**INFORMATION RETRIEVAL**

**Fall 2018**

**PROJECT REPORT**

# Group Members:

Darshit Chanpura

Dharmish Shah

Soumyadeep Sinha

**Instructor:**

Prof. Nada Naji

2. Introduction:

There are 3 phases of the project and each phase has its own tasks.

Phase-1 has 3 tasks. Task-1 consists of 4 retrieval systems namely BM\_25, TF\_IDF, JM\_Query\_Likelihood (lambda value is 0.35) and Lucene’s default retrieval system. In Task-2 we did *query enrichment* using ‘pseudo-relevance feedback’ for query expansion while considering BM\_25 system as the base. In Task-3, we generated new indexes: one with no stop words and other with stemming. These we generated using the 4 retrieval systems.

Phase-2 we created a Snippet generator which generates summary of the output page as ranked and the query term highlighted in the summary.

Phase-3 we assessed the performance of the system by evaluating results from 8 distinct runs (4 from task-1, 1 from task-2, 3 from task-3). We then calculated MAP, MRR, P@K with K=5 & 20, Precision & Recall. Lastly, we plot the Recall-Precision curve for all the 8 plots in one figure.

Dharmish Shah implemented Task-1 and Phase-2 of the project and contributed about the same in the report. Darshit Chanpura implemented Task-2 and Phase-3 of the project and contributed about the same in the report. Soumyadeep Sinha implemented as Task-3 and extra-credit of the project and contributed about the same in the report.

**Literature and resources**

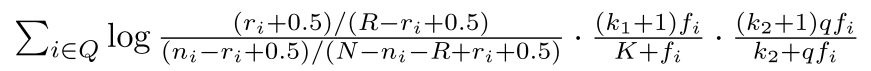
The course textbook and the slides were the primary source of reference. For programming related questions, we used the Python documentation and BeautifulSoup documentation.

**Implementation and Discussion**

**Phase 1: Indexing and Retrieval**

Task 1:

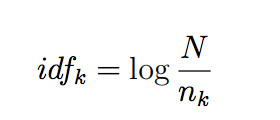
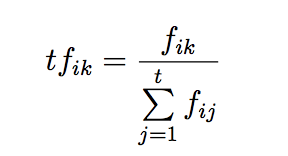
* To start with, we implemented the 4 Retrieval Systems, 3 of which (BM\_25, JM\_Querylikelihood, TF\_IDF) were coded from scratch and 4th was Lucene’s default model.
* We created a cleaned corpus using the corpus generator and then ran the 3 systems on this cleaned corpus.
* BM\_25, the first system: the assumptions – R = 0, ri = 0, b = 0.75, k1 = 1.2, k2 = 100
* Initially, we get all the queries and loop over them
* We split each query into query terms and remove stop words in the case of stop words.
* While looping over each query term, we find the idf component and for each document, we calculate the tf component using the formula given below. We did not consider relevance data.



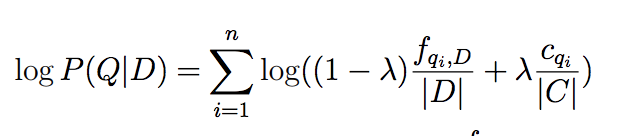


* For tf.idf, we again used the formula given in the textbook and we have used the index created by the common indexer. We multiplied both the factor
* Initially, we get all the queries and loop over them.

* We split each query into query terms and remove stop words in the case of stop words. While looping over each query term, we find the idf component initially and then multiply with the tf component for each document



* For Smoothed Query Likelihood model, we used the Formula given in the course Textbook and used the Lambda value as 0.35.
* It is different than previous models as it is a Language model and it would score all the relevant documents. Even if the query term does not exist in the document
* We initially get all the relevant documents. Relevant documents in this case is the list of documents names containing at least one query term. Then we rank all of the documents, even if the query term is not here. If the term is not found in the document, its score will add the smoothed component to the score.



* For Lucene, we used itsDefaultranking and indexing model.
* For All the retrieval system, we used term-at-a time strategy and accumulated score for each document.