Evaluation of IR Models

CSE 535- Information Retrieval

Nuka Saranya Illa
50248926

saranya@buffalo.edu

15-Nov-2018

Introduction: The goal of this project is to implement various IR models names Vector Space Model, Best Matching 25 and Divergence from Randomness and evaluate the IR system so as to improve the search result based on understanding and implementation of the same. The dataset used here is the collection of 3440 tweets from twitter and is index on solr in 3 different languages. We use Trec Evaluation System to analyze the performance of these different models.

Model Implementation: Individual cores are created for each module.The following fields are commonly indexed in all the cores.

```
<field name="id" type="string" multiValued="false" indexed="true" required="true" stored="true"/>
<field name="lang" type="strings"/>
<field name="text_de" type="text_de"/>
<field name="text_en" type="text_en"/>
<field name="text_ru" type="text_ru"/>
<field name="tweet_hashtags" type="strings" multiValued="true"/>
<field name="tweet_urls" type="strings" multiValued="true"/>
```

Similarity is a Lucene class used to score a document in searching. The <similarity/> element can be added either globally or specifically to an individual field type.

We improve the IR models from their default settings in a way that the Mean Average Precision and nDCG values obtained by TREC Evaluation are increased. We train our models on 15 queries and 3440 documents to get parameters.

Mean Average precision (MAP) is the mean of the average precision scores for each query calculated over a set of queries. (here 15 queries). MAP values are based on the number of relevant and retrieved document values for an individual query that in turn determine the recall and precision values averaged at certain points over all the given queries. In order to improve the number of documents that are going to be retrieved and as well as relevant, we do the following changes.An approach that has seen increasing adoption, especially when employed with machine learning approaches to ranking svm-ranking is measures of *cumulative gain* , and in particular ***normalized discounted cumulative gain*** (*NDCG*). NDCG is designed for situations of non-binary notions of relevance Like precision at k , it is evaluated over some number k of top search results.

In this report, we only work on the improvement of MAP Values and tune accordingly.

Methods to improve the MAP Value

Query Expansion: To give more weightage for the important words in the queries, add only those words to synonyms.txt and re-index the files after making relevant changes in the schema.xml

URL Filtering: As irrelevant urls inside text fields negatively contribute to the MAP Score, we remove them in the following way. We write this inside <analyzer> for every fieldType. We have also removed '@' and '#'.

```
<charFilter class="solr.PatternReplaceCharFilterFactory" pattern=
"([@#][\w\d]+)\s|(http(s)?://([A-Z0-9a-z._-]*(/)?)*)(\s)?" replacement=""/>
```

Boosting Query: We use advanced Query parser called as **dismax** to boost the scores by giving certain weights to the fields with respect to their language so as to retrieve more documents

from the query language. For easy computation of scores, python code is used to automate the process. Example: For a query which is originally in English, text_en is boosted more than text_ru and text_de.

http://localhost:8983/solr/BM/select?defType=edismax&fl=id,score&indent=on&q=Russia%27s%20intervention%20in%20Syria&rows=3440&wt=json&ps=3&qf=tweet_urls^2.0%20_text_^1.2%20tweet_hashtags^1.2%20text_en^0.2%20text_de^0.1%20text_ru^0.1

Better Indexing Techniques:

Adding extra fields with different analyzers and tokenizers to improve search result.

- **Multilingual Search**: We do Query Language Translation by creating a separate field for the 3 different languages. We do this to match query terms not in its language text field but also in other language fields. These would improve the number of retrieved documents.

```
<copyField source="text_ru" dest="text_ru_de"/>
<copyField source="text_ru" dest="text_ru_en"/>
<copyField source="text_de" dest="text_de_en"/>
<copyField source="text_de" dest="text_de_ru"/>
<copyField source="text_en" dest="text_en_ru"/>
<copyField source="text_en" dest="text_en_de"/>
```

