CSE 535 - INFORMATION RETRIEVAL

PROJECT 3

EVALUATION OF IR MODELS

Implementing the Default Configurations of the IR Models

VSM - Vector Space Model

Using the following Similarity class in the schema.xml file we can implement the Vector Space Model as a global configuration :

```
<similarity class="solr.ClassicSimilarityFactory"/>
```

After re-indexing the train.json provided for the above configured schema.xml file for the core on solr, we run the TREC_eval to get the MAP and nDCG values for the test queries provided to us along with their manual judgement file qrel.txt

```
./trec_eval -q -c -M1000 ../qrel.txt ../vsm.txt | grep map
```

```
./trec_eval -q -c -M 1000 -m ndcg ../qrel.txt ../vsm.txt
```

The screenshot for the above is as shown below :

```
timberlake {~/IR/trec_eval.9.0} > ./trec_eval -q -c -M1000 ../qrel.txt ../vsm11.txt | grep map
map                     001     0.3278
map                     002     0.4011
map                     003     0.5729
map                     004     0.6091
map                     005     0.5000
map                     006     0.4936
map                     007     0.8333
map                     008     1.0000
map                     009     1.0000
map                     010     1.0000
map                     011     1.0000
map                     012     0.4615
map                     013     0.1098
map                     014     0.6034
map                     015     0.7721
map                     016     0.8426
map                     017     0.2857
map                     018     0.6110
map                     019     1.0000
map                     020     0.4118
map                     all     0.6418
gm_map                  all     0.5708
```

Figure1

```
timberlake {~/IR/trec_eval.9.0} > ./trec_eval -q -c -M 1000 -m ndcg ../qrel.txt ../vsm11.txt
ndcg                    001     0.6932
ndcg                    002     0.6033
ndcg                    003     0.8620
ndcg                    004     0.8727
ndcg                    005     0.7244
ndcg                    006     0.7345
ndcg                    007     0.9639
ndcg                    008     1.0000
ndcg                    009     1.0000
ndcg                    010     1.0000
ndcg                    011     1.0000
ndcg                    012     0.8035
ndcg                    013     0.4346
ndcg                    014     0.8105
ndcg                    015     0.8979
ndcg                    016     0.9796
ndcg                    017     0.6978
ndcg                    018     0.7837
ndcg                    019     1.0000
ndcg                    020     0.7665
ndcg                    all     0.8314
```

Figure2

BM25 Model

Using the following Similarity class in the schema.xml file we can implement BM25 model :

```
<similarity class="solr.BM25SimilarityFactory">
    <str name="b">0.75</str>
    <str name="k1">1.2</str>
</similarity>
```

We use the below to get the BM25 MAP values and nDCG values for the default configuration :

```
./trec_eval -q -c -M1000 ../qrel.txt ../bm25.txt | grep map
./trec_eval -q -c -M 1000 -m ndcg ../qrel.txt ../bm25.txt
```

```
timberlake {~/IR/trec_eval.9.0} > ./trec_eval -q -c -M1000 ../qrel.txt ../bm25.txt | grep map
map                     001     0.3418
map                     002     0.3913
map                     003     0.5729
map                     004     0.6130
map                     005     0.5000
map                     006     0.4926
map                     007     0.8333
map                     008     1.0000
map                     009     1.0000
map                     010     1.0000
map                     011     1.0000
map                     012     0.6586
map                     013     0.1022
map                     014     0.5577
map                     015     0.8667
map                     016     0.9107
map                     017     0.2857
map                     018     0.6110
map                     019     1.0000
map                     020     0.4118
map                     all     0.6575
gm_map                  all     0.5829
```

Figure3

DFR - Divergence From Randomness

Using the following Similarity class in the schema.xml file we can implement BM25 model :

```
<similarity class="solr.DFRSimilarityFactory">
    <str name="c">1.0</str>
    <str name="normalization">H1</str>
    <str name="afterEffect">B</str>
    <str name="basicModel">G</str>
</similarity>
```

