GROUP 27
ASSIGNMENT 3: SEMANTIC INDEXING SEARCH ENGINE FOR Q&A DATA

- NIKOLAS SOKRATOUS: EC211277
- NEHA MUNDHE: EC211240
- SHIWANG JAISWAL: EC211249
- DAMITH SENADEERA: EC211252

OUTLINE

1. Problem statement
2. Approach
3. Dataset
4. Data preparation
5. System Architecture
6. Framework and Tools
7. Boolean Similarity
8. Lucene TF-IDF implementation (Default)
9. Bm25 Implementation
10. Language Model – Dirichlet Similarity
11. Results
12. Conclusion
13. Roles and Responsibilities
14. Demo
15. Reference

PROBLEM STATEMENT

- With the increase of documents on the internet, our queries and types of queries are also getting vast. Search Engines help users quickly find the most relevant information based on queries. Stack Overflow is a question-and-answer (Q&A) website for professional and enthusiast programmers. It only accepts questions about programming that are tightly focused on a specific problem.

- Retrieval in a question-and-answer dataset requires retrieving related questions which have been asked before with good answers for a user's query. In contrast to typical document retrieval, a retrieval model for this task should take into consideration the question similarity as well as the similarity of the relevant answers for the retrieved questions [1].

APPROACH

- LinkSo Dataset
  - After exploring several data sets such as the data dumps available from Quora, we decided to use the LinkSo Data set [2], which has been derived from the Stack Overflow question-answer data dump.
- Models analysed:
  - Lucen Default – The default similarity score based on TF-IDF
  - Boolean Model
  - BM25 – Best Match Okapi Model
  - Language Model – Paper [1] discusses for Q&A data, LM retrievals work better.
- Evaluation Criteria:
  - Precession @ 5
  - Precession @ 10
  - Recall
  - F-Measure
  - Average Retrieval Time

DATASET

- Stack Overflow Data

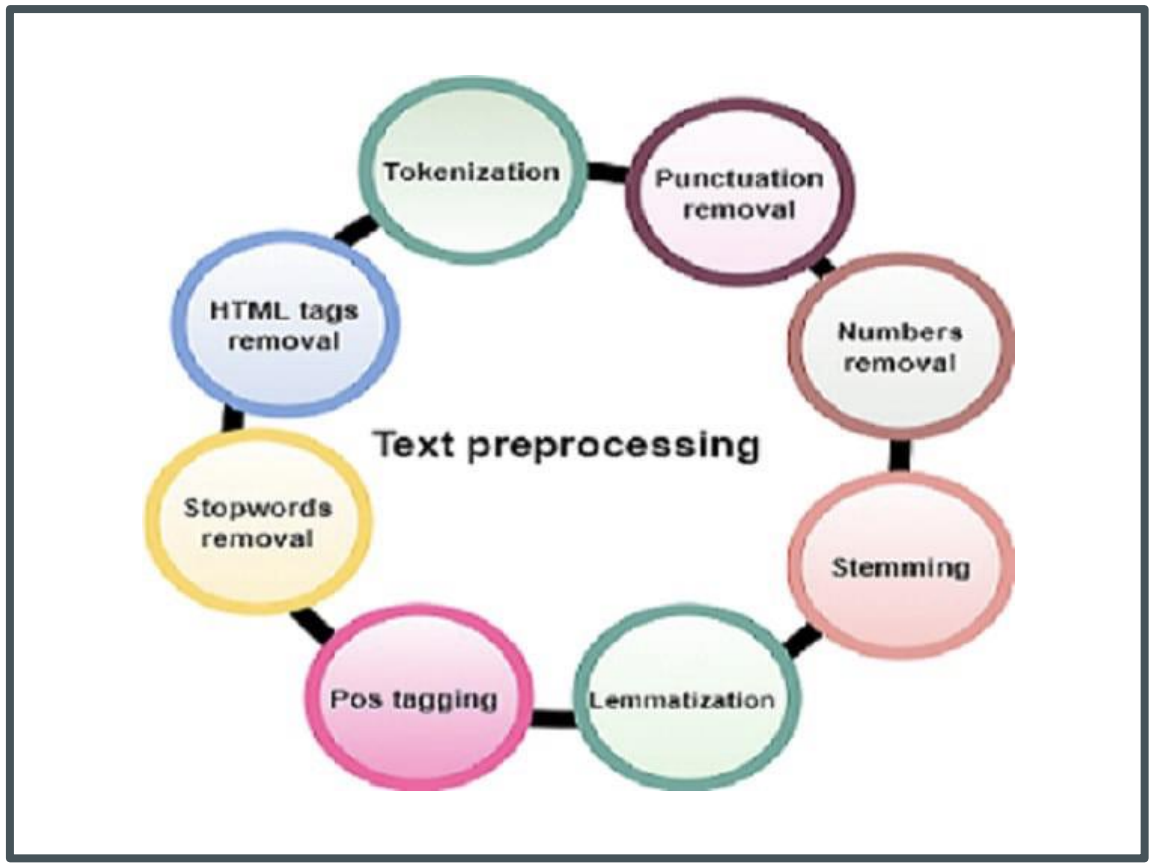
Stack Overflow has many Questions and Answers on specialized issues. The dataset is available in Stack Exchange [3] and it consists of 358 files each file has 7 more XML documents in it namely, Badges.xml, Comments.xml, Posts.xml, Posthistory.xml, Postlinks.xml, Users.xml, and Votes.xml

