Z534 SEARCH

ASSIGNMENT 2
Retrieval

Introduction

The assignment aims to use different retrieval models to retrieve search results based on query docs and analyse these results using some evaluation metrics such as Precision, Recall, NDCG, MAP and MRR.

Task 1: Implement your first search algorithm

For this task a ranking function is implemented based on TF-IDF. For each query we parse the same using Standard Analyser (which is used for creating index), and calculate the relevance score of each query. The ranking function is as follows :

$$F(q, doc) = \sum_{t \in q} \frac{c(t,doc)}{length(doc)} \cdot log(1 + \frac{N}{k(t)})$$

where q is user query, doc is the target (candidate document in AP89), t is the query term, c(t,doc) is the count of term t in document doc, N is the total number of documents in AP89, and kt is the total number of documents that have the term t.

Task 2: Test your search function with TREC topics

For this task we test our search algorithm on <title> and <desc> tags with TREC standardized topic collections and output the top 1000 search results to the text files mySearchAlgoShortQuery.txt and mySearchAlgoLongQuery.txt respectively for the two tags. The output files have the following format :

```
query-id Q0 document-id rank score Run-id
```

Task 3: Test Other Search Algorithms

For this task we test the following retrieval and ranking algorithms by using Lucene API. The following algorithms are tested and respective text file with top 1000 search results are generated for each model for short and long queries.

- **Vector Space Model** (org.apache.lucene.search.similarities.DefaultSimilarity)

In this documents and queries are viewed as vectors and the cosine similarity score between them is calculated. In this terms are weighted by importance. However the disadvantage is that it assumes terms are independent.

- **BM25** (org.apache.lucene.search.similarities.BM25Similarity)

It stands for Best Match 25. It is also based on a TF-IDF approach. Lucene's BM25 works relatively well by tweaking the IDF part to avoid negative scores.

- **Language Model with Dirichlet Smoothing** (org.apache.lucene.search.similarities.LMDirichletSimilarity)

This model uses a probabilistic approach and models documents using a multinomial distribution with Dirichlet prior. It works better than Jelinek Mercer as it applies smoothing based on sample size.

- **Language Model with Jelinek Mercer Smoothing** (org.apache.lucene.search.similarities.LMJelinekMercerSimilarity, set λ to 0.7)

This model is based on the probabilistic approach using maximum likelihood model along with the collection model. It uses a coefficient λ to control the influence of each model and optimize retrieval performance. This works better only for longer documents.

Task 4: Algorithm Evaluation

For this task, the results of the above search algorithms are evaluated using TRECEVAL. The following evaluation metrics are used:

TRECEVAL Result	What it returns?
P_5	Precision of the 5 first documents
P_10	Precision of the 10 first documents
P_20	Precision of the 20 first documents
P_100	Precision of the 100 first documents
recall_5	Recall of the 5 first documents
recall_10	Recall of the 10 first documents
recall_20	Recall of the 20 first documents
recall_100	Recall of the 100 first documents
map	Mean average precision
recip_rank	MRR - Reciprocal Rank
ndcg_cut_5	Normalized Discounted Cumulative Gain for 5 documents
ndcg_cut_10	Normalized Discounted Cumulative Gain for 10 documents
ndcg_cut_20	Normalized Discounted Cumulative Gain for 20 documents
ndcg_cut_100	Normalized Discounted Cumulative Gain for 100 documents

- ***Precision***

Precision is given as the ratio of true positives (TP) and the total number of predicted positives. Precision is therefore a ratio of the number of relevant items retrieved to the total number of items retrieved.

$$Precision = \frac{TP}{TP+FP}$$

- ***Recall***

Recall is given as the ratio of TP and total of ground truth positives. Recall is the ratio of the number of relevant items retrieved to the number of relevant items in the corpus.

$$Recall = \frac{TP}{TP+FN}$$

- ***MAP -Mean Average Precision***

This is the mean of the average precision scores for each query

- ***Reciprocal Rank***

Reciprocal Rank calculates the reciprocal of the rank at which the first relevant document was retrieved. This is averaged across queries, to measure the Mean Reciprocal Rank (MRR).

- ***NDCG***

NDCG or Normalized Discounted Cumulative gain considers the position of the document in the result set to measure relevance or gain. A lower position means less useful. The gain is accumulated at the top and discounted for lower ranked documents.