We use the below to get the BM25 MAP values and nDCG values for the default configuration :

```
./trec_eval -q -c -M1000 ../qrel.txt ../dfr.txt | grep map
./trec_eval -q -c -M 1000 -m ndcg ../qrel.txt ../dfr.txt
```

```
timberlake {~/IR/trec_eval.9.0} > ./trec_eval -q -c -M1000 ../qrel.txt ../dfr.txt | grep map
map                     001     0.3735
map                     002     0.3923
map                     003     0.5471
map                     004     0.6130
map                     005     0.5000
map                     006     0.4991
map                     007     0.8333
map                     008     1.0000
map                     009     1.0000
map                     010     1.0000
map                     011     0.9861
map                     012     0.7495
map                     013     0.1041
map                     014     0.5942
map                     015     0.8667
map                     016     0.8626
map                     017     0.2857
map                     018     0.6168
map                     019     1.0000
map                     020     0.4118
map                     all     0.6618
gm_map                  all     0.5891
```

Figure 4

Using MAP and nDCG for optimizing the model's default settings.

Mean average precision for a set of queries is the mean of the average precision scores for each query.

$$\mathrm{MAP} = \frac{\sum_{q=1}^{Q} \mathrm{AveP(q)}}{Q}$$

where *Q* is the number of queries.

nDCG - Normalized Discounted Cumulative Gain

DCG measures the usefulness or gain of a document based on its position in the result list. The gain is accumulated from the top of the result list to the bottom with the gain of each result discounted at lower ranks. Search result lists vary in length depending on the query,so the cumulative gain at each position for a chosen value of **p** should be normalized across queries.

$$\mathrm{nDCG_p} = \frac{DCG_p}{IDCG_p}$$ where $$\mathrm{IDCG_p} = \sum_{i=1}^{|REL|} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$ where |REL| represents the list of relevant documents (ordered by their relevance) in the corpus to position p

We try to improve the MAP values and the nDCG values of the whole system by making changes such as changing the parameters, analyzers, tokenizers and boosting the queries which is presented and shown below.

I)VSM

In the vector space model, both Documents and queries are represented as vectors in a high dimensional space.Every document j is not viewed as a vector of wf * idf values.

OPTIMIZED CONFIGURATION:

Using dismax query with different weightage to the different text fields and setting the value of phrase slop(ps) as 3:

- ➔ *text_en= 1.5*
- ➔ *text_de= 1.2*
- ➔ *text_ru= 0.2*

```
dict1={'q':query.replace(':',
''),'fl':'id,score','wt':'json','indent':'true','rows':20}

urlencoded=urllib.urlencode(dict1);

inurl='http://35.163.54.72:8983/solr/vsm2/select?defType=dismax&'
+urlencoded+'&qf=text_en^1.5%20text_de^1.2%20text_ru^0.2&wt=json&
ps=3'
```

MEASURE	VALUE(DEFAULT SETTING)	VALUE(OPTIMIZED)
MAP	0.6418	**0.7011**
nDCG	0.8314	**0.8615**

The final value after optimization for VSM model is, **MAP = 0.7011** and **nDGC=0.8615.**

EXPERIMENTS/TRIAL AND ERRORS:

1) Experimenting with dismax query and different weightage to different text fields after setting phrase slop to 3:

wt(text_de)	wt(text_en)	wt(text_ru)	MAP(INITIAL)	MAP(MODIFIED)
1.2	1.5	1.2	0.6418	0.6934
0.8	1.5	0.8	0.6418	0.6932
1.3	1.5	1.3	0.6418	0.6907
1.2	**1.5**	**0.2**	**0.6418**	**0.7011**
0.8	1.5	0.2	0.6418	0.6832

After testing for various weights in the process of query boosting, we got the maximum value of MAP as 0.7011.