From these 7, we are only going to use the posts.xml file. Post files contain both answers and questions. We are going to use the questions as our sample query later in the evaluation, and the question title, body, and the two most voted answers as our target document as described in the data set below [2].

- LinkSO Dataset

The team found a benchmark dataset prepared by Microsoft for the purpose of benchmarking the retrieval of Community Question Answering (cQA) data [2], called LinkSO. We use a subset of this data set filtered by the tag “python” – the questions and their 2 most voted answers as our documents to be indexed.

DATA PREPARATION

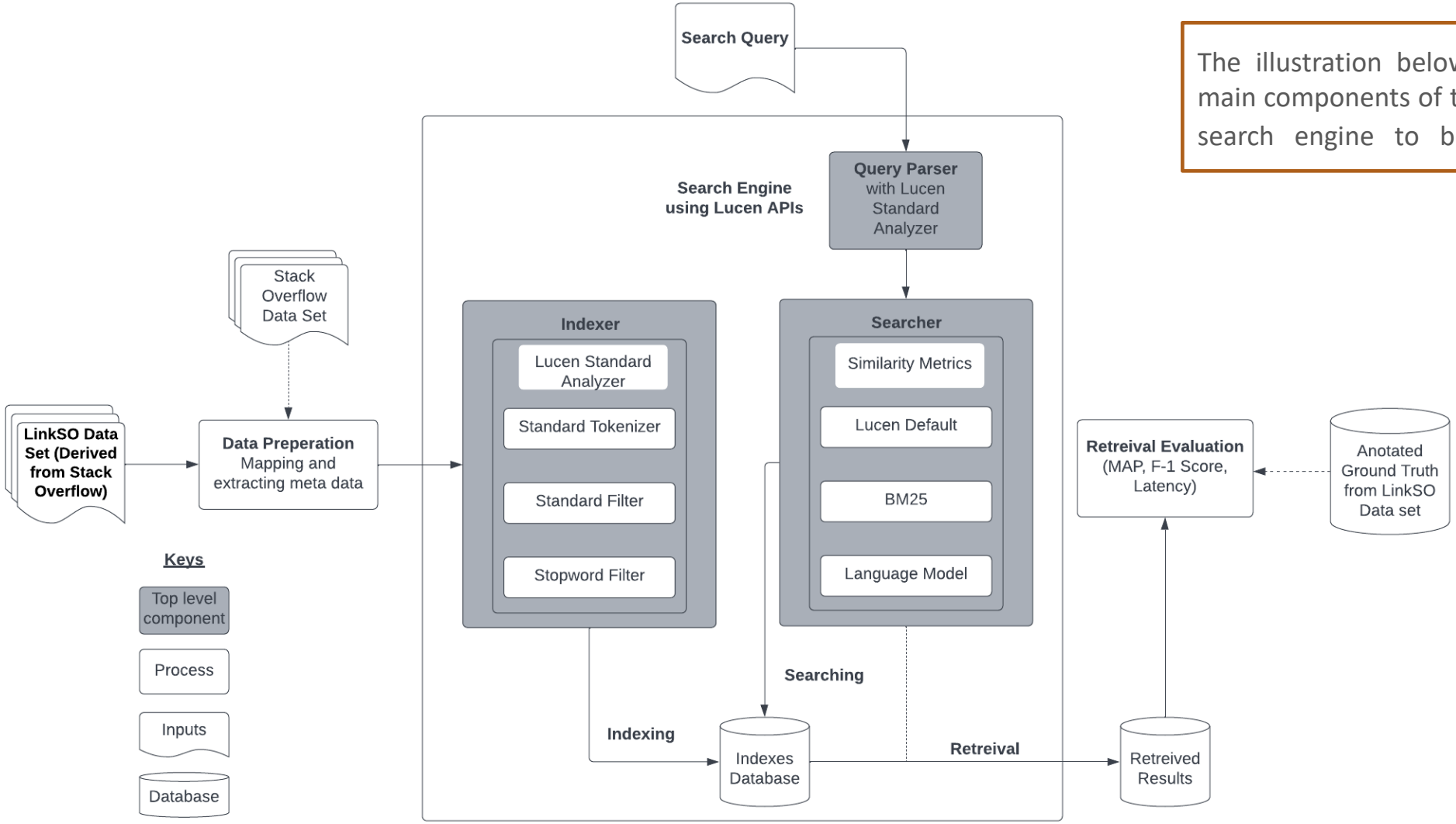


- After filtering the data set for the posts with the tag “python”, all non-Ascii characters, email addresses, URLs, code blocks, and stop words were removed from the documents to be indexed. Then if there were more than two answers to a question, all the other answers were removed and left with only the top two. So, after all this filtering the final data set contains 485,827 documents to be indexed.

- The main advantage of this data set we observed was that this group has annotated 6,410 questions with 7,406 other related questions. Therefore, we can use this data set as our ground truth for the evaluations. We will use these 6,410 questions as our testing queries to check if the annotated 7,406 related questions to these queries are retrieved to calculate the Mean Average Precision and F1 Score.

SYSTEM ARCHITECTURE