Results

- ***Short Queries***

Evaluation Metric	My algorithm	Vector Space Model	BM25	Language model with Dirichlet Smoothing	Language Model with Jelinek Mercer Smoothing
P@5	0.148	0.296	0.3080	0.3480	0.2880
P@10	0.146	0.302	0.3000	0.3260	0.2820
P@20	0.135	0.26	0.2700	0.2890	0.2440
P@100	0.0976	0.1638	0.1678	0.1692	0.1606
Recall@5	0.0234	0.0539	0.0488	0.0620	0.0534
Recall@10	0.0539	0.096	0.0879	0.0995	0.0913
Recall@20	0.0839	0.1416	0.1376	0.1467	0.1304
Recall@100	0.2191	0.3567	0.3572	0.3470	0.3324
MAP	0.1051	0.1969	0.2004	0.2059	0.1935
MRR	0.2862	0.4696	0.4772	0.4785	0.4637
NDCG@5	0.1627	0.3116	0.3246	0.3517	0.3063
NDCG@10	0.1621	0.3183	0.3199	0.3420	0.3026
NDCG@20	0.1619	0.3043	0.3117	0.3325	0.2897
NDCG@100	0.1916	0.3205	0.3259	0.3313	0.3125

- ***Long Queries***

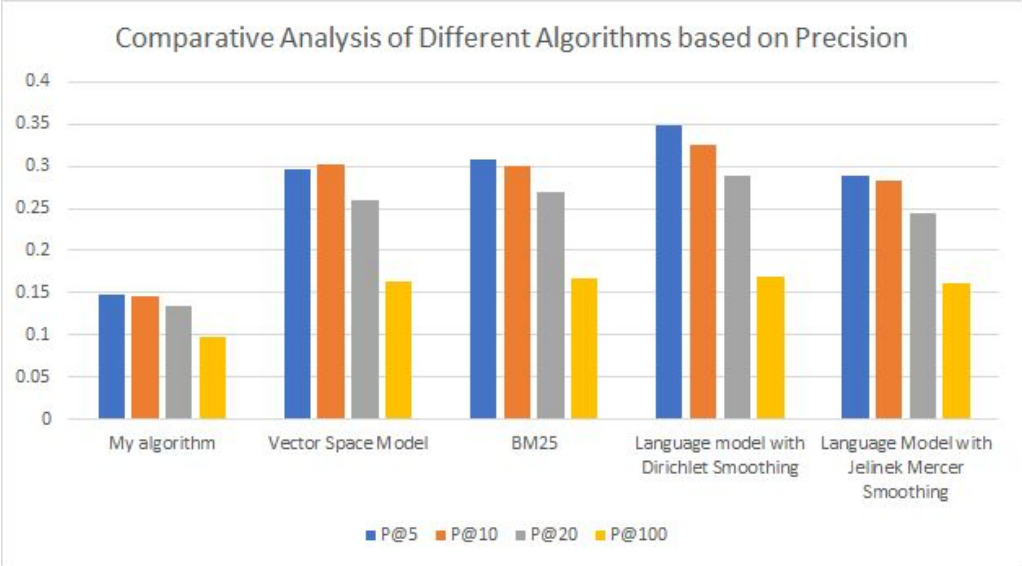
Evaluation Metric	My algorithm	Vector Space Model	BM25	Language model with Dirichlet Smoothing	Language Model with Jelinek Mercer Smoothing
P@5	0.128	0.264	0.284	0.256	0.236
P@10	0.124	0.248	0.246	0.242	0.214
P@20	0.114	0.221	0.234	0.234	0.213
P@100	0.0742	0.141	0.1488	0.146	0.1382
Recall@5	0.0187	0.0361	0.0402	0.0403	0.0413
Recall@10	0.0368	0.0634	0.071	0.0708	0.0658
Recall@20	0.0595	0.1058	0.1159	0.127	0.1142
Recall@100	0.1743	0.2942	0.3167	0.334	0.2914
MAP	0.0664	0.1538	0.1676	0.1588	0.1518
MRR	0.2563	0.449	0.4497	0.3483	0.3666
NDCG@5	0.1388	0.2865	0.3004	0.2499	0.2385
NDCG@10	0.1341	0.2713	0.2726	0.2475	0.2304
NDCG@20	0.131	0.2589	0.2709	0.2593	0.2506
NDCG@100	0.1466	0.2706	0.2866	0.2756	0.2621