- **Exact Case Matching:** The documents with exact query case terms are scored higher. Index the content twice, using different fields with different field Types (and different analyzers associated with those field Types). One analyzer will contain a lowercase filters for case-insensitive matches, and one will preserve case for exact-case matches. Here we have also restricted to have no synonyms, stemming filters.

```
<copyField source="text_de" dest="ECM_de"/>
<copyField source="text_en" dest="ECM_en"/>
<copyField source="text_en" dest="ECM_ru"/>

<fieldType name="ECM_de" class="solr.TextField" positionIncrementGap="100">
 <analyzer type="index">
   <charFilter class="solr.PatternReplaceCharFilterFactory" pattern=
   "([@#][\w\d]+)\s|(http(s)?://([A-Z0-9a-z._-]*(/)?)*)(\s)?" replacement=""/>
   <tokenizer class="solr.StandardTokenizerFactory"/>
   <filter class="solr.KeywordMarkerFilterFactory" protected="protwords.txt"/>
   <filter class="solr.StopFilterFactory" format="snowball" words="lang/stopwords_de.txt" ignoreCase="true"/>
 </analyzer>
 <analyzer type="query">
   <charFilter class="solr.PatternReplaceCharFilterFactory" pattern=
   "([@#][\w\d]+)\s|(http(s)?://([A-Z0-9a-z._-]*(/)?)*)(\s)?" replacement=""/>
   <tokenizer class="solr.StandardTokenizerFactory"/>
   <filter class="solr.KeywordMarkerFilterFactory" protected="protwords.txt"/>
   <filter class="solr.StopFilterFactory" format="snowball" words="lang/stopwords_de.txt" ignoreCase="true"/>
 </analyzer>
 <similarity class="solr.BM25SimilarityFactory">
   <str name="b">0.99</str>
   <str name="k1">0.1</str>
 </similarity>
</fieldType>
```

Modify the dismax query in a way that Exact case match values are given more weight than the other fields.

http://localhost:8983/solr/'+model+'/select?defType=dismax&fl=id,score&indent=on&q=RT%20@Free_Media_Hub&rows='+str(rows)+'&wt=json&qf=text_en^'+maxi+'%20text_de^'+min1+'%20text_ru^'+min2+'**%20ECM_en^'+(maxi+0.2)+'%20ECM_de^'+(min1+0.2)+'%20ECM_ru^'+(min2+0.2)+'%20'**

Vector Space Model

This model uses ***ClassicSimilarityFactory*** for its implementation. The following snapshot displays the explicit declaration of the <similarity/> element for text_de fieldtype. We repeat the same for text_ru and text_en fieldTypes inside schema.xml file of VSM Collection.

```
<fieldType name="text_de" class="solr.TextField" positionIncrementGap="100">
  <analyzer>
    <tokenizer class="solr.StandardTokenizerFactory"/>
    <filter class="solr.LowerCaseFilterFactory"/>
    <filter class="solr.StopFilterFactory" format="snowball" words="lang/stopwords_de.txt" ignoreCase="true"/>
    <filter class="solr.GermanNormalizationFilterFactory"/>
    <filter class="solr.GermanLightStemFilterFactory"/>
  </analyzer>
  <similarity class="solr.ClassicSimilarityFactory"></similarity>
</fieldType>
```

The map value of the default settings of VSM is **0.7280,** when evaluated on Trec-Eval System.

```
num_q           all     15
num_ret         all     9203
num_rel         all     225
num_rel_ret     all     222
map             all     0.7280
gm_ap           all     0.6908
R-prec          all     0.6715
bpref           all     0.8079
```

Tokenizer: Though different types of tokenizers have been tested on VSM model there is no much difference found in the MAP values. Hence, we proceed with the default setting**,** *Standard Tokenizer***.**

Stopword filter: The presence or absence of stop words during indexing and querying have made no much difference in the MAP value. So we proceed with the default setting of having stop words.