The illustration below depicts the main components of the text-based search engine to be developed.



FRAMEWORK AND TOOLS

Tool/API	Description
Python	The team will use Python as the programming language of the search engine. Python is a general-purpose language ideal for dealing with vast amounts of data.
PyLucene	"PyLucene is a Python extension for accessing Java Lucene™. Apache Lucene™ is a high-performance, full-featured search engine library. It is a technology suitable for any application that requires structured search, full-text search, faceting, nearest-neighbor search across high-dimensionality vectors, spell correction or query suggestions." [4]
Pandas / Numpy	Pandas is a python based library for manipulating datasets with ease for analysis. Numpy is a python library which helps manage mathematical functions on large numerical data sets.
GitHub	GitHub is a platform that will help in source control and collaboration.

BOOLEAN SIMILARITY

- Boolean queries are natural in systems where weights are binary. A term either applies or does not apply to a query. Each term T is associated with the set of documents D_T
  - A **AND** B : Retrieve all documents for which both A and B are relevant ($D_A \cap D_B$)
  - A **OR** B : Retrieve all documents for which either A or B are relevant ($D_A \cup D_B$)
  - A **NOT** B : Retrieve all documents for which A is relevant and B is not relevant ($D_A - D_B$)
- Consider two "unnatural" situations:
  - Boolean queries in systems that index documents with weighted terms.
  - Weighted Boolean queries in systems that use non-weighted (binary) terms.

