Information Retrieval Project Report

i.

Sr. No.	Project Members' Names	Course Name	Semester	Instructor Name
1	Akshat Shukla	MSCS	1	Prof. Nada Naji
2	Parshva Shah			
3	Virat Goradia			

ii. Introduction

The project is divided into 3 phases as explained below:
(Contribution of each Project member is mentioned in brackets)

1) Phase 1: Indexing and Retrieval

It comprises of 3 Tasks as below:

1) Task 1: This task comprises of 4 steps:
    1) Step 1: This step takes Raw HTML files (given corpus) as input, and returns same number of files but with each input file tokenized by case-folding and removing punctuations
    2) Step 2: This step takes output of Step 1 as input and generates an inverted index and a document which maps Document ID and its length.
    3) Step 3: This step takes the given query file and cleans it to convert into a file containing a mapping between query id and corresponding query.
    4) Step4: This step constitutes implementing the following models for score calculation and finding top 100 documents for each:
        - BM25 (Considering no relevance)
        - BM25 (Considering relevance)
        - Lucene
        - Query Likelihood Model
        - Tf-idf
2) Task 2: In this task, Pseudo Relevance Model is used to re-compute scores calculated for BM25 Model which considered relevance. The output comprises of top 100 documents according to this model.
3) Task 3: This task comprises of 2 parts:
    1) Part A: This part consists of performing stopping on the corpus as well as queries (using common_words.txt) to perform scoring by performing 4 steps same as those in Task 1:
        1) Step 1: Tokenization
        2) Step 2: Inverted Index Generation
        3) Step 3: Query generation
        4) Step 4: Score calculation and finding top 100 documents using:
            - BM25 (Considering no relevance)
            - BM25 (Considering relevance)
            - Query Likelihood Model
            - Tf-idf
    2) Part B: This part is divided into 3 steps:
        1) Step 1: Tokenization of the given stemmed corpus (Using cacm_stem.txt)
        2) Step 2: Inverted List Creation
        3) Step 3: Finding top 100 documents by calculation scores for each using the following models:
            - BM25 (Considering relevance)
            - Lucene
            - Query Likelihood Model
            - Tf-idf
            -

2) Phase 2: Displaying Results:

This phase consists of snippet generation for the top 100 documents found for each query for each of the following models:

- BM25 (Considering relevance)
- Lucene
- Query Likelihood Model
- Tf-idf

3) Phase 3: Evaluation:

This phase consists of evaluating the results found by applying the following models for calculating scores:

- BM25 (Considering no relevance)
- BM25 (Considering relevance)
- Lucene
- Query Likelihood Model
- Tf-idf

It consists of 2 steps:

1) Step 1: This step consists of forming encoded dictionaries which are used by the Step 2 for evaluating the results generated by the above-mentioned models.
2) Step 2: This step consists of evaluating the results by calculating different evaluation measures for each result set like precision-recall tables, MAP, MRR, P@K.

NOTE: Used term-at-a-time evaluation for better efficiency. (CMS Page 168)

4) Extra-credit:

This phase consists of computing document scores considering the positions of two terms (proximity) in the same order. It consists of two parts:

1) Part A: No Stopping
2) Part B: With Stopping

iii) Literature and Resources:

Following approaches were used while performing each of the below mentioned tasks:

1) Phase 1 Task 1 Step 1: While de-punctuating, occurrences of '-', ':', ',' or '.' Within digits are retained and occurrence of '$' before a number is retained
2) Phase 1 Task 1 Step 3: For extracting queries from the given cacm_queries.txt, <ROOT> tag is surrounded around whole file and <QUERY> tag is surrounded around each of the 64 queries, followed by converting it into .xml and extracting query IDs and their corresponding queries.
3) Phase 1 Task 1 Step 4:
   - Two versions of BM25 are generated, one with assumption of no relevance consideration, while other with relevance considered. Reason for keeping BM25 without relevance is that it is used in Phase 2 (Snippet Generation) for generating snippets for all 64 queries (BM25 with relevance would have given snippets only for 52 queries whose relevance information is given according to cacm.rel.txt).
   - The Lucene model uses Standard Analyzer for tokenizing and analyzing text.
   - The smoothing parameter (λ) is set to 0.35 for computing scores using Query Likelihood Model.

