Evaluation of IR Models

Miki Padhiary
UBID: mikipadh (50286289)

November 16, 2018

Abstract

The objective of this project is to implement various IR models and evaluate their performance based on Mean Average Precision(MAP) values. A dataset containing twitter data in three different languages(English, German and Russian), 15 sample queries and corresponding sample manual relevance were provided. The twitter data provided were atfirst indexed in Solr and then evaluated by using Trec eval program. The scores obtained gave us an idea of how relevant were the results given by our Solr instance. Different techniques used to obtain greater map values are described in the later sections. The best values obtained for BM25, Divergence From Random(DFR) and Vector Space Model(VSM) are 0.7601, 0.7507 and 0.7507 respectively.

1 Introduction

The main goal of this project is to improve search result of Solr instance using different techniques. We have implemented IR models such as BM25, Divergence From Randomness (DFR) and Vector Space Model(VSM) and used the training queries provided to judge the relevance/precision of our results with the help of the TREC tool. Mean Average Precision(MAP) is used to judge the relevancy of our system. Following are the techniques used:

- Tuning the parameters of IR Models
- Creating a common text_all field and Boosting
- Using various filters and tokenizers
- Using query parsers and query expansion techniques
- Multilingual Support
- Translation of queries and Synonyms

2 Overview of IR Models

2.1 Best Matching(BM25)

The Best Matching (BM25), Okapi Weighting is a probabilistic Information Retrieval (IR) model. It is used by search engines to rank matching documents according to their relevance to given search query.

2.2 Divergence From Randomness(DFR)

The Divergence from Randomness (DFR) paradigm is a generalisation of one of the very first models of Information Retrieval, Harter's 2-Poisson indexing-model. It is based on the following components : Randomness Model, First Normalization and Term Frequency Normalization.

2.3 Vector Space Model(VSM)

Alternative name is Term Vector Model, and is an algebraic model used for representing documents and queries as vectors in the term space. It allows computing a continuous degree of similarity between queries and documents. It ranks documents according to their possible relevance and partial matching.

We have successfully implemented the 3 models and created 3 cores for the same*[Refer Figure 1]*

