Nicholas Morgan, Aswath Ilangovan, Michael Hanna

CS 6200: Information Retrieval

Fall 2018

Professor Nada Naji

I. Introduction

Our information retrieval project, a text retrieval system with evaluation, includes two distinct parts: Lucene and Non-Lucene. Non-Lucene utilizes a Tokenizer/Indexer with the capability to run BM25, TF-IDF, or JM-Smoothing for retrieval. Query-time stopping, index and retrieval of a stemmed corpus/queries, snippet generation, and query enrichment have also been implemented. Our evaluation generates data for precision, recall, mean reciprocal rank, and mean average precision for all data stored in the Outputs directory.

Task	**Contributors**
Lucene, Evaluation, Query Enrichment	Aswath Ilangovan
BM25, Snippets	Michael Hanna
TFIDF, JM-Smoothing, Stopping/Stemming, Query Enrichment, Evaluation	Nicholas Morgan

II. Literature and Resources

Retrieval Method Parameters:

BM25 utilized parameters found in the TREC-6 conference and the CMS textbook [1,2]. TFIDF was performed with heuristic modification found in the CMS textbook and the TREC-3 conference methodology [1, 3].

Query Enrichment:

For query enrichment, pseudo-relevance feedback was used via Rocchio's algorithm and TFIDF for weighting. This was done with an adaptation of methodology found in the TREC-3 Conference [3].

Snippet Generation:

Qualities of a good snippet, such as length of the snippet, query term frequency in the sentence, and token significance, were based on the cited works [1, 5, 6]. The algorithm of choice coalesced extensions of Luhn's work to rank sentences and weighting query term frequencies [1, 5, 6].

III. Implementation and Discussion

We implemented a Tokenizer and an Indexer to generate an index. The Tokenizer was responsible for parsing the document into tokens by using BeautifulSoup, punctuation removal, and case-folding. Our Indexer transformed these tokens (stored in a doc_name: token list index) into a unigram frequency index and a unigram position index(delta encoded for space)[1].

Retrieval methods, modularized into their own files, are coordinated via the Retrieval.py module. This module combines our Indexer and Tokenizer, and executes with following options:

- Stemming
- Re-indexing
- Retrieval Method
- Stopping
- Snippet Generation

JM-smoothing utilized 0.35 as the lambda parameter per the requirements. BM25 retrieval used 1.2 for k1, 0.75 for b and 500 for k2. These parameters were set per standards found in Croft et al. and TREC-6 [1,2]. The k2 value was adjusted amongst the range 0 - 1000, but it was found that moving above or below 500 did not impact the results [1]. The smoothing parameter for TFIDF was tuned to 0.01 due to small corpus term frequencies [1].

Doc_id and query text were parsed from the query file with BeautifulSoup using the <DOC> tag. Then we provided same tokenization as the main corpus, with punctuation removal and case folding.

For query enrichment, we used the Rocchio algorithm with pseudo-relevance feedback. The version used was found in CMS:

$$P' = \alpha . P + \beta . \frac{1}{|Rel|} \sum_{D_i \in Rel} D_i - \gamma . \frac{1}{|Nonrel|} \sum_{D_i \in Nonrel} D_i$$

[1, 3].

For weighting the terms, the methodology of TFIDF from the third TREC conference was chosen[3]. This was adapted into simplified TF*IDF scoring, which resulted in better output [3]. As found in [4], TFIDF performs just as well as a probabilistic weighted version when the result set is limited to around 20 documents. BM25 was the retrieval method used, first to generate the top 20 initial results, and later to rerun the query with the added term. The parameters for Rocchios were retrieved from TREC-3 [3]. These ended as the same parameters as the CMS text [1] where alpha, beta, gamma were 8, 16, 4, respectively.

For stopping and stemming, the same index and retrieval methods were applied. Stemming was moved to its own file due to the index and queries being retrieved from different files than the standard corpus. Stopping was performed only on the queries, due to indexing the entire token set being preferred per [1].

