INTRODUCTION TO INFORMATION RETRIEVAL

CSE 535

FALL 18

PROJECT-3 REPORT

EVALUATION OF IR MODELS

SRICHARAN ANAND

50291478

sanand3

EVALUATING OF INFORMATION RETRIEVAL MODELS

1. INTRODUCTION

The project deals with the implementation of 3 IR models: Vector Space model, BM25 and DFR using solr based on given twitter data and the results are evaluated using TREC_eval program.

We are given 15 training queries and 5 testing queries in 3 different languages: English, German and Russian. The main goal of this project is to improve the performance of IR systems by considering MAP(Mean Average Precision) score as the evaluation measure.

2. METHODOLOGY

In this project, our aim to improve the MAP score for the 3 models. Initially, the MAP scores are determined by the standard query parser for the 3 similarity classes. Then the parameters are tuned for the improvement. For further improvement is done by boosting the fields during querying using parsers like dismax and edismax. MAP values are also collected by introducing a list of synonyms upon careful examination of queries.

3. EXPERIMENTS ON THE MODELS:

3.1 DEFAULT CONFIGURATION

For the default configuration, 3 cores are created- one for each model: VSM (Vector Space Model), BM25 and DFR (Divergence From Randomness). The schema.xml files have been modified by introducting the following similarity classes:

- VSM (Vector Space Model)
  In this model, the queries and documents are represented in the form of vector quantities in the multidimensional space, where each indexed

document is the dimension and the corresponding weights are the TF-IDF values. We can implement this model in Solr using the class: *solr.ClassicSimilarity* .

- Okapi BM25:
  This model is a probabilistic IR model. We can implement it in solr using the following class:
  solr.BM25Similarity. This is also the Default similarity for Solr version 6.0 and above. So it can also be defined by: *solr.DefaultSimilarity*

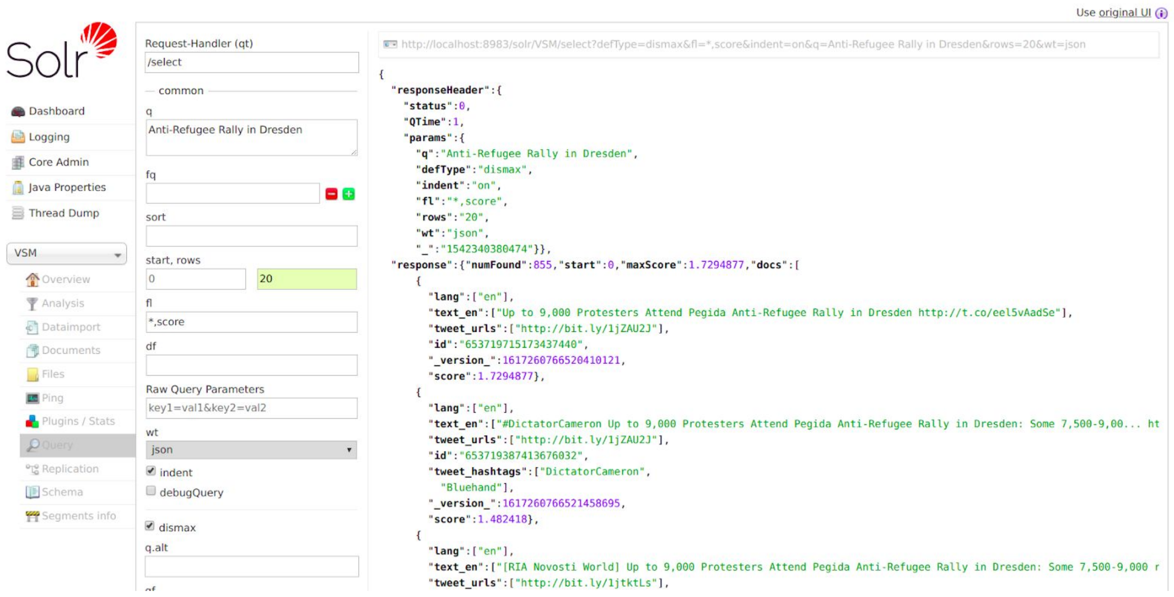
- Divergence from randomness (DFR):
  In this model, the term weight is inversely proportional to the probability of TF. We can implement it in Solr using the following similarity class:
  solr.DFRSimilarityFactory

A python script which was given to parse the queries one by one is modified to generate the output file such that all queries are parsed line by line from text file and read individually and are passed on as single query and the results are stored in the format for TREC evaluation.

Using the default configurations (standard query parser), the following MAP scores were obtained for the training queries:

1. VSM: **0.6639**
2. Okapi BM25(default k=1.2 and b=0.75) : **0.6675**
3. DFR (Default - H2 Normalization, Basic model G and Bernoulli) : **0.6682**

Initially the query was run against the training set and test set. The score seemed to change with minor changes to query. So the it was necessary we tuned the parameters of the model. And the also the need to add synonyms to further increase the MAP score.