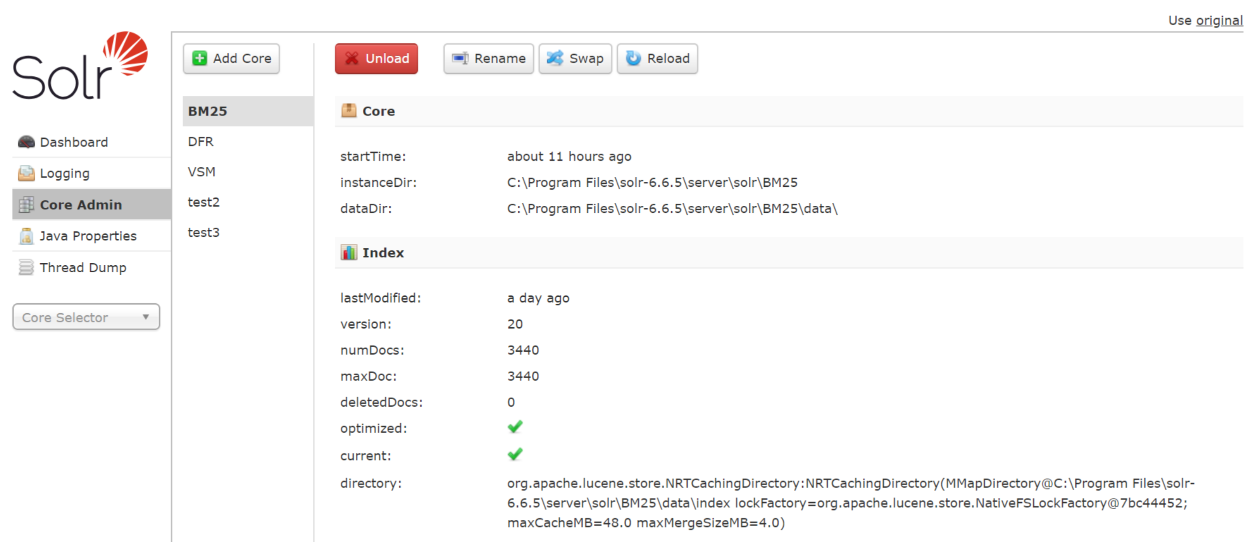


Figure 1: Successful Implementation of Various IR Models

3 Experiments and Results

3.1 Tuning the parameters of IR Models

3.1.1 Best Matching(BM25)

The default similarity class in solr-6.2.0 is BM25. If no similarity class is added to the schema, then the default is **BM25SimilarityFactory**. However, we can add BM25SimilarityFactory to the schema to tweak the values of k1 and b

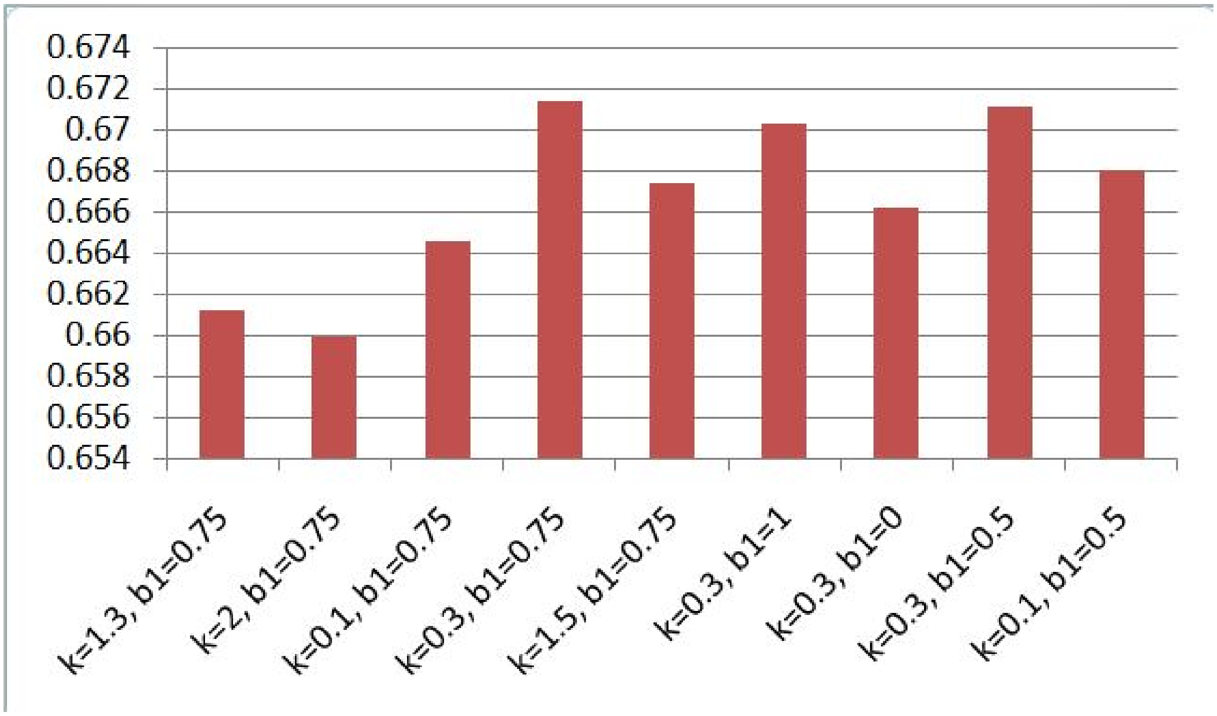
```
<similarity class="solr.BM25SimilarityFactory">
  <float name="k1">1.2</float>
  <float name="b">0.75</float>
</similarity>
```