For snippet generation, longer sentences and sentences with query terms were weighted highly. In [5] it was found that with a 100 character snippet length, clickthrough jumped significantly compared to snippets with less than 25 characters [5]. Query terms were more heavily weighted as the average document length was approximately 70-80 tokens. The algorithm used to rank sentences based on their token significance is:

$$f_{d,t} \geq \begin{cases} 7 - 0.1 \times (25 - s_d), & \text{if } s_d < 25 \\ 7, & \text{if } 25 \leq s_d \leq 40 \\ 7 + 0.1 \times (s_d - 40), & \text{otherwise,} \end{cases}$$

[1].

This was weighted with query frequencies, and the top five sentences were reported with query terms highlighted (to the same degree they were present in the original query) [1, 6]. The sentence rank was calculated by giving each sentence a point for each significant token, and then an additional three points for each query tokens present.

Query By Query Analysis

Baseline:

	Top 5 Retrieved Docs Delta		
Query No	BM25 - TFIDF	BM25 - JM Smoothing	TFIDF - JM Smoothing
6	3	3	1
59	5	5	2
61	3	2	1

Analysis of delta in the top 5 retrieved documents for the baselines runs was performed on Q6, Q59, and Q61. Selection of these queries is discussed in the Evaluation section below. For all queries, TFIDF and JM Smoothing have a higher degree of overlap compared to BM25. TFIDF and JM smoothing have a higher weight on the TF and do not normalize against the avg document length like BM25. With short documents for the index, the lack of penalty against document length could cause the improved performance of BM25 and similarity observed between TFIDF and JM Smoothing.

Stemmed:

Stemmed		Top 5 Retrieved Doc Delta		
Query No	Total Terms	BM25 - TFIDF	BM25 - JM Smoothing	TFIDF - JM Smoothing
1	3	1	1	1
4	5	5	2	4
6	7	4	1	3

Three queries were chosen from the stemmed queries for analysis due to their variances in length. Q1 was shortest with three terms. These three terms did not appear to be commonly stopped query terms, and the results show there were very little differences in the makeup of the top five documents. With Q4 and Q6, there were several extra terms. For example, some of these were common stopwords like “and” and “in”. These caused divergence to occur between TFIDF and the other two retrieval methods. The stopped words could have caused a bigger shift in the weighting. In addition, more terms may have been linked to the stemmed terms, causing differences in the overall rankings. Even with stemmed tokens, the rankings varied between the retrieval methods.

IV. Results

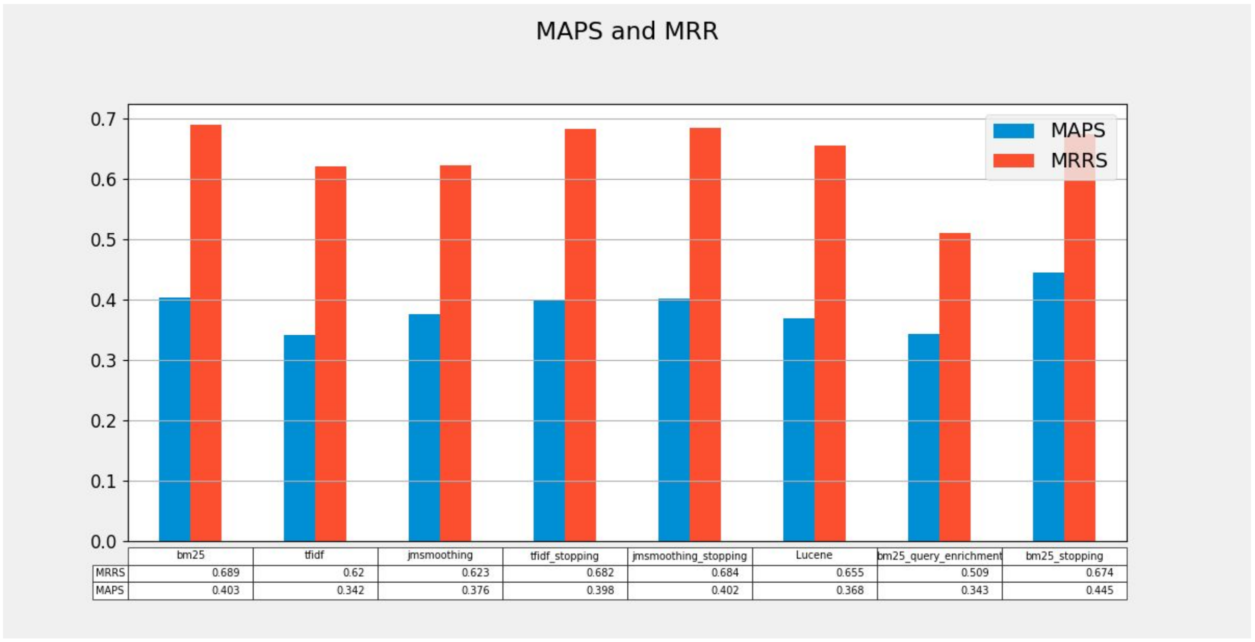
The following table outlines the results and their locations.