Stemming and lemmatizationFilters: It is observed that upon changing different parameters for stemming such as *GermanLightStemFilterFactory, RussianLightStemFilterFactoty* and *SnowballPorterFilterFactory* , there is no much difference observed in the MAP values, hence we proceed with the default parameter settings of Stemming and lemmatization.

Synonyms: Addition of synonym files in VSM has improved the score of the Map to **0.7282.**

This is because a document may contain the information need but may not be worded as in the query, there are high chances of the synonyms of the query words to match with the words in document.

```
num_q             all     15
num_ret           all     9248
num_rel           all     225
num_rel_ret       all     222
map               all     0.7282
gm_ap             all     0.6910
R-prec            all     0.6750
```

Multi-lingual Search: Even after changing the index structure to add fields in different languages, VSM showed no change in the values of MAP. The value remained to be 0.7280 as we are not making changes in the weights of the fields matched which need to be done using dismax query parser.

URL Filtering: It is observed that there is no much change observed even after the addition of url filtering to VSM model and also on the removal of '@' and '#'.

Boosting Fields: The values are set in a way that the documents with same query language words are scored both. QL1 will be the weight for the weight of the text_lang field where lang = language of the query. QL2 and QL3 are changed accordingly to increase the MAP values. QL1 > QL2> QL3. (Query boosting)_text_ and tweet_urls are given a weight of 1. Also, tweet_hastags is given the maximum weight(QL1) and the values are plotted for different weights of text_de, text_en,text_ru.n Slop is set to 3

Using edismax increases the number of docs that are going to be retrieved but the precision can be compromised leading to less MAP scores.

QL1	QL2	QL3	MAP
0.2	0.1	0.1	0.7084
1.5	1.2	0.2	0.6357
0.25	0.15	0.05	0.7038

Divergence from Randomness

This method uses DFRSimilarityFactory class and the following values are chosen as default. We do the same for text_en and text_ru as well.

```
<fieldType name="text_de" class="solr.TextField" positionIncrementGap="100">
  <analyzer>
    <tokenizer class="solr.StandardTokenizerFactory"/>
    <filter class="solr.LowerCaseFilterFactory"/>
    <filter class="solr.StopFilterFactory" format="snowball" words="lang/stopwords_de.txt" ignoreCase="true"/>
    <filter class="solr.GermanNormalizationFilterFactory"/>
    <filter class="solr.GermanLightStemFilterFactory"/>
  </analyzer>
  <similarity class="solr.DFRSimilarityFactory">
    <str name="normalization">H2</str>
    <str name="afterEffect">B</str>
    <str name="basicModel">G</str>
    <float name="c">1</float>
  </similarity>
</fieldType>
```

With the above setting as default, the MAP value obtained was **0.7204**

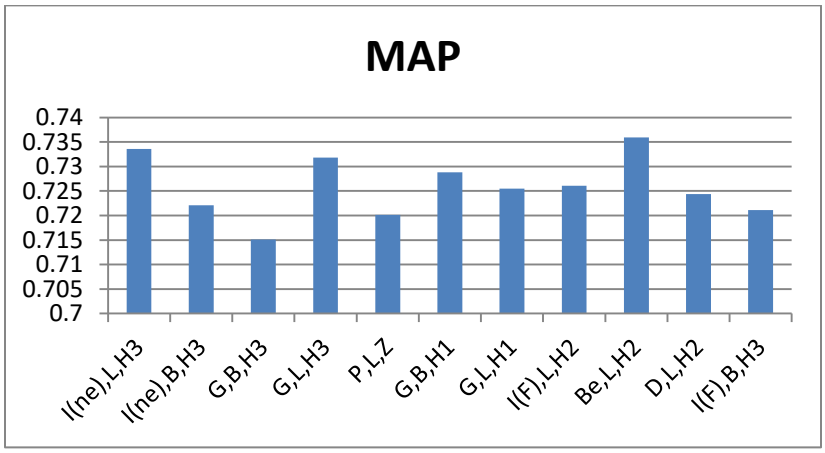
```
num_q             all     15
num_ret           all     9203
num_rel           all     225
num_rel_ret       all     222
map               all     0.7204
gm_ap             all     0.6782
R-prec            all     0.6634
bpref             all     0.8328
```

The DFR scoring formula is composed of three separate components: the *basic model*, the *aftereffect* and an additional *normalization* component, represented by the classes BasicModel, AfterEffect and Normalization, respectively.