The default MAP and nDCG score obtained were **0.6636 and 0.8292** as shown below:

map	001	0.2854
map	002	0.4202
map	003	0.5729
map	004	0.5724
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	1.0000
map	012	0.6616
map	013	0.1041
map	014	0.6386
map	015	0.8667
map	all	0.6636
gm_map	all	0.5838

ndcg	001	0.5676
ndcg	002	0.6143
ndcg	003	0.8672
ndcg	004	0.8590
ndcg	005	0.7244
ndcg	006	0.7126
ndcg	007	0.9639
ndcg	008	1.0000
ndcg	009	0.9931
ndcg	010	1.0000
ndcg	011	1.0000
ndcg	012	0.8972
ndcg	013	0.4312
ndcg	014	0.8673
ndcg	015	0.9407
ndcg	all	0.8292

Below is the graph that represents MAP values for different values of **k1** and **b** for 20 rows



Results for BM25: Usually the default values should suffice for BM25 model. The parameter b controls to what degree document length normalizes tf values, we got the best values for MAP when keeping b to 0.75. Reason is the document length for this corpus is more or less equal throughout, without much variance, hence the default value of b works good.
The optimum MAP and nDCG value was obtained keeping k1 = 0.3 and b = 0.75.
The MAP and nDCG values obtained through trec_eval were **0.6714 and 0.8428**, if 20 documents were retrieved.

3.1.2 Divergence From Randomness(DFR)

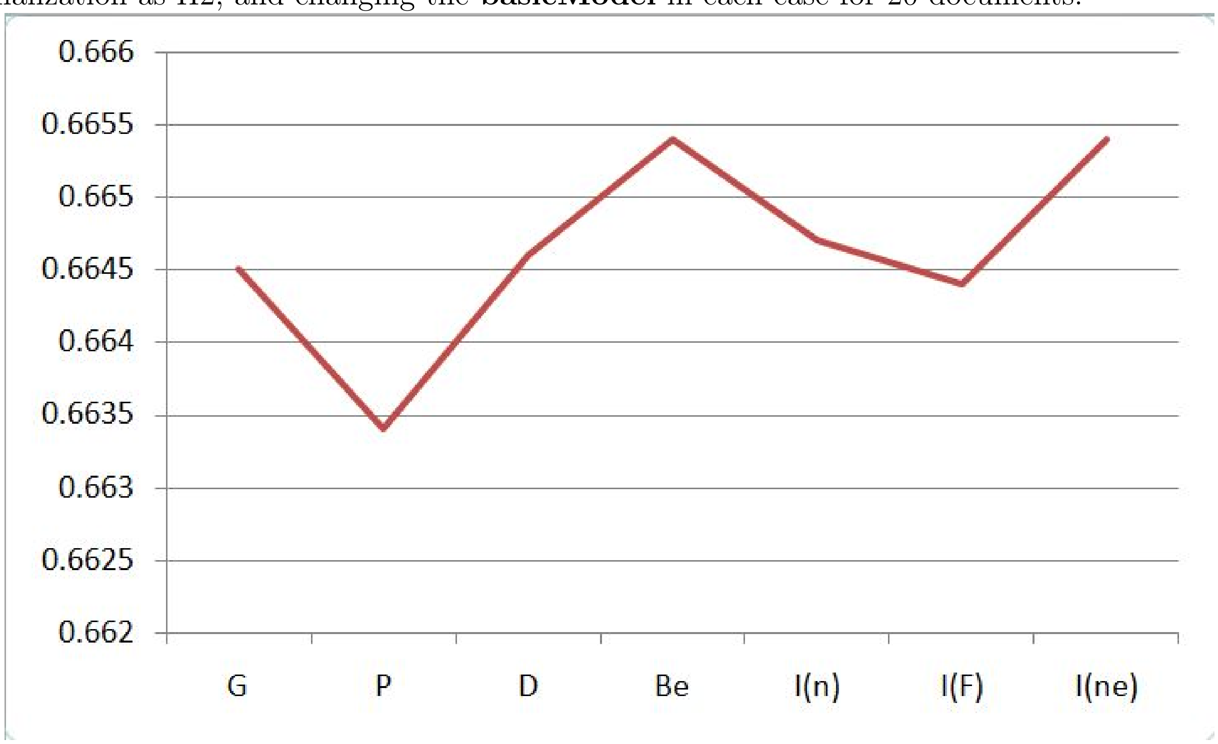
The default DFR model had the following settings:

```
<similarity class="solr.DFRSimilarityFactory">
  <str name="basicModel">G</str>
  <str name="afterEffect">B</str>
  <str name="normalization">H2</str>
</similarity>
```