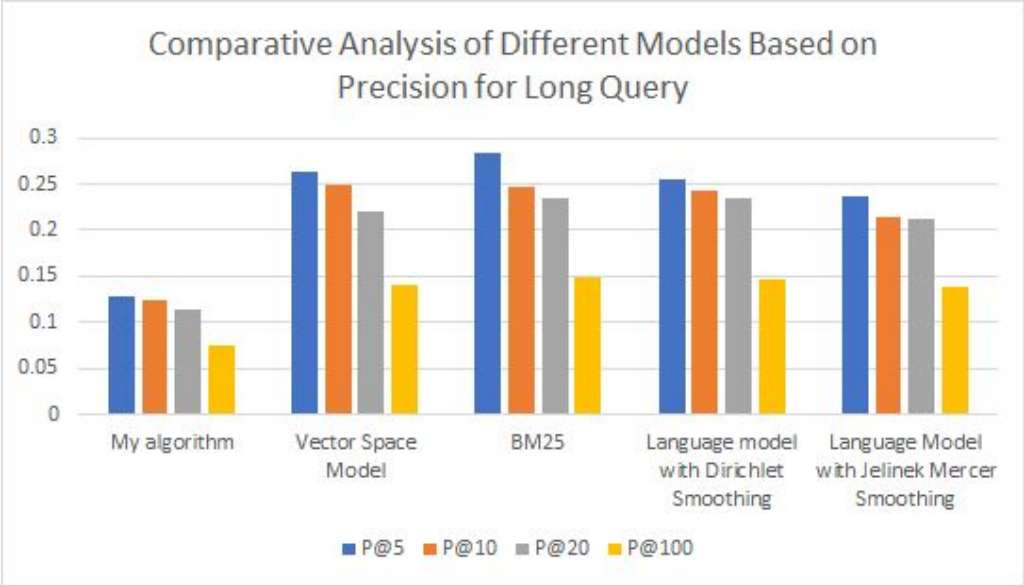
Summary of findings:

- **Model Comparison:**
    - **Based on Precision:**

For the 5 search models, the language model with Dirichlet Smoothing gives the best precision values for short queries. For long queries, BM25 performs slightly better than Dirichlet Smoothing for fewer documents and is almost comparable with Dirichlet when we consider more documents. Also, the precision is best for fewer documents for long and

short queries. This is obvious as more items are relevant from all retrieved documents for 5 documents than for 1000 documents. Our search algorithm based on TF-IDF performs the worst.

