EVALUATION OF INFORMATION RETRIEVAL MODELS

Upasana Ghosh
UBIT Name - upasanag
Department of Computer Science
University at Buffalo
Buffalo, NY 14214
upasanag@buffalo.edu

Introduction:

In this project, we deal with the implementation of different IR models such as the BM25 model, DFR models and the Language Models based on Solr and using the twitter data. The results are evaluated using the TREC_eval tool. We are given 15 training queries and 10 test queries in languages – English, German and Russian. Our main goal in this project is to improve the performance of the IR systems by considering primarily the MAP (Mean Average Precision) score as the evaluation measure.

Experimentation:

Default Setup:

We have created 3 cores – one for each model – BM25, DFR (Divergence from Randomness) and the LM (Learning Models) by modifying the schema.xml file for each model. The following similarity classes have been used for each model:

1. BM25 Model:

   Okapi BM25 model is a probabilistic information retrieval model which was originally designed for short-length documents. In Solr, the similarity class for this is solr.BM25SimilarityFactory.

2. DFR (Divergence from Randomness) Model:

Divergence from Randomness is a framework including multiple models and normalization techniques. They all share the same principle: the term may occur in a document randomly, following a certain distribution. The more a document diverges from our configured random distribution, the higher would be the score. The term-weight is inversely related to the probability of the term-frequency within the document obtained by a model of randomness. In Solr, the similarity class for this is given by solr.DFRSimilarityFactory.

3. LM (Language Models):

A Language model computes the probability that a query is generated by a document. Language models basically revolve over the idea of smoothing scores based on unseen words (i.e. document length). The similarity class used to implement this model in Solr is given by solr.LMDirichletSimilarityFactory

Using the default settings which has the standard query parser, we obtained the following MAP values:

i) BM25 - 0.6985 - default k1 =1.2 and b=0.75
ii) DFR - 0.7055 – given defaults = H2 normalization, Basic model G and Bernoulli
iii) LM - 0.6299 - default mu = 2000

A Python script was developed that parses the queries in the queries.txt file one by one and returns the query results into a new text file for each of the models by running the query URL. Three output text files are generated corresponding to each of the cores (BM25, DFR and LM). Initially we queried against the given sets of training queries and observed the tweets which were returned as results. We noticed the changes in the resultant tweets and the impact on the scores by making minor changes to the queries. We tested out the case sensitivity of the queries and as expected, there were no changes in the results as the queries are changed to lowercase at query time and documents are set to lowercase at indexing time. With the help of the relevant scores provided to us for the training queries, we used the following command on Trec_eval to calculate the MAP value for the three models:

```
trec_eval -q -c -M 1000 qrel.txt <sample_query_output.txt>
```

To improve the MAP value for each of the models, the parameters and the settings of each of the models were tuned.

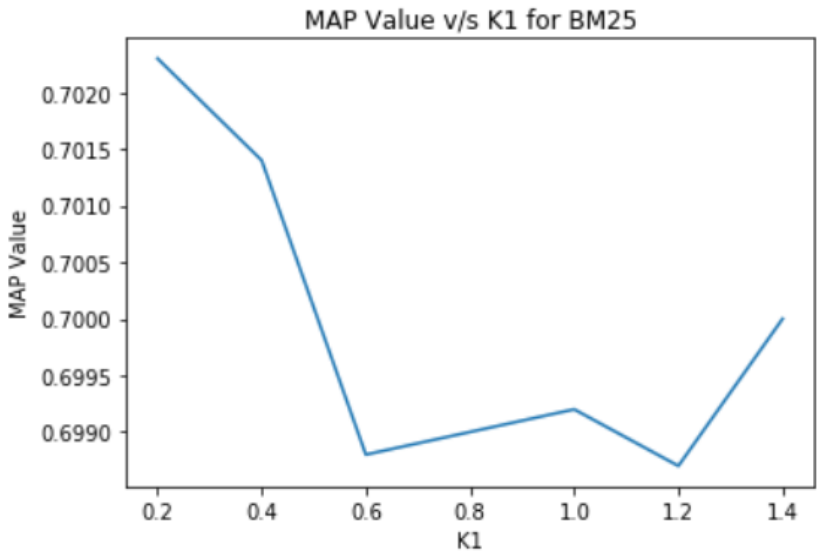
Tuning the parameter values for each model:

The tuning process involves the tweaking of parameters to improve the performance of the models.

- ### Tuning B and K1 values in BM25:

  The BM25 model has the default setting: b = 0.75 and k1 = 1.2. For the default setting we got a MAP = 0.6985. We started off by varying the values of b and keeping k1 = 1.2. We observed that the MAP value remained unchanged at MAP = 0.698. It is generally recommended that the values of b range from 0.5 to 0.8. We chose the value of b = 0.8 and varied k1 from 0.2 to 1.4 as shown below in the table. At k1 = 0.2, a peak in the graph can be observed and the MAP value corresponding to this peak value is 0.7023. Hence, we chose the values of b and k1 as 0.8 and 0.2 for a good MAP value.