Description	**Location**	**Comments**
Unigram Model	unigram_stem_frequency.txt unigram_positional.txt unigram_frequency.txt	Stemmed inverted index Positional inverted index Unigram inverted index
Document Term Counts	Doc_Term_Counts_Stem.txt Doc_Term_Counts.txt	Document statistics
Query Output	Outputs/*	File naming format: **Model_Query#_special.txt** *BM25_1.txt or BM25_1_stemmed.txt*
Query Level Precision/Recall	figures/precision_recall/*.csv	File naming format: **Model_evaluation.csv** *BM25_recalls.csv* *BM25_precisions.csv*
Precision@K	figures/p_at_k5_k20.csv figures/p_at_k.csv figures/Precision@k.png	Precisions@k=5 and 20 Precisions@k(1-100) Figure of table of K=5, 20
Precision, Query Level	figures/p_per_query.csv figures/p_per_query.png	MAP table for each model per query Bar plot of data
Recall, Query Level	figures/r_per_query.csv figures/r_per_query.png	Recall table for each model per query Bar plot of data
MAPS&MRR	figures/MAPS_MRR.png	Bar Plot with Table
Precision vs Recall Graph	figures/mean_avgPrecision@Delta_vs_Recall.pn g figures/mean_avgPrecision_vs_Recall.png	Precision@Delta Recall Precision@Every step

V. Conclusion

Analysis of Results

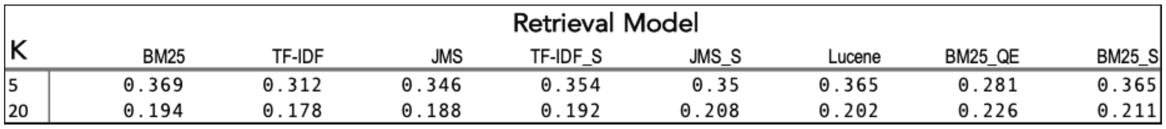
Retrieved documents from models were evaluated against the relevant document set. Mean Average Precision and Mean Reciprocal Rank were calculated for each model. Precisions and recalls for each iteration k were calculated as average precision and recall for all queries in the model at k.



	bm25	tfidf	jmsmoothing	tfidf_stopping	jmsmoothing_stopping	Lucene	bm25_query_enrichment	bm25_stopping
MRRS	0.689	0.62	0.623	0.682	0.684	0.655	0.509	0.674
MAPS	0.403	0.342	0.376	0.398	0.402	0.368	0.343	0.445

Figure 1: Mean Reciprocal Ranks (MRRS) and Mean Average Precisions (MAPS) for each Retrieval Model. MAPS and MRRS are calculated at each step where Recall updates for a query.

Figure 1 shows BM25 is the best baseline model in terms of MRRS and MAPS. TF-IDF performs the worst. When stopping was performed, all MAPS were improved. BM25 was the only model where MRRS was not improved with stopping. TF-IDF saw the greatest improvement and this is likely due removal of high TF stop words that were adding noise especially in the short documents in this corpus. BM25 seems relatively unhindered by this likely due to normalization against the document length. BM25_QE appears to have the worst MRRS. The query expansion technique might be expanding the terms too liberally with our parameters.