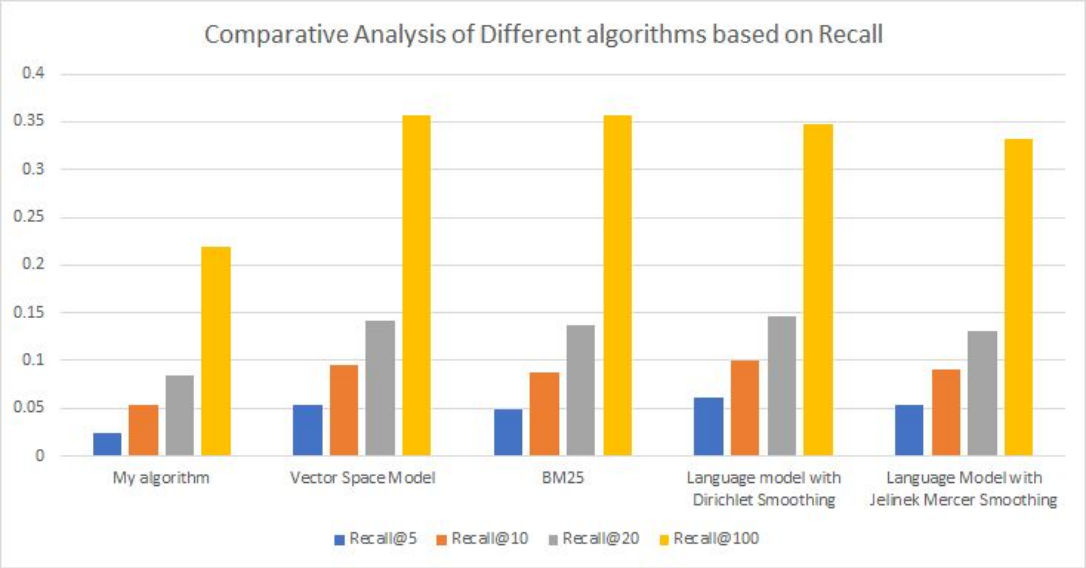


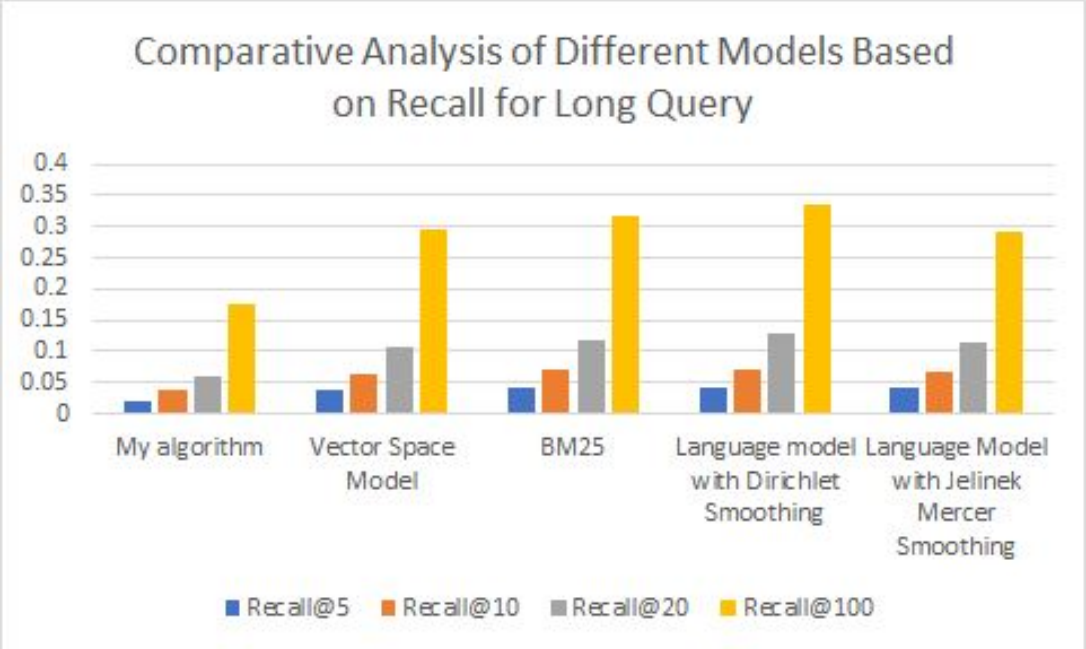


- **Based on Recall:**

For the 5 search models,the recall values for Dirichlet Smoothing, BM25 and Vector Space Model are almost comparable for short queries and Dirichlet is slightly better for long query. Converse to precision, the recall is best for more documents We can think of this as

for more documents, more relevant items are retrieved from all relevant items in the corpus. Our search algorithm based on TF-IDF performs the worst.