The default MAP and nDCG score obtained were **0.6645 and 0.8273** as shown below:

map	001	0.2979	ndcg	001	0.5732
map	002	0.4173	ndcg	002	0.6136
map	003	0.5610	ndcg	003	0.8683
map	004	0.5804	ndcg	004	0.8601
map	005	0.5000	ndcg	005	0.7244
map	006	0.4474	ndcg	006	0.6972
map	007	0.8333	ndcg	007	0.9639
map	008	1.0000	ndcg	008	1.0000
map	009	1.0000	ndcg	009	0.9931
map	010	1.0000	ndcg	010	1.0000
map	011	1.0000	ndcg	011	1.0000
map	012	0.7211	ndcg	012	0.9158
map	013	0.1041	ndcg	013	0.4312
map	014	0.6386	ndcg	014	0.8278
map	015	0.8667	ndcg	015	0.9407
map	all	0.6645	ndcg	all	0.8273
gm_map	all	0.5840	ndcg		

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.
Below is the graph that shows the change in MAP values keeping AfterEffect as B and Normalization as H2, and changing the **basicModel** in each case for 20 documents:



Below is the table for further changes in basic settings for DFR:

basicModel	afterEffect	normalization	c	Map Values
Be	B	H2	1000	0.6801
Be	B	H2	1.2	0.6649
I(ne)	B	H2	1000	0.6678
Be	L	H2	1000	0.6921

Table 1: Map Values for changes in basicModel, afterEffect, normalization and c

Results for DFR: We found an optimum value keeping **basicModel as Be, Aftereffect as L and 2nd Normalization as H2**. Hence we have chosen the above parameters for our DFR Model. The MAP and nDCG values obtained through trec_eval were **0.6921 and 0.8443**, if 20 documents were retrieved.

3.1.3 Vector Space Model(VSM)

This was the default similarity class in previous versions i.e before Solr-6.0. The VSM similarity class is implemented through the ClassicSimilarityFactory.

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

The default MAP and nDCG score obtained were **0.6483 and 0.8217** as shown below:

```
$ ./trec_eval -q -c -M1000 D:/MS/1stSem/   ndcg      001   0.5802
map        001   0.2817                    ndcg      002   0.5954
map        002   0.3923                    ndcg      003   0.8620
map        003   0.5729                    ndcg      004   0.8590
map        004   0.5724                    ndcg      005   0.7244
map        005   0.5000                    ndcg      006   0.7321
map        006   0.5257                    ndcg      007   0.9639
map        007   0.8333                    ndcg      008   1.0000
map        008   1.0000                    ndcg      009   0.9816
map        009   1.0000                    ndcg      010   1.0000
map        010   1.0000                    ndcg      011   1.0000
map        011   1.0000                    ndcg      012   0.8035
map        012   0.4615                    ndcg      013   0.4346
map        013   0.1098                    ndcg      014   0.8916
map        014   0.7028                    ndcg      015   0.8979
map        015   0.7721                    ndcg      all   0.8217
map        all   0.6483
gm_map     all   0.5701
```

There are no parameters which can be configured for VSM, hence there is no tuning required.