2)Experimenting with charFilter to remove “#” and “@” in both the index and query for text_en :

```
<analyzer type="index">
<charFilter class="solr.PatternReplaceCharFilterFactory"
pattern="([@#])" replacement=""/>
</analyzer type>
<analyzer type="query">
<charFilter class="solr.PatternReplaceCharFilterFactory"
pattern="([@#])" replacement=""/>
</analyzer type>
```

MEASURE	VALUE(INITIAL)	VALUE(MODIFIED)
MAP	0.6418	0.6469
nDCG	0.8314	0.8335

MAP and nDCG values increased when the filter was used, but then the values decreased when the filter was used with query boosting.

3)Experimented by adding the similarity class to the text field (with query boosting):

```
<fieldType name="text" class="solr.TextField">
<analyzer
class="org.apache.lucene.analysis.standard.StandardAnalyzer"/>
<similarity class="solr.ClassicSimilarityFactory"/>
</fieldType>
```

MEASURE	VALUE(WITH QUERY BOOSTING)	VALUE(MODIFIED)
MAP	0.7011	0.6803
nDCG	0.8615	0.8528

Experimenting with this similarity class, showed us that it has a negative impact on the MAP and nDCG values.

II)BM25

It is an IR model based on probabilistic retrieval framework.BM25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of the inter-relationship between the query terms within a document.

OPTIMIZED CONFIGURATION:

1) **Tuning the values of parameter k1 and b:**

```
<similarity class="solr.BM25SimilarityFactory">

<float name="k1">1.3</float>

<float name="b">0.39</float>

</similarity>
```

2) **Using URLTokenizer instead of standard tokenizer for text_en for analyzer type query**

```
<analyzer type="query">

<tokenizer class="solr.UAX29URLEmailTokenizerFactory"/>

</analyzer>
```

3) **Using dismax query with equal weightage to all the text fields and setting the value of phrase slop(ps) as 3:**

dict1={'q':query.replace(':', ''),'fl':'id,score','wt':'json','indent':'true','rows':20}

urlencoded=urllib.urlencode(dict1);

inurl='http://35.163.54.72:8983/solr/vsm2/select?defType=dismax&'+urlencoded+'&qf=text_en^1.3%20text_de^1.3%20text_ru^1.3&wt=json&ps=3'

4) Translating queries:

Translating the queries of all three languages , increased the number of documents returned . This in turn increased the map and nDCG value.

MEASURE	VALUE(DEFAULT SETTING)	VALUE(OPTIMIZED)
MAP	0.6575	0.6971
nDCG	0.8376	0.8683

The final value after optimization for BM25 model is, **MAP = 0.6971** and **nDGC=0.8683.**

EXPERIMENTS/TRIAL AND ERRORS:

1) Experimenting with filters/tokenizers:

- ***Using Regular Expression Pattern Tokenizer for text_en(with query boosting):***

```
<analyzer>
  <filter class="solr.PatternReplaceFilterFactory"
  pattern="([^A-Z][^a-z])" replacement="" replace="all"/>
</analyzer>
```

MEASURE	VALUE(INITIAL)	VALUE(MODIFIED)
MAP	0.6575	0.6182
nDCG	0.8376	0.7921

- ***Using UAX29 URL Email Tokenizer(with query boosting):***

```
<analyzer type="query">
   <tokenizer class="solr.UAX29URLEmailTokenizerFactory"/>
</analyzer>
```

MEASURE	VALUE(INITIAL)	VALUE(MODIFIED)
MAP	0.6575	0.6818
nDCG	0.8376	0.8589

After testing with various tokenizers, we obtained better results with URL Email Tokenizer.

2) TUNING THE VALUES OF K1 AND B(without query boosting):

k1 - Controls non-linear term frequency normalization (saturation).

b - Controls to what degree document length normalizes the tf values.