1. BasicModel: Basic model of information content:
    a. BasicModelBE: Limiting form of Bose-Einstein
    b. BasicModelG: Geometric approximation of Bose-Einstein
    c. BasicModelP: Poisson approximation of the Binomial
    d. BasicModelD: Divergence approximation of the Binomial
    e. BasicModelIn: Inverse document frequency
    f. BasicModelIne: Inverse expected document frequency [mixture of Poisson and IDF]
    g. BasicModelIF: Inverse term frequency [approximation of I(ne)]
2. AfterEffect: First normalization of information gain:
    a. AfterEffectL: Laplace's law of succession
    b. AfterEffectB: Ratio of two Bernoulli processes
    c. AfterEffect.NoAfterEffect: no first normalization
3. Normalization: Second (length) normalization:
    a. NormalizationH1: Uniform distribution of term frequency
    b. NormalizationH2: term frequency density inversely related to length
    c. NormalizationH3: term frequency normalization provided by Dirichlet prior
    d. NormalizationZ: term frequency normalization provided by a Zipfian relation
    e. Normalization.NoNormalization: no second normalization

Basic model	After Effect	Normalization	Map value
I(ne)	L	H3	0.7336
I(ne)	B	H3	0.7221
G	B	H3	0.7151
G	L	H3	0.7318
P	L	Z	0.7201
G	B	H1	0.7288
G	L	H1	0.7255
I(F)	L	H2	0.7261
Be	L	H2	0.7359
D	L	H2	0.7244
I(F)	B	H3	0.7211

The above table gives different MAP values for different parameters of basic Model. AfterEffect and Normalization. It is observed that the setting of Be,L,H2 gives best values next to I(ne),L,H3. The above is the table with constant float "c" of value 3 for H2 Normalization.



Synonyms: After adding the same synonyms, the MAP value got increased to **0.7387**.

```
num_q           all     15
num_ret         all     9248
num_rel         all     225
num_rel_ret     all     222
map             all     0.7387
gm_ap           all     0.6974
R-prec          all     0.7212
bpref           all     0.8505
```

Using edismax query parser: The values are set in a way that the documents with same query language words are scored both. QL1 will be the weight for the weight of the text_lang field where lang = language of the query. QL2 and QL3 are changed accordingly to increase the MAP values. QL1 > QL2> QL3. (Query boosting)._text_ and tweet_urls are given a weight of 1. Also, tweet_hastags is given the maximum weight (QL1) and the values are plotted for different weights of text_de, text_en,text_ru.

QL1	QL2	QL3	MAP
0.2	0.1	0.1	0.7247
3	2	0.1	0.7007
1.5	1.2	0.2	0.7121
1.5	1.5	1.5	0.7230

Best Matching 25

This model uses BM25SimilarityFactory.It is implicitly implemented by Solr 6.6.5 and takes the default value of k =1.2 and b=0.75 . The following snapshot displays the explicit declaration of the <similarity/> element for text_de fieldtype. We repeat the same for text_ru and text_en fieldTypes inside schema.xml file of BM Collection.

```
<fieldType name="text_de" class="solr.TextField" positionIncrementGap="100">
  <analyzer>
    <tokenizer class="solr.StandardTokenizerFactory"/>
    <filter class="solr.LowerCaseFilterFactory"/>
    <filter class="solr.StopFilterFactory" format="snowball" words="lang/stopwords_de.txt" ignoreCase="true"/>
    <filter class="solr.GermanNormalizationFilterFactory"/>
    <filter class="solr.GermanLightStemFilterFactory"/>
  </analyzer>
  <similarity class="solr.BM25SimilarityFactory">
  <float name="k1">1.2</float>
  <float name="b">0.75</float>
  </similarity>
</fieldType>
```