	Retrieval Model							
K	BM25	TF-IDF	JMS	TF-IDF_S	JMS_S	Lucene	BM25_QE	BM25_S
5	0.369	0.312	0.346	0.354	0.35	0.365	0.281	0.365
20	0.194	0.178	0.188	0.192	0.208	0.202	0.226	0.211

Table 1: Average Precision for all 64 queries for given models, at iteration K.

Relatively poor performance of TF-IDF can be observed by the average precision at k=5 and 20 shown in *Table 1*. Initial precision for BM25_QE at k=5 is worse compared to other algorithms, especially, since it performs best without QE. However, at k=20 the performance of BM25_QE is better compared to the rest. QE with Rocchio's algorithm results in decreased initial precision, but with sufficient k entries, performance is gained.

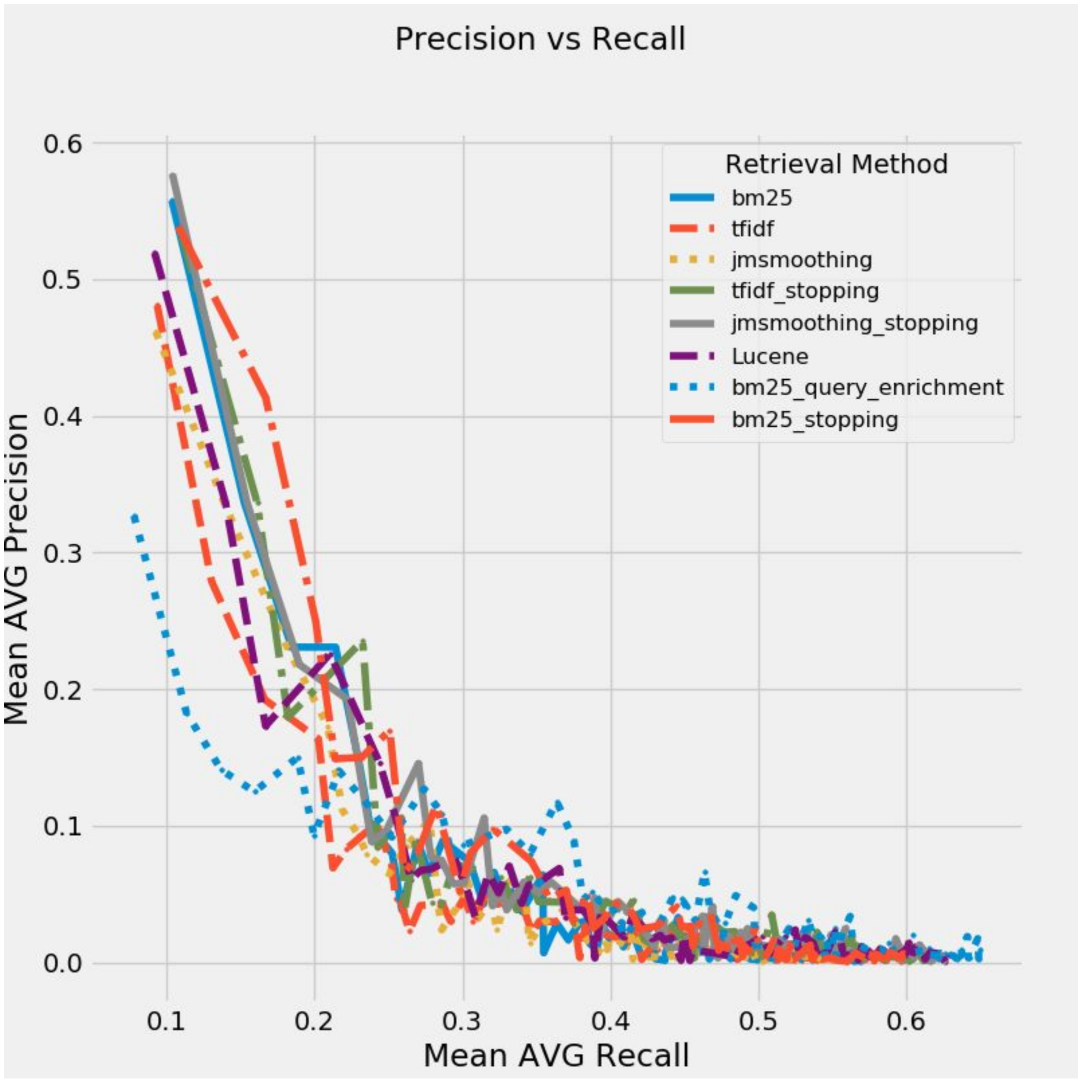


Figure 2A

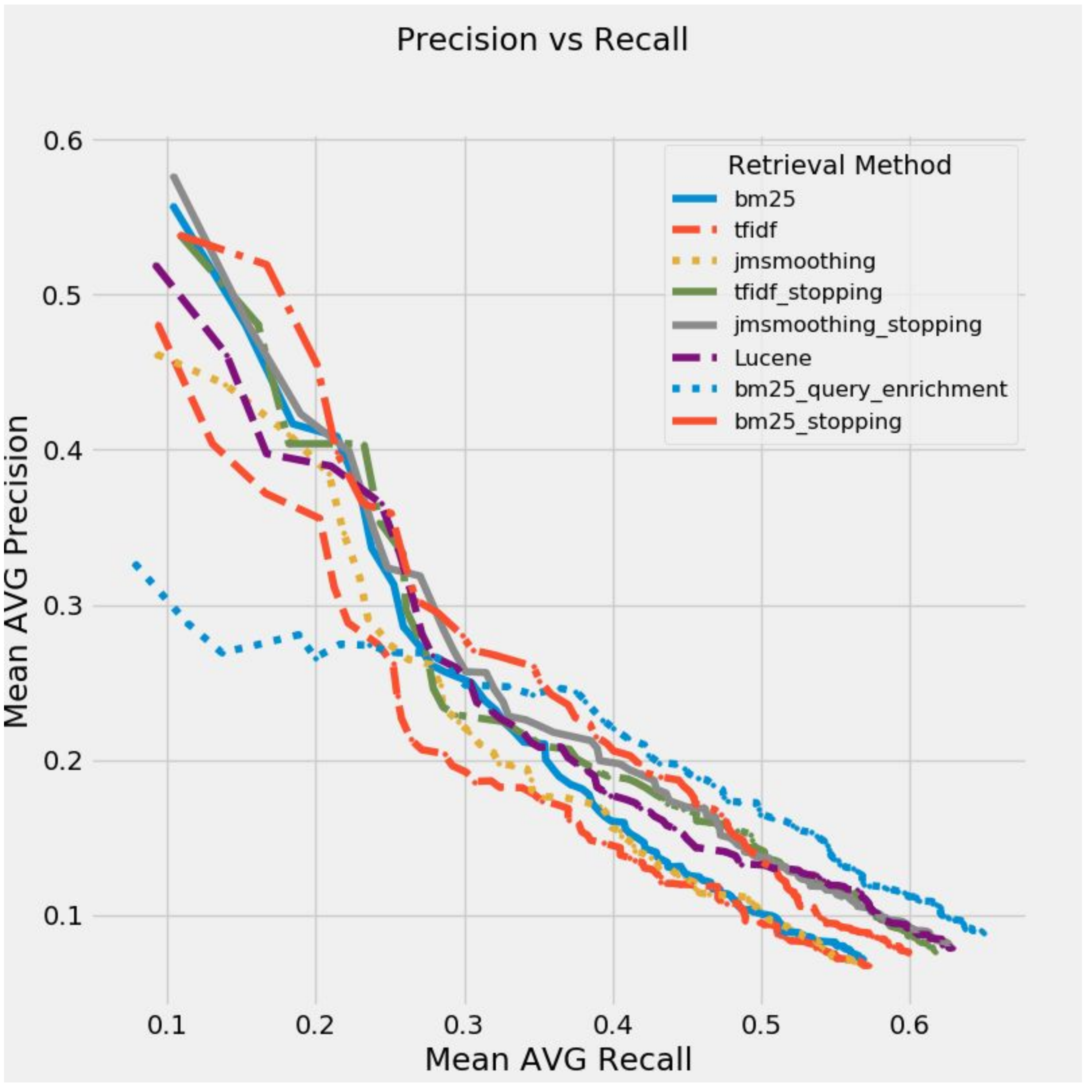


Figure 2B