Results for VSM: The MAP and nDCG values obtained through trec_eval were **0.6483 and 0.8217**, if 20 documents were retrieved.

3.2 Creating a common text_all field and Boosting

We created a common field named **text_all** to store all text_en, text_ru and text_de data as phrases,not as tokens.

As the relevance depends on how similar the search result is based on a query, the creation of a field which can store all the text from various other language specific text fields can help in increasing the relevancy. Boosting the score for that newly created field, using the qf and pf parameters in the query syntax helped in increasing MAP values to some extent.

```
<field name="text_all" type="text_all" indexed="true" stored="true" multiValued="true"/>
<copyField source="text_de" dest="text_all"/>
<copyField source="text_en" dest="text_all"/>
<copyField source="text_ru" dest="text_all"/>
```

3.3 Using various filters and tokenizers

For text_en, text_de and text_ru fields, we have used **UAX29URLEmailTokenizerFactory** as this tokenizer splits the text field into tokens, treating whitespace and punctuation as delimiters.
KeywordTokenizerFactory is used for **text_all** field as it does not break the text into tokens, and stores them as phrases. This helped us match the queries to a better extent. **KeywordTokenizerFactory** didnot go well with DFR and VSM and hence for those **StandardTokenizerFactory** is being used.

Other filters which helped in increasing MAP values are :

- LowerCaseFilterFactory
- StopFilterFactory
- PorterStemFilterFactory
- RemoveDuplicateTokens

The above filters are applied at both indexing and query time.

Results for BM25: Using **KeywordTokenizerFactory** the map values for BM25 **increased** from **0.6714 to 0.6756**.

3.4 Using query parsers and query expansion techniques

The standard query parser searches mainly in the default fields specified in the config, and returns the results, depending on the cumulative score from all the fields for a given document. However, the DisMax query parser is designed to process simple phrases entered by users and to search for individual terms across several fields using different weighting (boosts) based on the significance of each field. The Extended DisMax (eDisMax) query parser is an improved version of the DisMax query parser.

We have used the DisMax Query Parser using **defType=dismax** in the query. As hashtags contains most of the relevant terms we had given more weightage to **tweet_hashtags and tweet_urls**. **text_all** is boosted as it contains phrases and is much better than having tokenized fields.

Following clause is added in the url:
$qf = text\_all^{2} + tweet\_hashtags^{1.0} + tweet\_urls^{1.0} + text\_en^{2} + text\_de^{1.2} + text\_ru^{1.2} \& pf =$

text_all[2]

Results for BM25: The MAP value found was **0.6781**, if 20 documents were retrieved.
Results for DFR: The MAP value found was **0.6801**, if 20 documents were retrieved.
Results for VSM: The MAP value found was **0.6684**, if 20 documents were retrieved.

3.5 Multilingual Support

For multilingual support, we had added **ICUTokenizerFactory**, as it processes multilingual text and tokenizes it appropriately based on its script attribute.

```
<fieldType name="text_all" class="solr.TextField" positionIncrementGap="100" >
<analyzer type="index">
   <tokenizer class="solr.ICUTokenizerFactory"/>
     <filter class="solr.LowerCaseFilterFactory"/>
```

3.6 Translation of queries and Synonyms

After implementing all the above techniques we weren't getting satisfactory results. Also, there were many relevant tweets in languages other than the query language. There were many texts which were not getting captured in solr. For instance,
Refugee → refugee,flüchtling,Flüchtlinge,Flüchtling
General → general,Generäle,officers