With the default value setting (k1=1.2 and b=0.75) the MAP value obtained is **0.7230**.

```
num_q           all     15
num_ret         all     9203
num_rel         all     225
num_rel_ret     all     222
map             all     0.7230
gm_ap           all     0.6826
R-prec          all     0.6679
bpref           all     0.8315
```

The following is the formula to determine scores using BM25SimilarityFactory, where

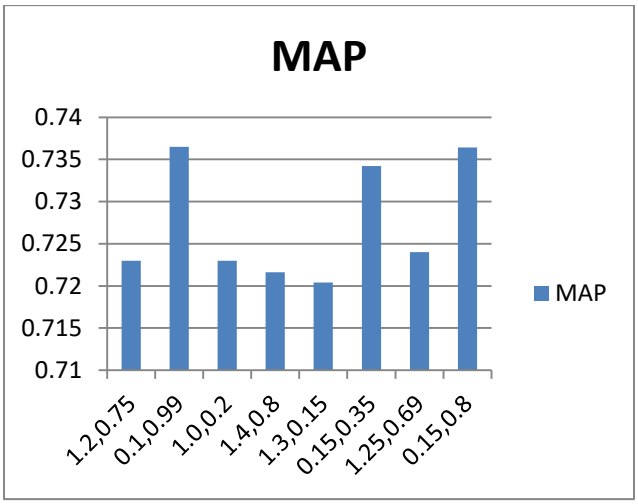
k1 - Controls non-linear term frequency normalization, b - Controls to what degree document length normalizes tf values.

$$IDF * ((k + 1) * tf) / (k * (1.0 - b + b * (|d|/avgDl)) + tf)$$

The below MAP values are tabulated for different k1 and b values. It is observed that for the k1, b values less than their default values gave better improvements in the MAP score for the given set of data

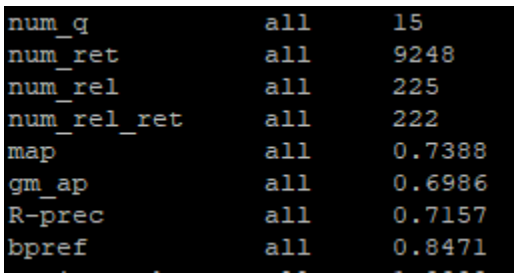
k1	b	MAP
1.2	0.75	0.7230
0.1	0.99	0.7365
1.0	0.2	0.7230
1.4	0.8	0.7216
1.3	0.15	0.7204
0.15	0.35	0.7342
1.25	0.69	0.7240
0.15	0.8	0.7364

Graph is plotted accordingly.



URL Filtering and Multilingual Search: Keeping the K1,b values as 0.15 and 0.8,the is no change observed in the MAP value by url filtering and multilingual search.

Synonyms: After adding the same synonyms.txt file and tested at the tuned parameters k1,b at 0.15 and 0.8r respectively, the MAP value got increased to **0.7388.**



```
num_q             all     15
num_ret           all     9248
num_rel           all     225
num_rel_ret       all     222
map               all     0.7388
gm_ap             all     0.6986
R-prec            all     0.7157
bpref             all     0.8471
```

Using edismax query parser: The values are set in a way that the documents with same query language words are scored both. QL1 will be the weight for the weight of the text_lang field where lang = language of the query. QL2 and QL3 are changed accordingly to increase the MAP values. QL1 > QL2> QL3. (Query boosting)._text_ and tweet_urls are given a weight of 1. Also, tweet_hastags is given the maximum weight (QL1) and the values are plotted for different weights of text_de, text_en,text_ru. As edismax query does Boolean OR, precision gets decreased and hence we see the decreased values of MAP.