Default values: k1 = 1.2, b = 0.75

- ***Increasing the value of k1 and decreasing the value of B***

K1=1.3, B=0.39

MEASURE	**VALUE(INITIAL)**	**VALUE(MODIFIED)**
MAP	0.6575	0.6598
nDCG	0.8376	0.8367

- ***Increasing the value of K1 and increasing the value of B***

K1=1.3, B=0.79

MEASURE	**VALUE(INITIAL)**	**VALUE(MODIFIED)**
MAP	0.6575	0.6582
nDCG	0.8376	0.8383

- ***Decreasing the value of k1 and decreasing the value of B***

K1=1.1, B=0.39

MEASURE	**VALUE(INITIAL)**	**VALUE(MODIFIED)**
MAP	0.6575	0.6580
nDCG	0.8376	0.8365

- ***Decreasing the value of K1 and increasing the value of B***

K1=1.1, B=0.79

MEASURE	**VALUE(INITIAL)**	**VALUE(MODIFIED)**
MAP	0.6575	0.6553
nDCG	0.8376	0.8370

After testing with many values, we obtained better results when we increased the value of k1 and decreased the value of b.

3) Experimenting with dismax query and different weightage to different text fields after setting phrase slop to 3:

wt(text_de)	wt(text_en)	wt(text_ru)	MAP(initial)	MAP(modified)
1.2	1.5	0.2	0.6575	0.6788
0.8	1.5	0.2	0.6575	0.6351
1.2	1.5	0.9	0.6575	0.6829
1.3	**1.3**	**1.3**	**0.6575**	**0.6912**

After testing with various combinations, we noticed that better results are obtained when all the query terms were weighted the same.

III)DFR (Divergence from Randomness)

The DFR has three parameters **BasicModel** which is the basic model of the information content, **AfterEffect** specifies the first normalization of information gain and **Normalization** refers to the second normalization. A parameter 'c' that controls the term frequency normalization with respect to the document length which is specified for normalization H1 and H2.

1)Tuning parameters for the DFR model

The following table summarises the various values obtained for the change of the parameters for the DFR model using query boosting are as follows:

Qf = text_en^1.3 and text_de^1.2 and text^ru^1.2 and keeping ps =3 as the dismax values for query boosting

Normalization	AfterEffect	BasicModel	C	MAP	nDCG
H2	B	G	7	0.6910	0.8437
H2	B	G	5	0.6921	0.8663
H2	B	G	3	0.6889	0.8635
H2	L	G	4	0.6904	0.8665

H2	L	P	4	0.6900	0.8612
H1	B	G	7	0.6942	0.8667
H1	B	P	3	0.6900	0.8612
H1	B	G	5	0.6944	0.8669
H1	B	G	3	0.6965	0.8659
H3	B	P	-	0.6709	0.8321
Z	L	P	-	0.6896	0.8616
H1	B	G	3	0.6970	0.8654
H1	B	I(F)	7	0.6742	0.8348
H1	B	D	7	0.6949	0.8635

From the above we notice that by decreasing value of c parameter for the H1 and H2 normalization parameters, the MAP value increases and so does the nDCG value, hence we adapt a smaller value of c but not too small.

Similarly trying out various models such as Bose-Einstein, Poisson approximation of B-E, Divergence approximation of the Binomial, AfferEffect as Laplace's law of succession and Ratio of Bernoulli processes, we get the MAP of 0.6970 as the highest.

2)Using query boosting to get better values for MAP and nDCG along with URL Tokenizer

```
<analyzer type="query">
     <tokenizer class="solr.UAX29URLEmailTokenizerFactory"/>
</analyzer>
```

text_en^1.5 and text_de^1.3 and text_ru^1.2

MEASURE	VALUE(INITIAL)	VALUE(MODIFIED)
MAP	0.6970	0.6981
nDCG	0.8654	0.8656

text_en^1.5 and text_de^1.4 and text_ru^1.2

MEASURE	VALUE(INITIAL)	VALUE(MODIFIED)
MAP	0.6970	0.6992
nDCG	0.8654	0.8670

Therefore, by using dismax and doing query boosting we get max MAP value as 0.6992

3)Using query expansion and translating queries

By using query expansion and synonyms match for the given test data, we notice that the MAP and nDCG values increased.