LUCENE TF-IDF IMPLEMENTATION (DEFAULT)

$\texttt{tf}(t, d)$

	blue	bright	can	see	shining	sky	sun	today
1	1/2	0	0	0	0	1/2	0	0
2	0	1/3	0	0	0	0	1/3	1/3
3	0	1/3	0	0	0	1/3	1/3	0
4	0	1/6	1/6	1/6	1/6	0	1/3	0

x

$\texttt{idf}(t, D)$

	blue	bright	can	see	shining	sky	sun	today
	0.602	0.125	0.602	0.602	0.602	0.301	0.125	0.602

- TF-IDF: Multiply TF and IDF scores, use to rank importance of words within documents
  - Most important word for each document is highlighted

$\texttt{tfidf}(t, d, D) = \texttt{tf}(t, d) \cdot \texttt{idf}(t, D)$

	blue	bright	can	see	shining	sky	sun	today
1	**0.301**	0	0	0	0	0.151	0	0
2	0	0.0417	0	0	0	0	0.0417	**0.201**
3	0	0.0417	0	0	0	**0.100**	0.0417	0
4	0	0.0209	**0.100**	**0.100**	**0.100**	0	0.0417	0

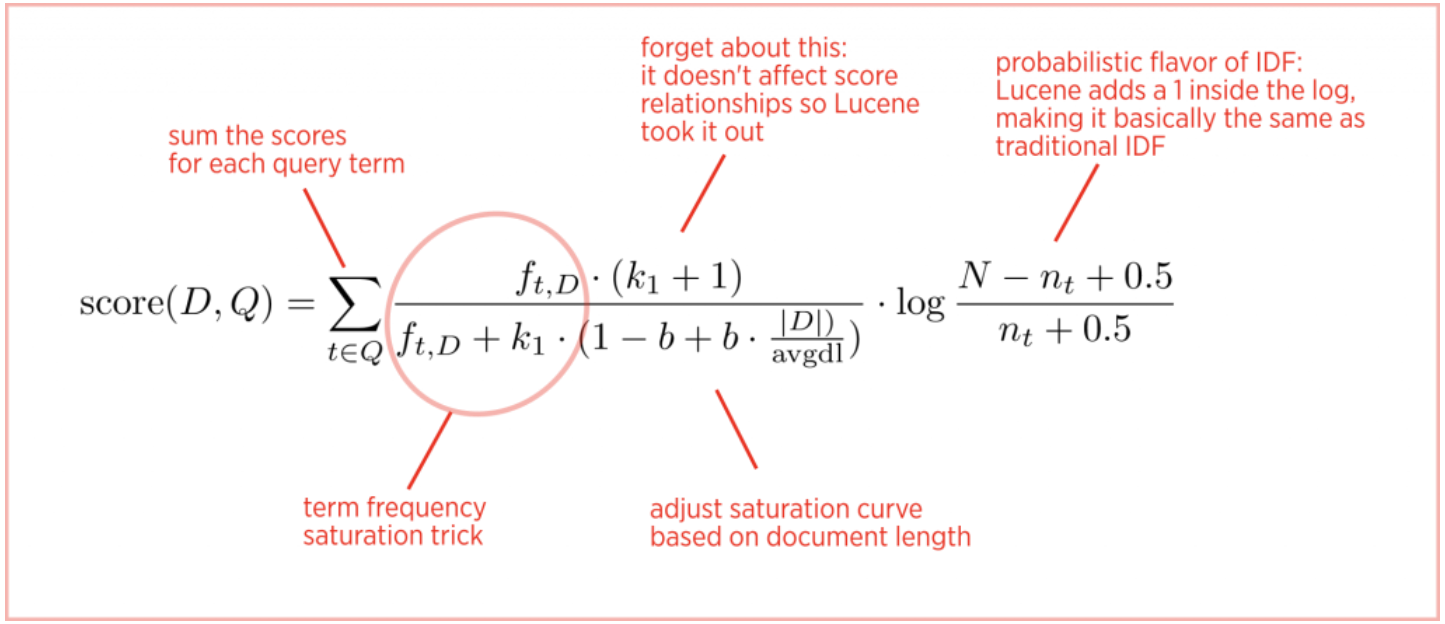
$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$

