl_id,gold_standard_flag) values ('Knee Osteoarthritis',9385849,1);

insert into stats_calc(search_term,appl_id,gold_standard_flag) values ('Knee Osteoarthritis',9375095,1);

insert into stats_calc(search_term,appl_id,gold_standard_flag) values ('Knee Osteoarthritis',9371389,1);

insert into stats_calc(search_term,appl_id,gold_standard_flag) values ('Knee Osteoarthritis',9364179,1);

insert into stats_calc(search_term,appl_id,gold_standard_flag) values ('Knee Osteoarthritis',9353269,1);

insert into stats_calc(search_term,appl_id,gold_standard_flag) values ('Knee Osteoarthritis',9310337,1);

insert into stats_calc(search_term,appl_id,gold_standard_flag) values ('Knee Osteoarthritis',9197607,1);
```

Step 2: How to Store Oracle search results for given query term? (Knee Osteoarthritis)

```
MERGE INTO stats_Calc sc  USING ( select distinct arch.appl_id appl_id  ,'Knee
Osteoarthritis' search_term
  from extractions_t arch
  where contains(arch.extracted_text, 'Knee Osteoarthritis' , 1) > 0
      and arch.template_section_code is null
    and arch.fy= 2017
  ) ex
 ON (ex.appl_id = sc.appl_id AND ex.search_term = sc.search_Term)
 WHEN MATCHED THEN UPDATE SET sc.oracle_fts_flag =1
 WHEN NOT MATCHED THEN INSERT(appl_id,search_Term,oracle_fts_Flag)
VALUES (ex.appl_id,ex.search_term,1);
  ;
```

Step 3: How to Store Solr search results for given query term? (Knee Osteoarthritis)

```
public void  setStatsCalc(Long applId, String searchTerm , String flag) {
                MapSqlParameterSource params = new MapSqlParameterSource();

                StringBuilder nativeQL = new StringBuilder("MERGE INTO stats_Calc sc  USING
(SELECT :appl appl_id,:st search_term FROM dual) ex " );
                nativeQL.append("  ON (ex.appl_id = sc.appl_id AND ex.search_term =
sc.search_Term) " );
                nativeQL.append("  WHEN MATCHED THEN UPDATE SET sc." + flag + " =1 " );
                nativeQL.append("  WHEN NOT MATCHED THEN
INSERT(appl_id,search_Term,"+flag+" ) VALUES (ex.appl_id,ex.search_term,1) " );