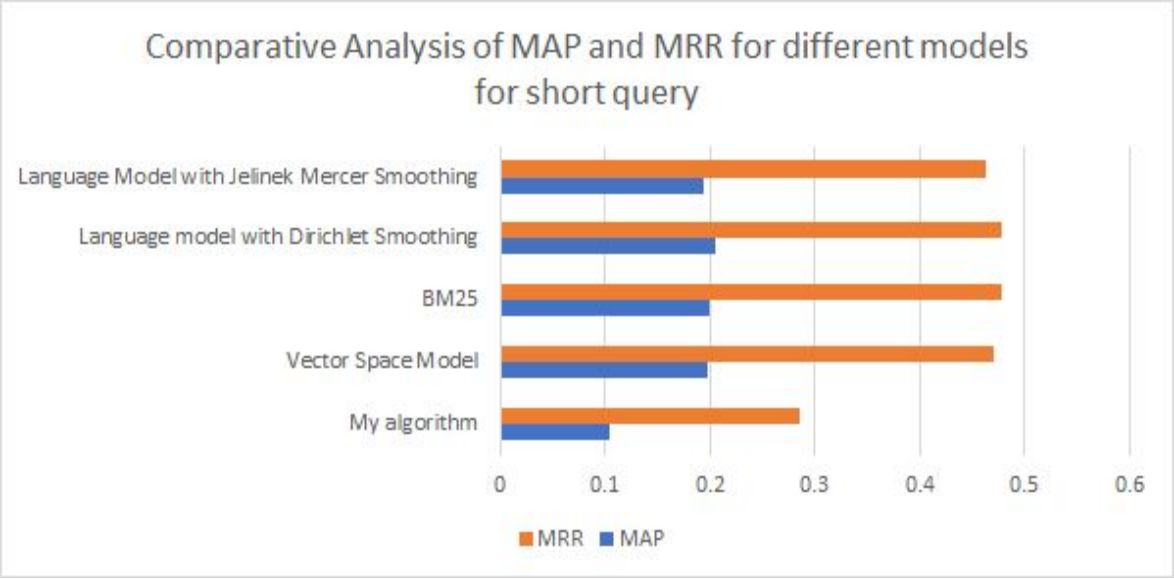


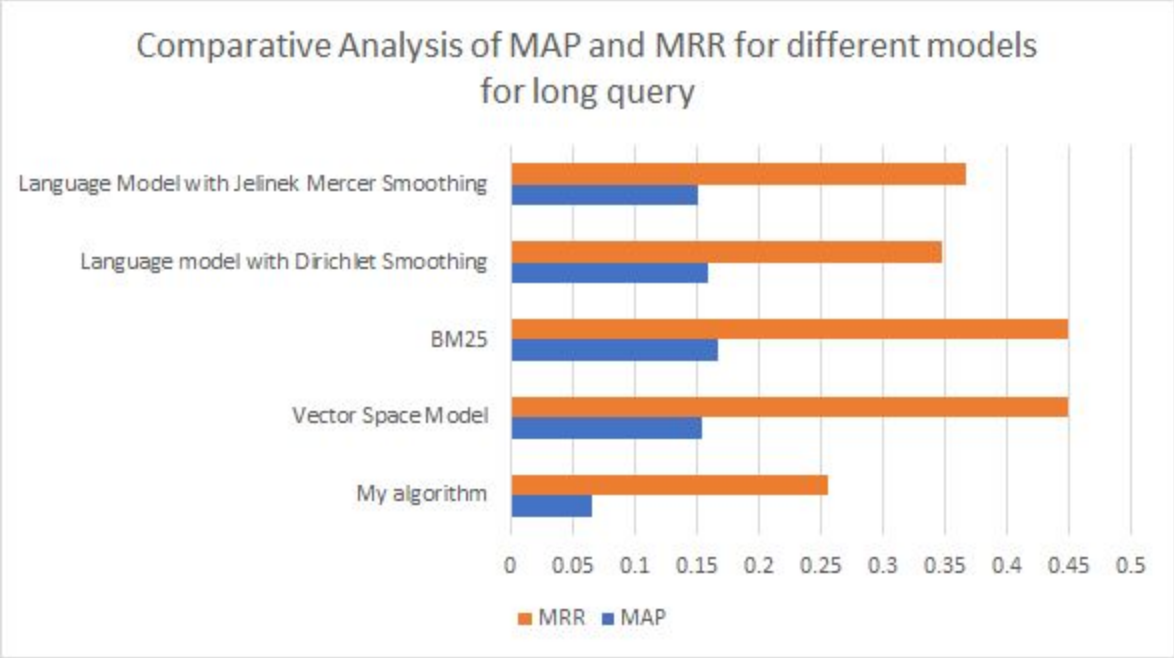


- **MAP and MRR:**

Model with Dirichlet Smoothing and BM25 gives a slightly better mean average precision indicating that on an average this model performs better than other for short queries. For long queries BM25 and Vector Space Model produce best results. Mean Reciprocal values