QL1	QL2	QL3	MAP
0.2	0.1	0.1	0.7247
3	2	0.1	0.7007
1.5	1.2	0.2	0.7121
1.5	1.5	1.5	0.7230

Conclusion

The final settings of the BM25.

K1= 0.15, b=0.8

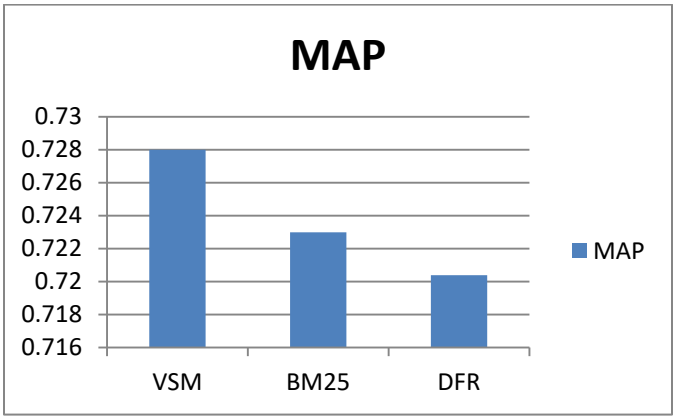
The final settings of the DFR.

AfterEffect L : Basic Model: Be, Normalization: H2,float= 3

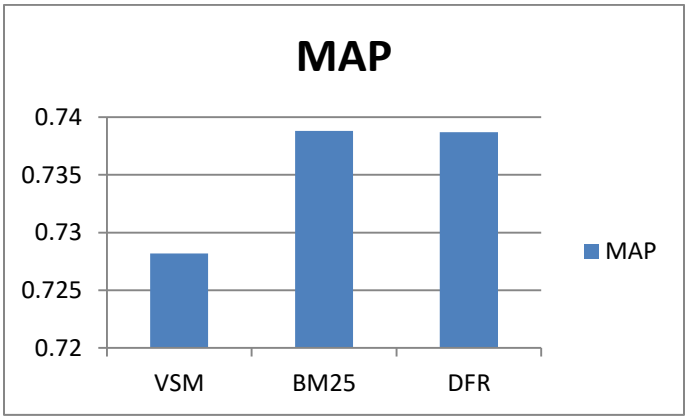
Stopwords and synonym files are included for all three models.

All three models are re-indexed for Multi-lingual Search and Exact Case matching.

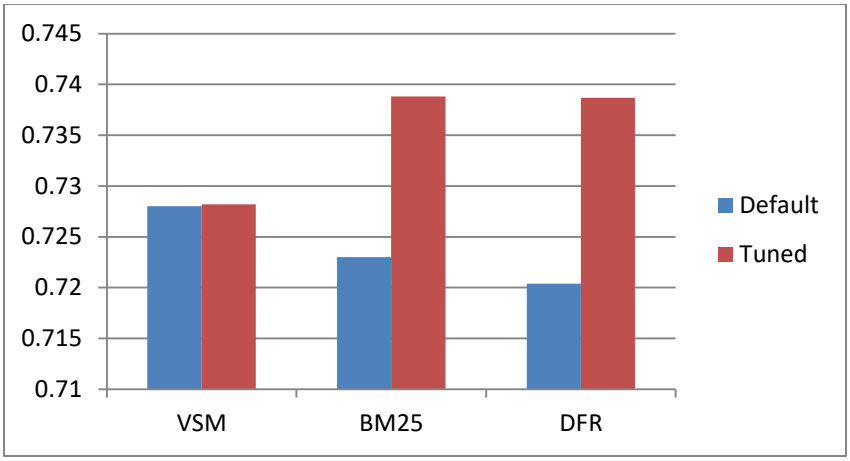
Comparison of IR models on default settings:



Comparison of IR models on final settings:



Comparison of default and final MAP values of each model :



Attempts are made to train the models in a way that the parameters are tuned so as to generalize the data by avoiding over-fitting or under-fitting of the model.