Querying in all the three languages could have helped in fetching more relevant tweets.In order to implement the same, we tried to add synonyms for the various tokens which were used in the queries as a part of multi-language query expansion.For example, if someone queries Putin, then Solr would also search for путин, which is the Russian translation for Putin. Simliarly, there were many words which were translated into English, Russian and German, and added as synonyms in the **synonyms.txt** file.

```
million,Mio
Airbnb,Instacart,Kickstarter,Tech Companies,Tech Firm
asyl,убежище,asylum
civil war,Bürgerkrieg,гражданская война
Flüchtlingshilfe,refugee relief
US,U.S,USA,America,Америка,Amerika,США
Syrien,Syria,Сирия,SYRIA
ISIS,terrorist
russia,Russia,Russia's,Russian,Россия,Russische,
interview,Interview,Vorstellungsgespräch,vorstellungsgespräch,Bewerbungsgespräch,bewerbungsgespräch,интервью
challenges,проблемы,Herausforderungen,herausforderungen,Herausforderung,herausforderung
обаму,obama
бьет,beats,beat
путин,putin,Putin
беженцев,refugees,Refugees,Flüchtlinge
убит,Kills,kill,killed,Killed,ISIS,funeral
полицией,police,Police
Problematik,problem
begrüßt,hailed,welcomed
germany,Germany,Deutschland,Германия
krise,crisis,кризис
bombed,bombardiert,fegen,bombardieren,разбомбленный
terrorist,terroristisch,terrorist,террорист
air force,luftwaffe,luftstreitkraft,luft kraft,воздушные силы
```

Implementing query expansion using synonyms gave us a significant boost in relevance for all of the different models by around **10%**.

Results for BM25:
The MAP and nDCG value found was **0.7601 and 0.8805**, if 20 documents were retrieved.
The MAP and nDCG value found was **0.8384 and 0.9400**, if 1000 documents were retrieved.

Results for DFR:
The MAP and nDCG value found was **0.7507 and 0.8772**, if 20 documents were retrieved.
The MAP and nDCG value found was **0.7971 and 0.9371**, if 1000 documents were retrieved.

Results for VSM:
The MAP and nDCG value found was **0.7507 and 0.8772**,if 20 documents were retrieved.
The MAP and nDCG value found was **0.7971 and 0.9371**,if 1000 documents were retrieved.

4 Conclusion

We have successfully implemented the three IR models and we tried to improve the performance of all the three models by using various techniques described above.
The comparison between default and final model of **BM25 for 20 retrieved documents**:

map	001	0.2854
map	002	0.4202
map	003	0.5729
map	004	0.5724
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	1.0000
map	012	0.6616
map	013	0.1041
map	014	0.6386
map	015	0.8667
map	all	0.6636
gm_map	all	0.5838

map	001	0.3195
map	002	0.6003
map	003	0.5729
map	004	0.6957
map	005	0.6875
map	006	1.0000
map	007	1.0000
map	008	1.0000
map	009	1.0000
map	010	1.0000
map	011	1.0000
map	012	0.7352
map	013	0.2857
map	014	0.6386
map	015	0.8667
map	all	0.7601
gm_map	all	0.7119

The comparison between default and final model of **DFR for 20 retrieved documents**:

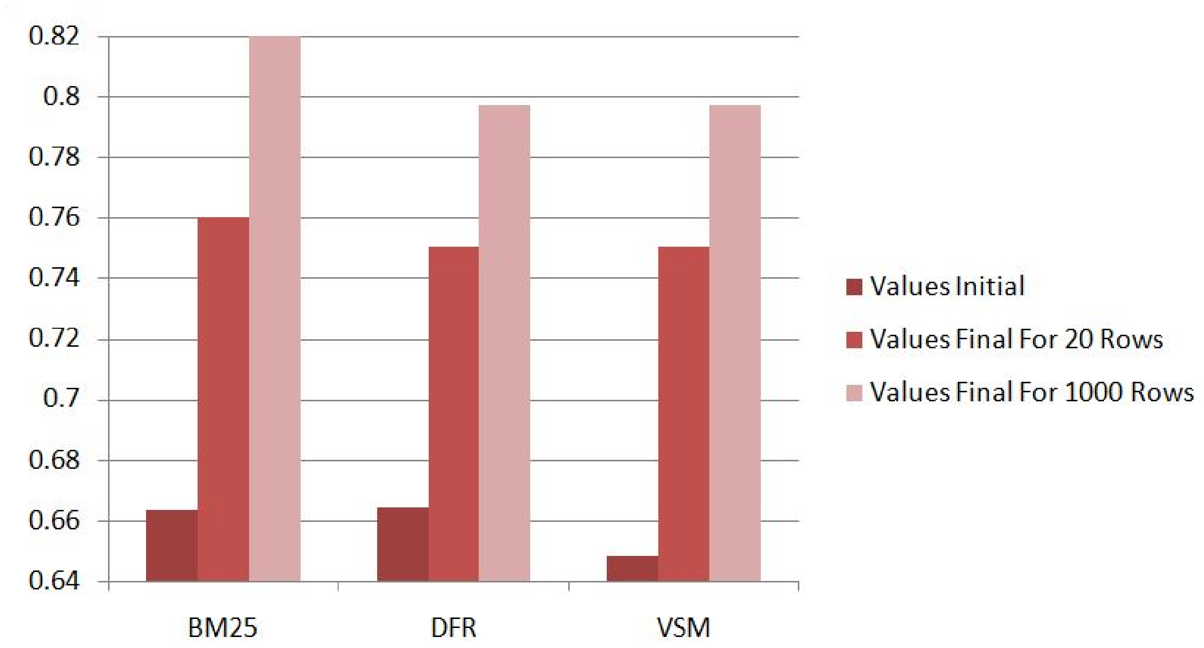
map	001	0.2979	map	001	0.2165
map	002	0.4173	map	002	0.5999
map	003	0.5610	map	003	0.7162
map	004	0.5804	map	004	0.7346
map	005	0.5000	map	005	0.6500
map	006	0.4474	map	006	0.9889
map	007	0.8333	map	007	1.0000
map	008	1.0000	map	008	1.0000
map	009	1.0000	map	009	1.0000
map	010	1.0000	map	010	1.0000
map	011	1.0000	map	011	1.0000
map	012	0.7211	map	012	0.4423
map	013	0.1041	map	013	0.2244
map	014	0.6386	map	014	0.7663
map	015	0.8667	map	015	0.9216
map	all	0.6645	map	all	0.7507
gm_map	all	0.5840	gm_map	all	0.6801

The comparison between default and final model of **VSM for 20 retrieved documents**:

```
$ ./trec_eval -q -c -M1000 D:/MS/1stSem/
```

map	001	0.2817	map	001	0.2165
map	002	0.3923	map	002	0.5999
map	003	0.5729	map	003	0.7162
map	004	0.5724	map	004	0.7346
map	005	0.5000	map	005	0.6500
map	006	0.5257	map	006	0.9889
map	007	0.8333	map	007	1.0000
map	008	1.0000	map	008	1.0000
map	009	1.0000	map	009	1.0000
map	010	1.0000	map	010	1.0000
map	011	1.0000	map	011	1.0000
map	012	0.4615	map	012	0.4423
map	013	0.1098	map	013	0.2244
map	014	0.7028	map	014	0.7663
map	015	0.7721	map	015	0.9216
map	all	0.6483	map	all	0.7507
gm_map	all	0.5701	gm_map	all	0.6801

Below is a graph depicted to show the initial and final values:



For generalized system, BM25 model proved to be most efficient model after all the efforts made for improvement.The techiques used above provided some improvement to the system, but there is a lot more scope of improvement to do in future prospects.

References

[1] http://ir.dcs.gla.ac.uk/wiki/DivergenceFromRandomness.

[2] https://lucene.apache.org/solr/guide/6_6/the-standard-query-parser.html

[3] https://lucene.apache.org/solr/guide/6_6/the-dismax-query-parser.html

[4] https://lucene.apache.org/solr/guide/6_6/tokenizers.htmlTokenizers-UAX29URLEmailTokenizer

[5] https://lucene.apache.org/solr/guide/6_6/tokenizers.html