Figure 2A & B: Mean Precision vs Recall Curves, (B) shows precision calculated as an average for all queries at each step. (A) is similar to (B) but precision is calculated only when recall changes. These curves are plotted against average recall for all queries at each step. In lieu of interpolation, since there are only n=64 queries, and k=100 iterations, it was tractable and quick to calculate average precision for all queries at each step and plot for a smoother visualization of changes.

Figure 2 reaffirms that TF-IDF is the worst performing model in terms of precision and recall. *Figure 2A* and *2B* differ in the time points for calculation of recall. In lieu of interpolation, since n=64 queries, and k=100 iterations, it was tractable to calculate average precision for all queries at each step for another plot to see smoother visualization of changes. BM25 is the best baseline run. Stopping can be observed to improve all baseline models with respect to both final average recall and overall average precision. Lucene performs closest to the runs with stopping included, and this is likely due to Standard Analyzer using a stop word list. The most interesting model charted was the BM25_QE. The initial average precision was poor compared to the other models, but with increasing recall it performed the best with respect to average precision. Final recall was also higher for BM25_QE compared to all others. If initial average precision could be improved, it would be the best performing model tested.

Figure 3, below, demonstrates performance of each model specific to query. Many queries like Q6 with poor MAP have great recall, this is likely due to only few (3) relevant documents. Queries like Q61 have relatively high MAP and lower recall, but, many (30) relevant documents. Q59 responded best to QE. Q59 had (42) relevant documents which is relatively high. Expanding the query likely helped boost the relevant documents and improved overall MAP.