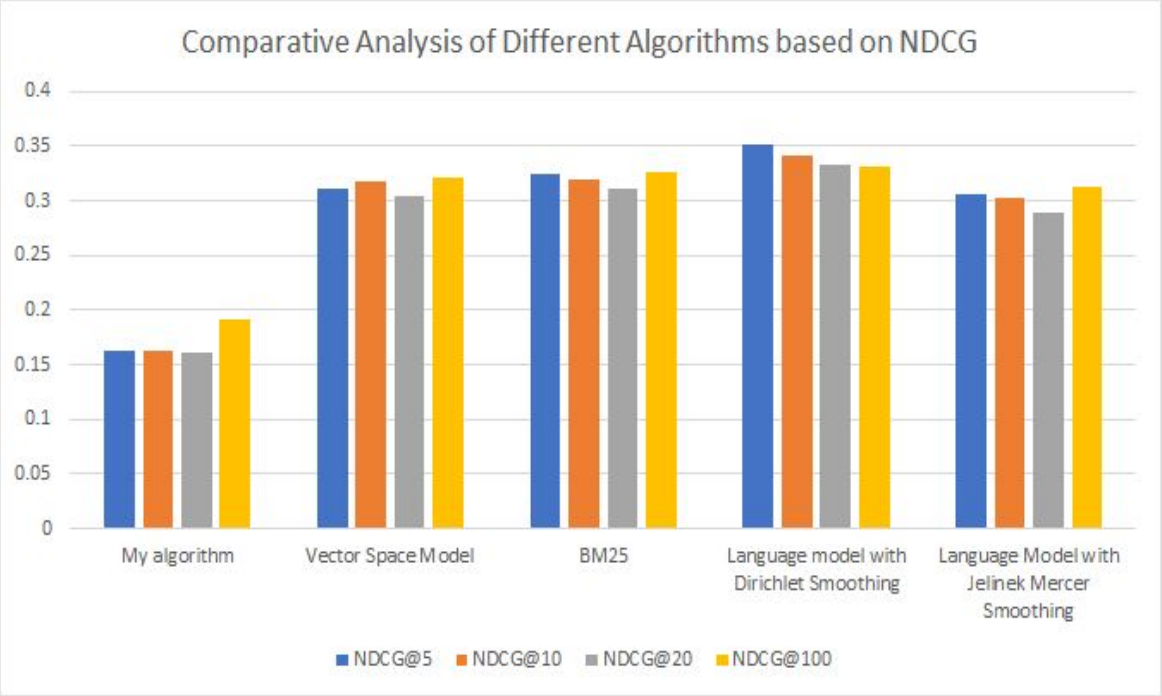
display a similar trend. The TF-IDF based retrieval model performs the worst. For TF-IDF based model 1, MAP suggests that precision of search results is low and MRR suggests that the rank at which the best document is retrieved is low compared to other models.