Request-Handler (qt): /select
common
q: Anti-Refugee Rally in Dresden
fq:
sort:
start, rows: 0 20
fl: *,score
df:
Raw Query Parameters: key1=val1&key2=val2
wt: json
☑ indent
☐ debugQuery
☑ dismax
q.alt:

http://localhost:8983/solr/VSM/select?defType=dismax&fl=*,score&indent=on&q=Anti-Refugee Rally in Dresden&rows=20&wt=json

```
{
  "responseHeader":{
    "status":0,
    "QTime":1,
    "params":{
      "q":"Anti-Refugee Rally in Dresden",
      "defType":"dismax",
      "indent":"on",
      "fl":"*,score",
      "rows":"20",
      "wt":"json",
      "_":"1542340380474"}},
  "response":{"numFound":855,"start":0,"maxScore":1.7294877,"docs":[
      {
        "lang":["en"],
        "text_en":["Up to 9,000 Protesters Attend Pegida Anti-Refugee Rally in Dresden http://t.co/eel5vAadSe"],
        "tweet_urls":["http://bit.ly/1jZAU2J"],
        "id":"653719715173437440",
        "_version_":1617260766520410121,
        "score":1.7294877},
      {
        "lang":["en"],
        "text_en":["#DictatorCameron Up to 9,000 Protesters Attend Pegida Anti-Refugee Rally in Dresden: Some 7,500-9,00... ht
        "tweet_urls":["http://bit.ly/1jZAU2J"],
        "id":"653719387413676032",
        "tweet_hashtags":["DictatorCameron",
          "Bluehand"],
        "_version_":1617260766521458695,
        "score":1.482418},
      {
        "lang":["en"],
        "text_en":["[RIA Novosti World] Up to 9,000 Protesters Attend Pegida Anti-Refugee Rally in Dresden: Some 7,500-9,000 r
        "tweet_urls":["http://bit.ly/1jtktLs"],
```

Example of solr running a query

3.2 TUNING PARAMETER

3.2.1 Tuning k and b1 for BM25

Using the default setting, we can see that BM25 gives a MAP score of 0.6675 on the training queries for b=0.75 and k1=1.2. From the observations made b value took a range of 0.3 and 0.8. I choose the value b=0.8 and k1=0.3. For the corresponding values of parameters, the MAP score reached a value of 0.6774.