                params.addValue("appl",applId);
                params.addValue("st", searchTerm);
                this.namedParameterJdbcTemplate.update(nativeQL.toString(), params);
        }
```

Step 4: Calculate Stats

Stats calculation for one Software (Oracle FTS) is given here. Same steps can be repeated for another software evaluated.

	User Y	User N
System +	TP	FP
System -	FN	TN

Precision = TP/(TP+FP)

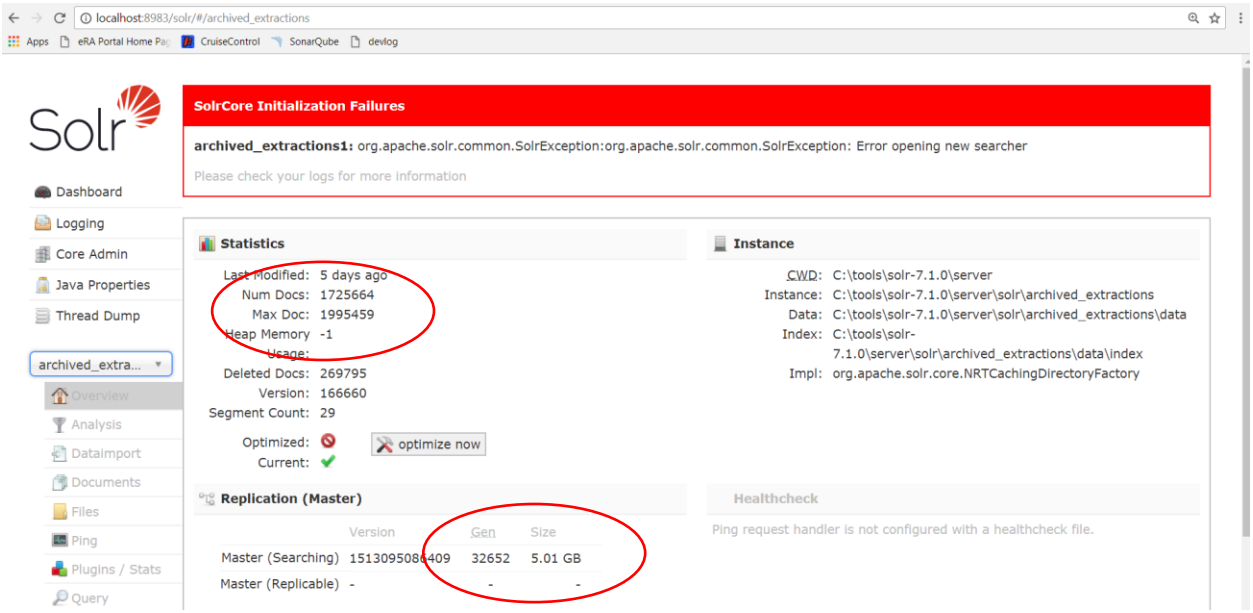
Recall = TP/(TP+FN)

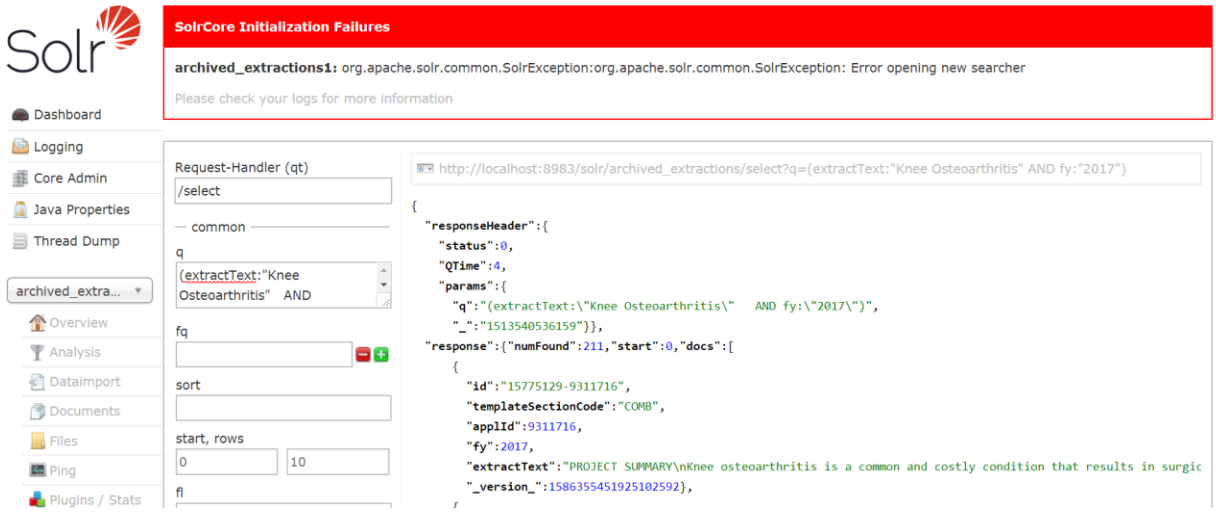
Oracle free text search TP = Count(records) where oracle_fts_flag = 1 and gold_standard_flag = 1
Oracle free text search FP = Count(records) where oracle_fts_flag = 1 and gold_standard_flag = 0

Oracle free text search FN= Count(records) where oracle_fts_flag = 0 and gold_standard_flag = 1

Oracle Precision free text = oracle fts TP/(oracle fts TP +oracle fts FP)

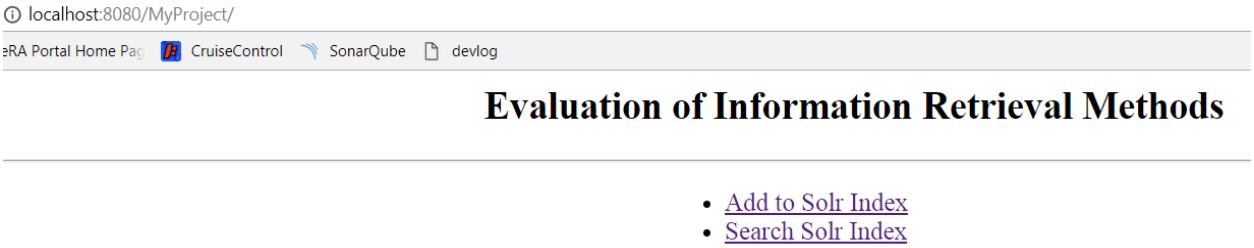
4. Solr Admin Screenshots



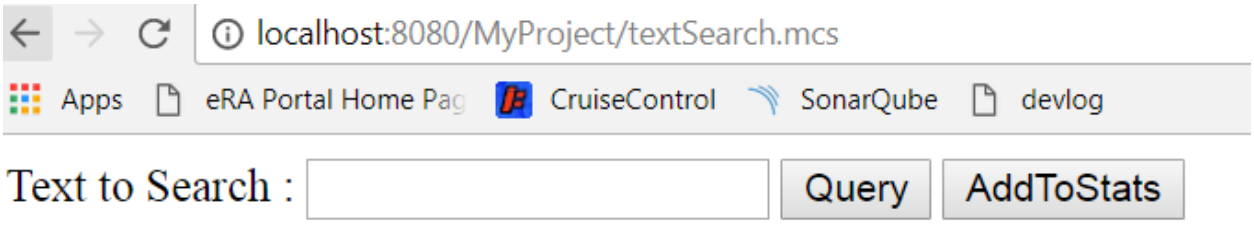


5. Evaluation Framework Webpage Screenshots

Evaluation Framework Home Page



Evaluation Framework Search Solr Index page



localhost:8080/MyProject/searchTerm.mcs?searchTerm=Knee%20Osteoarthritis

Apps | eRA Portal Home Pa | CruiseControl | SonarQube | devlog

Text to Search : Knee Osteoarthritis | Query | AddToStats

Found 198 appls

Application ID	
9218	This laboratory study will evaluate whether a cannabinoid enhances the analgesic efficacy of an opioid in a clinical sample of chronic pain patients, laying the foundation for a larger clinical trial evaluating the utility of cannabinoids as potentially opioid sparing adjuvants to chronic opioid therapy for pain. Heightened CNS processing, or central s[redacted] which a cannabinoid enhances opioid analgesia in a model chronic pain population (e.g., patients with knee osteoarthritis). Human laboratory studies are an ideal method for this initial evaluation because they provide the opportunity to tightly control study drug dosing and afford high sensitivity of measurements, and enable tight control of con[redacted]