MEASURE	VALUE	VALUE(FINAL)
MAP	0.6992	**0.7013**
nDCG	0.8670	**0.8693**

Therefore, final value after optimization for DFR model obtained was **MAP=0.7013 and nDCG =0.8693.**

SUMMARY

Therefore after the optimization of the default model settings we obtained the following as the MAP and nDCG values :

Model	Initial	Modified
VSM	0.6418	0.7011
BM25	0.6575	0.6971
DFR	0.6618	0.7013

MAP values for the models

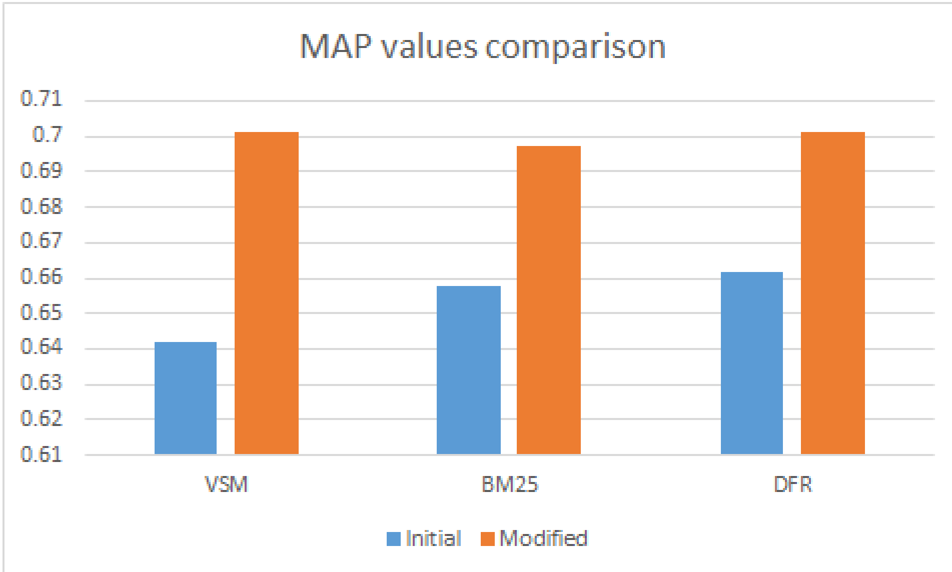

Fig4. MAP Values comparison

Model	Initial	Modified
VSM	0.8314	0.8615
BM25	0.8376	0.8683
DFR	0.8403	0.8693

nDCG values for the models

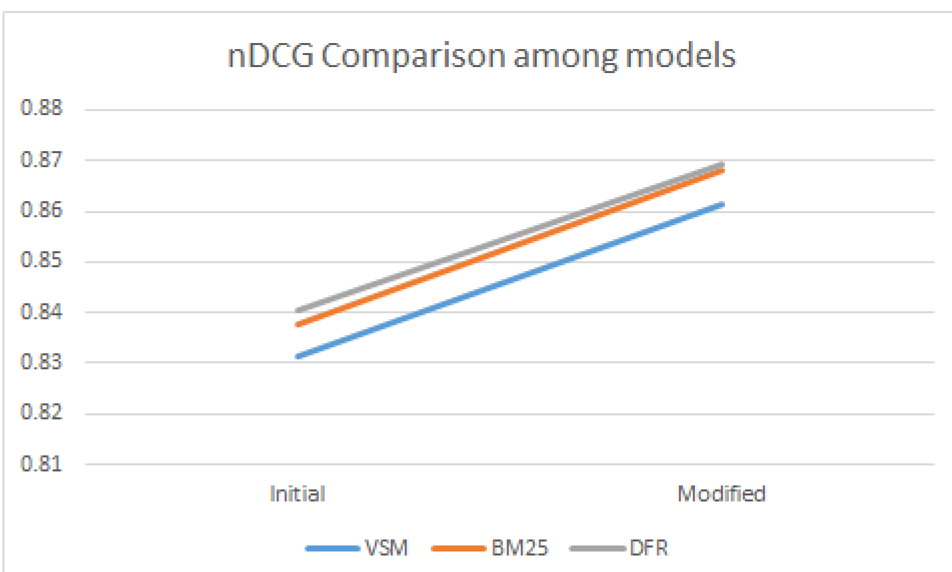

Fig5. nDCG Values comparison