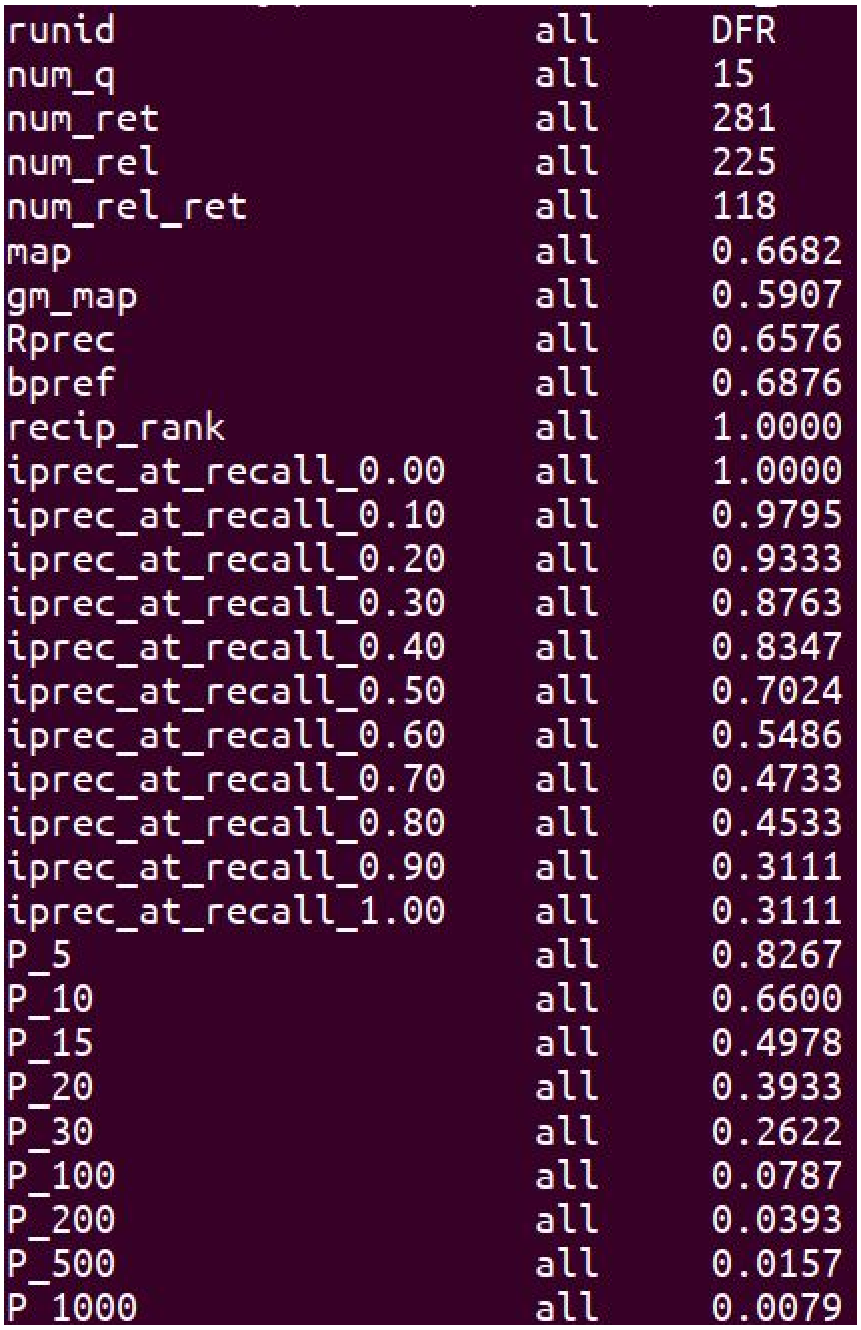
3.2.2 Tuning parameters in DFR

I tried out various combinations of parameters such as normalization, aftereffect and basic model. We started out the implementation using the default settings given to us which are normalization – H2, Aftereffect – B and Basic model – G. We observe that for normalization –Z, aftereffect – B and basic model – G, the results improve very well. So I chose these parameters for the DFR model.

3.2.3 Synonyms List

By providing list of synonyms to all of the cores, I could not see significant change in the MAP score.

The following MAP scores were obtained upon addings synonyms:
DFR-**0.6682**
VSM-**0.6652**
BM-25-**0.6730**



```
runid                   all     DFR
num_q                   all     15
num_ret                 all     281
num_rel                 all     225
num_rel_ret             all     118
map                     all     0.6682
gm_map                  all     0.5907
Rprec                   all     0.6576
bpref                   all     0.6876
recip_rank              all     1.0000
iprec_at_recall_0.00    all     1.0000
iprec_at_recall_0.10    all     0.9795
iprec_at_recall_0.20    all     0.9333
iprec_at_recall_0.30    all     0.8763
iprec_at_recall_0.40    all     0.8347
iprec_at_recall_0.50    all     0.7024
iprec_at_recall_0.60    all     0.5486
iprec_at_recall_0.70    all     0.4733
iprec_at_recall_0.80    all     0.4533
iprec_at_recall_0.90    all     0.3111
iprec_at_recall_1.00    all     0.3111
P_5                     all     0.8267
P_10                    all     0.6600
P_15                    all     0.4978
P_20                    all     0.3933
P_30                    all     0.2622
P_100                   all     0.0787
P_200                   all     0.0393
P_500                   all     0.0157
P_1000                  all     0.0079
```

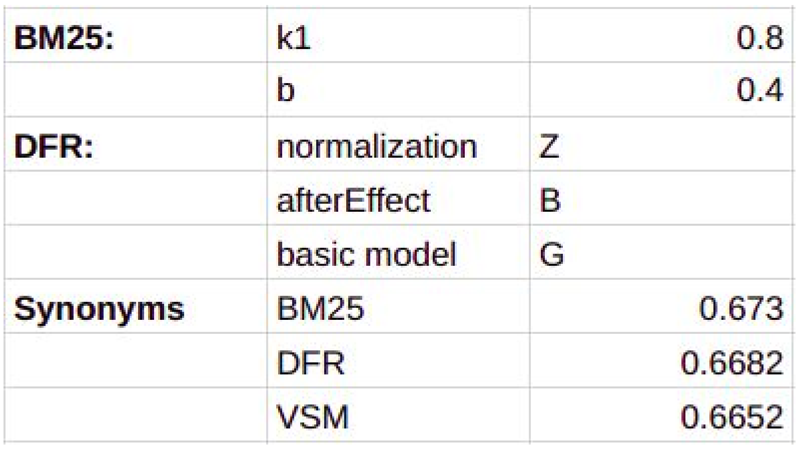
Sample TREC eval score for DFR after tuning

3.3 Creating a custom parser

Extended dismax query parser is the next version of dismax parser. Since the default query processor is data and query independent. There is a need to improve the score and recover more relevant documents from the given query. This can be done by many techniques. One of them is Edismax where we can define a custom parser by extending the class. In the custom parser we can boost the fields based on the query.
I have tried writing my custom parser but could not make it work because of an error. But I have included it in the source and removed the query parser tag in the solrconfig.xml.

4. CONCLUSION

In conclusion we can see that the MAP value can improved by constant experimentation and proper understanding of data and query processing. The following results were collected after constant tuning:



BM25:	k1	0.8
	b	0.4
DFR:	normalization	Z
	afterEffect	B
	basic model	G
Synonyms	BM25	0.673
	DFR	0.6682
	VSM	0.6652

RESULTS