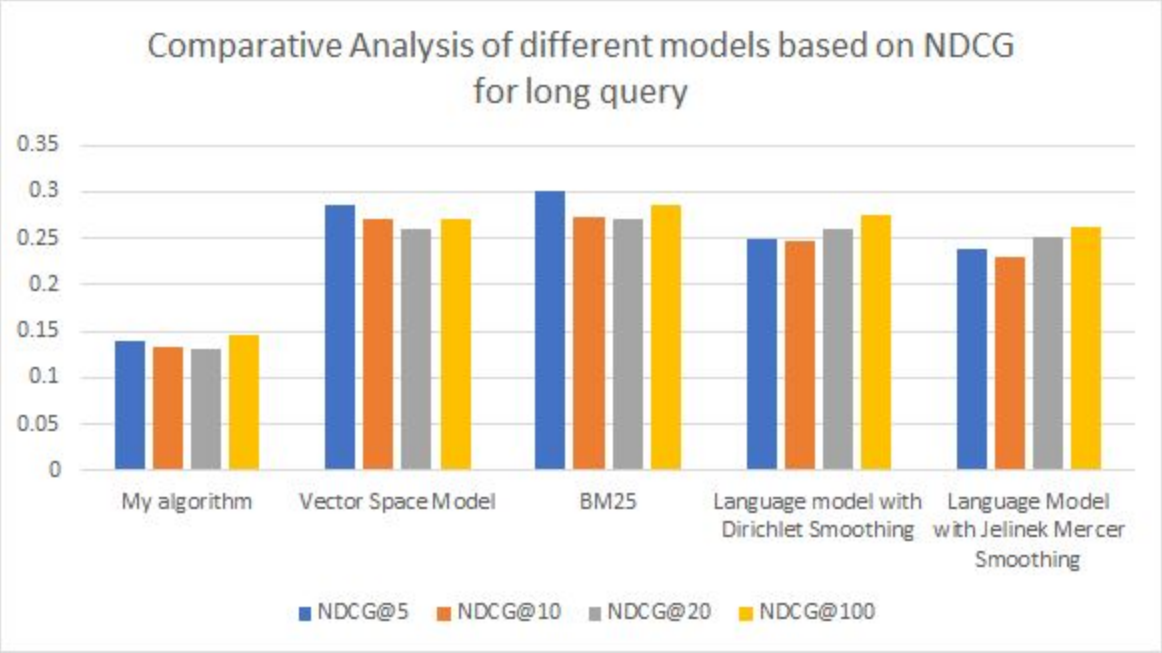




- **NDCG:**

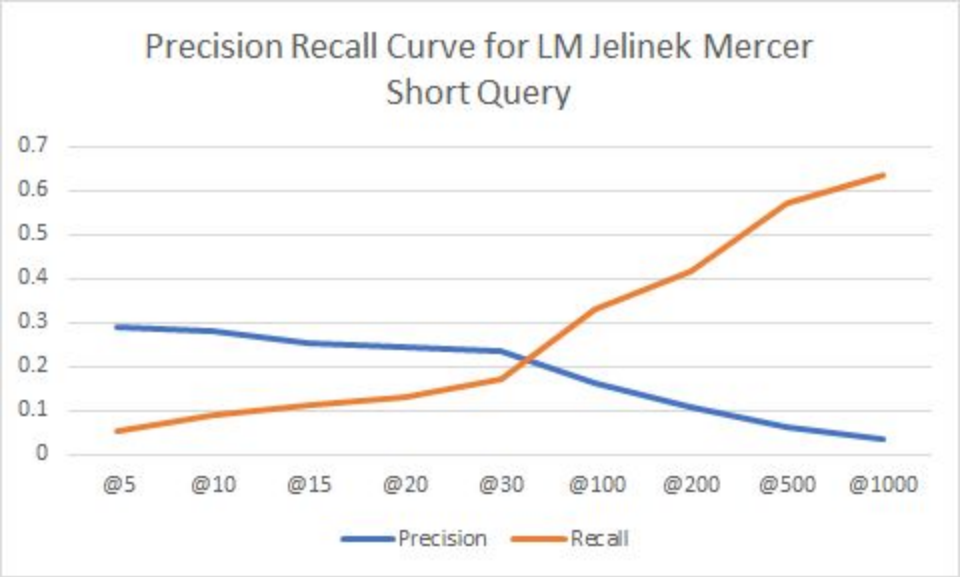
For short queries, Dirichlet smoothing model and for long queries BM25 gives best NDCG sores. The scores are almost comparable for 5, 10, 20 and 100 documents.