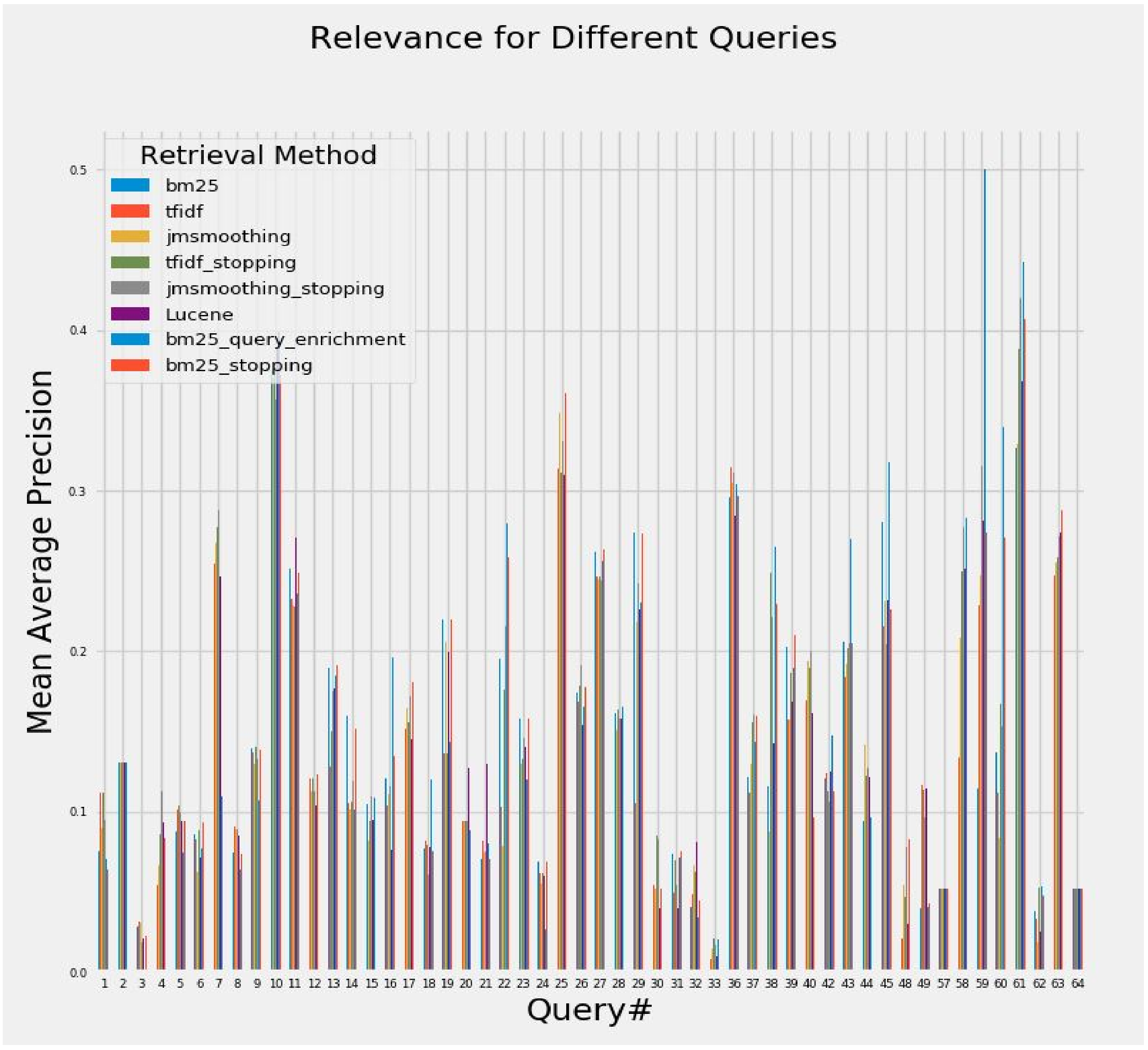


Figure 3A

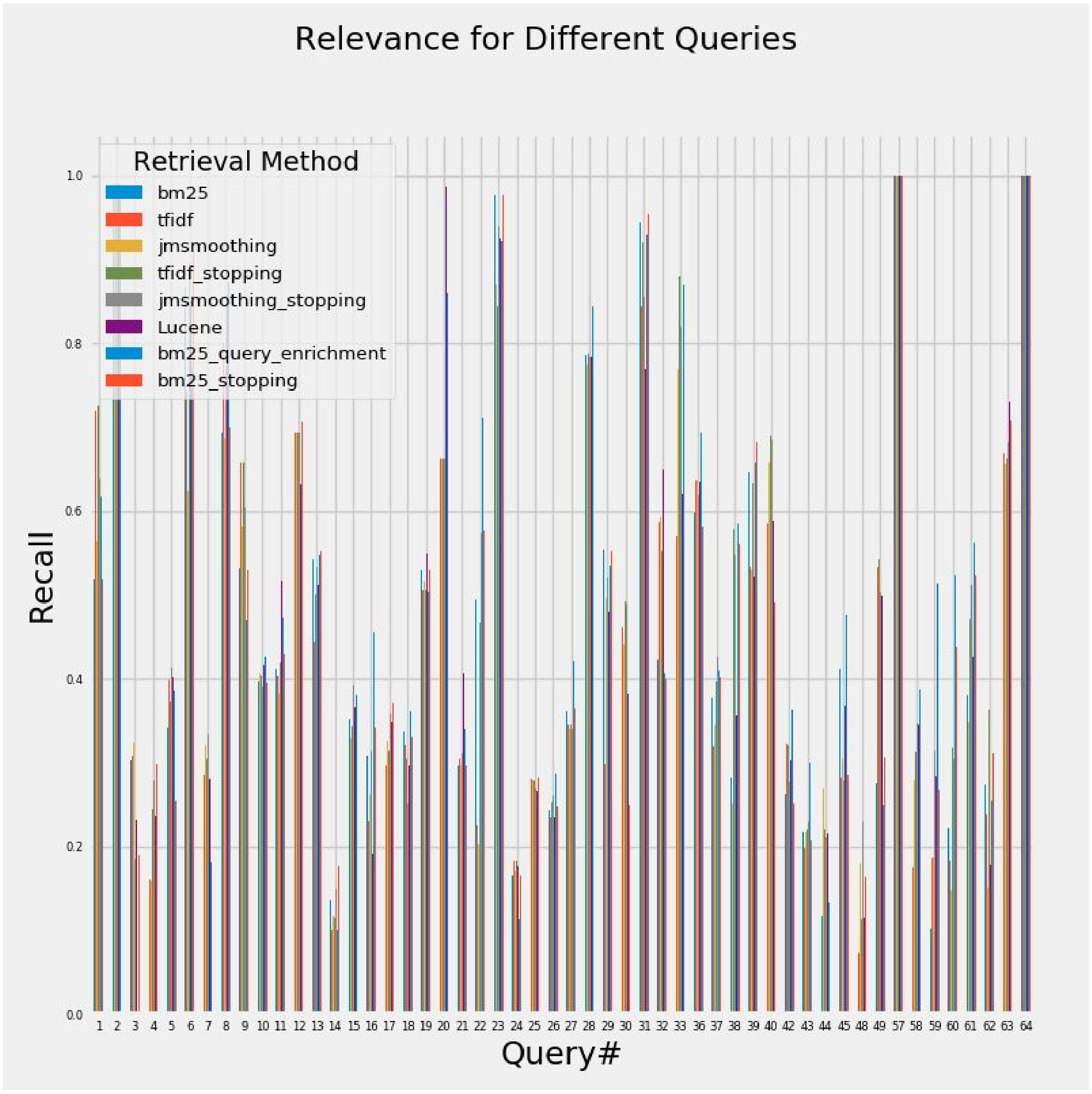


Figure 3B

Figure 3A&B: Query Level MAP and Recall at k=100.

Conclusion

Given the results of MAP and recall, BM25 with stopping performed the best over the other variations of retrieval. TFIDF consistently performed the worst across all of the retrieval variations. The MAPs were less than 0.5. This could initially be accounted for by most documents in the corpus and several of the queries being short. This does indicate there are opportunities for improvement in the retrieval methodology. Improvements that could contribute to better scoring would be utilizing the given relevance feedback in the BM25 retrieval method. Stopping showed an improvement in the retrieval, and possibly combining this with stemming would lead to improvements overall in the precision and recall of the system. In the query enrichment, results could have been improved by utilizing relevance feedback. This could be used to adjust the weights of terms that appear in confirmed relevant documents instead of assumed relevance just based on scoring high in the initial retrieval run.

V. Citations

1. Croft, W. B., Metzler, D., & Strohman, T. (2010). *Search engines: Information retrieval in practice*. Boston: Addison-Wesley.
2. Hancock-Beaulieu, M., & Gatford, M. (1997). Interactive Okapi at TREC-6. *TREC*.
3. Overview of the Third Text Retrieval Conference (TREC-3). (n.d.). Retrieved from https://dl.acm.org/citation.cfm?id=524557
4. Joachims, T. (1997). A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. *Proceedings of International Conference on Machine Learning (ICML)*. Retrieved December 04, 2018, from http://www.cs.cornell.edu/~tj/publications/joachims_97a.pdf
5. Clarke, C. L., Agichtein, E., Dumais, S., & White, R. W. (2007). The influence of caption features on clickthrough patterns in web search. *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR 07*. doi:10.1145/1277741.1277767
6. Turpin, A., Tsegay, Y., Hawking, D., & Williams, H. E. (2007). Fast generation of result snippets in web search. *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR 07*. doi:10.1145/1277741.1277766