- Normalization is performed to the scores computed by TF-IDF Model by dividing each score by the respective document's length.

4) Phase 1 Task 2: In pseudo-relevance feedback model, words occurring frequently in top documents are added in query to perform query expansion. (CMS Page 208).
5) Phase 2: While dividing the document content into sentences, an assumption is made that each sentence consists of a maximum of 10 tokens. The algorithms for snippet generation include a Luhn's Approach of calculating significant factor to select top sentences for summary (CMS Page 216). To find whether a word is significant or not, a modified version of frequency-based criterion has been used (CMS Page 216, 217).
6) Phase 3: For a uniform test for all runs, only the queries with relevance information given have been used, rest have been excluded i.e. 52 queries have been evaluated for each model in this case.

iv) Implementation and Discussion:

Thorough descriptions for tasks are as below:

1) Phase 1 Task 2: First, consider top "k" words from each of the top "n" documents. Consider these factors(k , n) each to be 5. Thus, 5 * 5 = 25 words are to be added to each given query, thereby performing query expansion. Since the top 5 words from each document can be a "stopword", do not include the stopwords as per 'common_words.txt'. Also, remove the words that are present in this expansion term list which is already present in the original query. Run the bm25 model with relevance model for these expanded queries.
2) Extra Credit:
    a) Generate an inverted index dictionary with term as key and (document_id, position_list) as it's corresponding value.
    NOTE: position_list denotes the list of all positions of that term in document_id.
    b) Generate bigram terms for the query. Split every bigram term and store it in an array having first bigram term as the first element and second bigram term as the second element. From the inverted index dictionary, get the value part of the first bigram term. This value part will have the documents you want, it will be a tuple (document_name, position_list). Get this second part of tuple, which is the position list.
    c) Repeat b) for the second bigram term too.
    d) For the position_list of the first bigram term, subtract each of its element by each element of position_list of the second bigram term. For each such pair having a difference in the range [4,0) which ensures no more than 3 words between these two, score will be calculated as:
    Score = difference + 5 (More score for closer words)
    and added to the total document score.
    e) Documents are ordered in decreasing order of scores for each query and top 100 are shown.

A comprehensive query-by-query analysis is placed in "Query-By-Query-Analysis.txt"

v) Results:

Results of top 100 documents for each query for each model is formatted into one .txt file in each of the directories of the code of the project.

vi. Conclusion and outlook:

Retrieval Model	Mean Average Precision	Mean Reciprocal Rank
BM25 (With Relevance)	0.591	0.873
BM25 (With Relevance With PRF)	0.342	0.5
Lucene	0.42	0.702
Query Likelihood Model	0.101	0.133
TF-IDF	0.172	0.252

- After analyzing the evaluation of top 100 documents of each query by all the models, a conclusion can be made that for the given combination of corpus and queries, BM25 with relevance considered, gave the best results (Mean Average Precision of 0.591 and Mean Reciprocal Rank of 0.873)
- As BM25 harnessed the relevance information provided in cacm.rel.txt unlike other models, it tends to show better results for this test collection.
- After trying Pseudo Relevance Feedback for BM25 (With Relevance), it is observed the evaluation values decreased as compared to BM25 (With Relevance) without pseudo relevance feedback. The assumption made of top 5 documents and top 5 words from each for query expansion could be changed to some other combination to improve the scores comparatively.
- Also, there may be a scope of improving the scoring algorithm designed by us which also considers proximity of two words.
- Moreover, if we were provided with any user query logs information (such as session history, etc), the query refinement technique which involved query expansion could use query logs as its source for better results for expanded queries than using pseudo relevance feedback. PRF is only as we have no query log information which is the best source for a knowing effective context of the query.