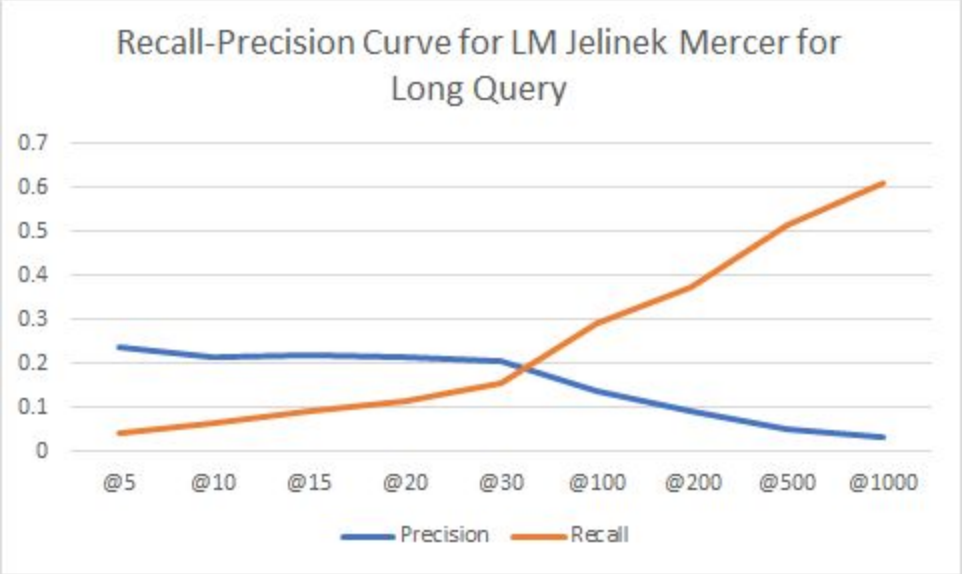


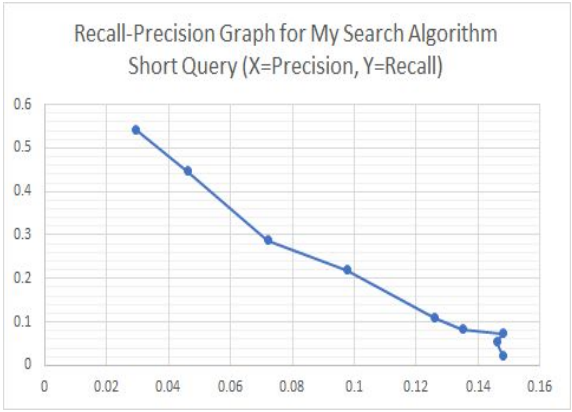


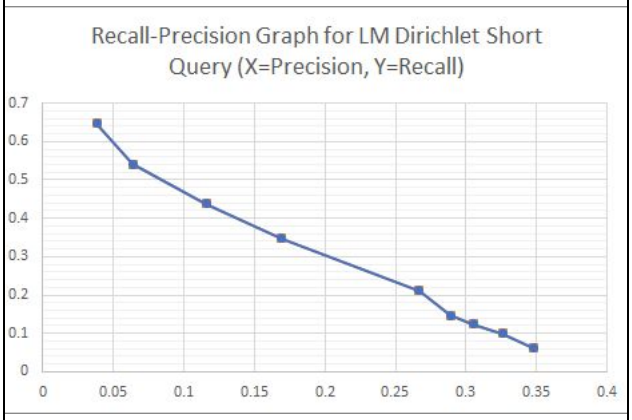
- NDCG looks like the better evaluation metric compared to precision and recall whose scores vary significantly with different number of documents retrieved.
- **Recall-Precision Tradeoff:**

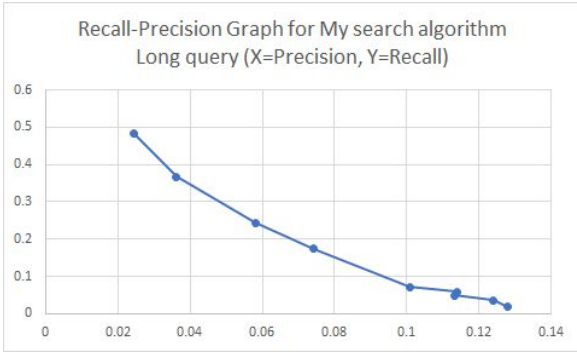
The values of precision and recall suggest that recall increases when more documents are considered, while precision reduces. It is therefore not possible to maximize both precision and recall metrics at the same time. This looks intuitive. If we have to recall everything, we will have to keep generating results which are not accurate, hence lowering the precision.

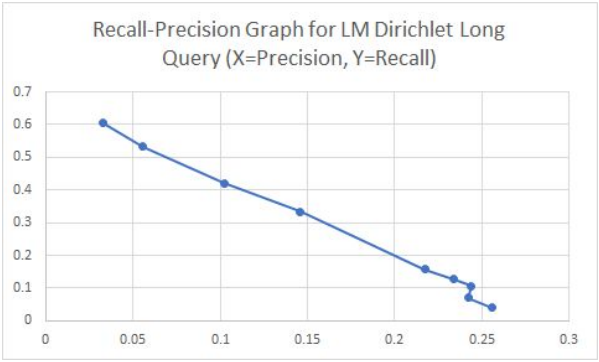





Recall-Precision Graph for My Search Algorithm
Short Query (X=Precision, Y=Recall)
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16


Recall-Precision Graph for LM Dirichlet Short
Query (X=Precision, Y=Recall)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4


Recall-Precision Graph for My search algorithm
Long query (X=Precision, Y=Recall)
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14


Recall-Precision Graph for LM Dirichlet Long
Query (X=Precision, Y=Recall)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.05
0.1
0.15
0.2
0.25
0.3