K1	B	MAP Value
0.2	**0.8**	**0.7023**
0.4	0.8	0.7014
0.6	0.8	0.6988
1	0.8	0.6992
1.2	0.8	0.6987
1.4	0.8	0.7



```
P_200                   015     0.0650
P_500                   015     0.0260
P_1000                  015     0.0130
runid                   all     BM25
num_q                   all     15
num_ret                 all     280
num_rel                 all     225
num_rel_ret             all     131
map                     all     0.7023
gm_map                  all     0.6343
Rprec                   all     0.6956
bpref                   all     0.7102
recip_rank              all     1.0000
iprec_at_recall_0.00    all     1.0000
iprec_at_recall_0.10    all     0.9667
iprec_at_recall_0.20    all     0.9286
iprec_at_recall_0.30    all     0.8984
iprec_at_recall_0.40    all     0.8480
iprec_at_recall_0.50    all     0.8201
iprec_at_recall_0.60    all     0.7153
iprec_at_recall_0.70    all     0.5399
```

- Tuning Normalization, Aftereffect and Basic Model values in DFR:
  We tried various combinations of parameters such as normalization, aftereffect and basic model. We started the implementation using the default settings provided to us:
  - Normalization – H2
  - Aftereffect – B
  - Basic model – G.

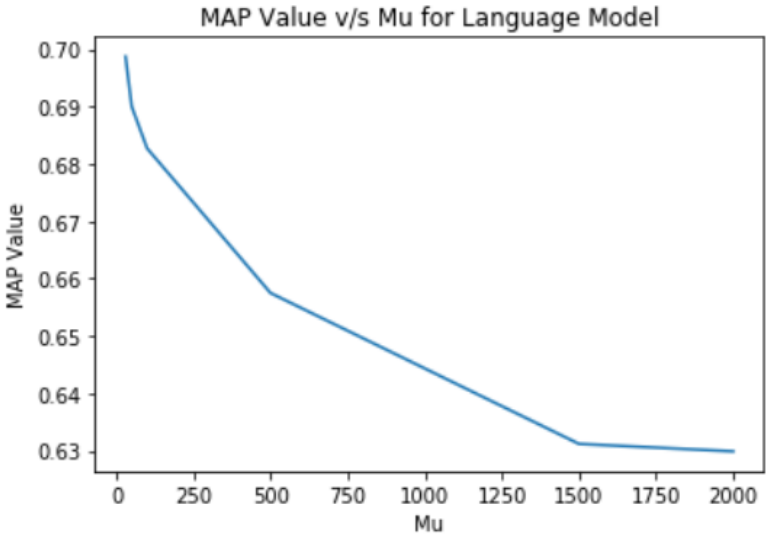
  We found that the results improved very well with the default settings. So, we chose these parameters for the DFR model.

Normalization	Aftereffect	Basic Model	MAP Value
h3	b	i(f)	0.6897
h3	b	g	0.6952
h2	b	i(f)	0.6982
z	b	i(f)	0.6999
z	b	g	0.7019
h2	**b**	**g**	**0.7055**

```
P_500                   015     0.0260
P_1000                  015     0.0130
runid                   all     DFR
num_q                   all     15
num_ret                 all     280
num_rel                 all     225
num_rel_ret             all     130
map                     all     0.7055
gm_map                  all     0.6382
Rprec                   all     0.6890
bpref                   all     0.7124
recip_rank              all     1.0000
iprec_at_recall_0.00    all     1.0000
iprec_at_recall_0.10    all     0.9667
iprec_at_recall_0.20    all     0.9286
iprec_at_recall_0.30    all     0.9009
```

- Tuning Mu values in Learning Model:
  We started off with the default settings for Learning Models which is by taking the value of Mu as 2000. We found that the results improved significantly as we reduced the value of Mu. So, we chose the value of Mu as 30 for the Language Model:

Mu	MAP Value
2000	0.6299
1500	0.6312
500	0.6575
100	0.6827
50	0.69
30	**0.6987**



P_20	015	0.0500
P_30	015	0.4333
P_100	015	0.1300
P_200	015	0.0650
P_500	015	0.0260
P_1000	015	0.0130
runid	all	LanguageModel
num_q	all	15
num_ret	all	280
num_rel	all	225
num_rel_ret	all	130
map	all	0.6987
gm_map	all	0.6274
Rprec	all	0.7011
bpref	all	0.7035
recip_rank	all	1.0000
iprec_at_recall_0.00	all	1.0000
iprec_at_recall_0.10	all	0.9649
iprec_at_recall_0.20	all	0.9286
iprec_at_recall_0.30	all	0.8802
iprec_at_recall_0.40	all	0.8387
iprec_at_recall_0.50	all	0.8121
iprec_at_recall_0.60	all	0.7048
iprec_at_recall_0.70	all	0.5499
iprec_at_recall_0.80	all	0.4437
iprec_at_recall_0.90	all	0.3333

Conclusion:

After enhancing the search engine performance of the models, we have been obtained the following results:

Results		
BM25:	K1	0.2
	b	0.8
DFR:	Normalization	h2
	Aftereffect	b
	Basic Model	g
LM:	Mu	30