$$IDF(t) = log\frac{N}{1+df}$$

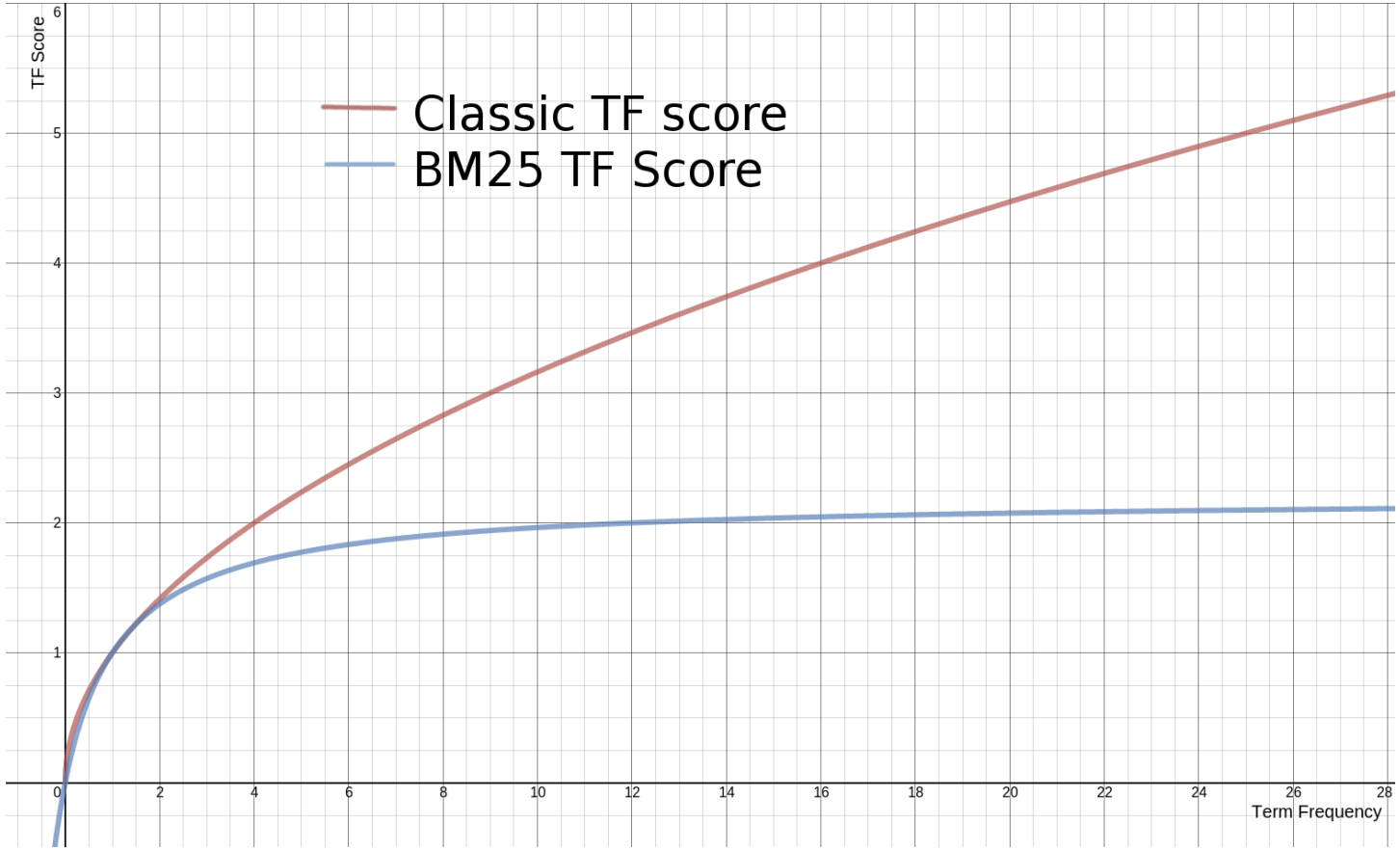
$$TF - IDF(t,d) = TF(t,d) * IDF(t)$$

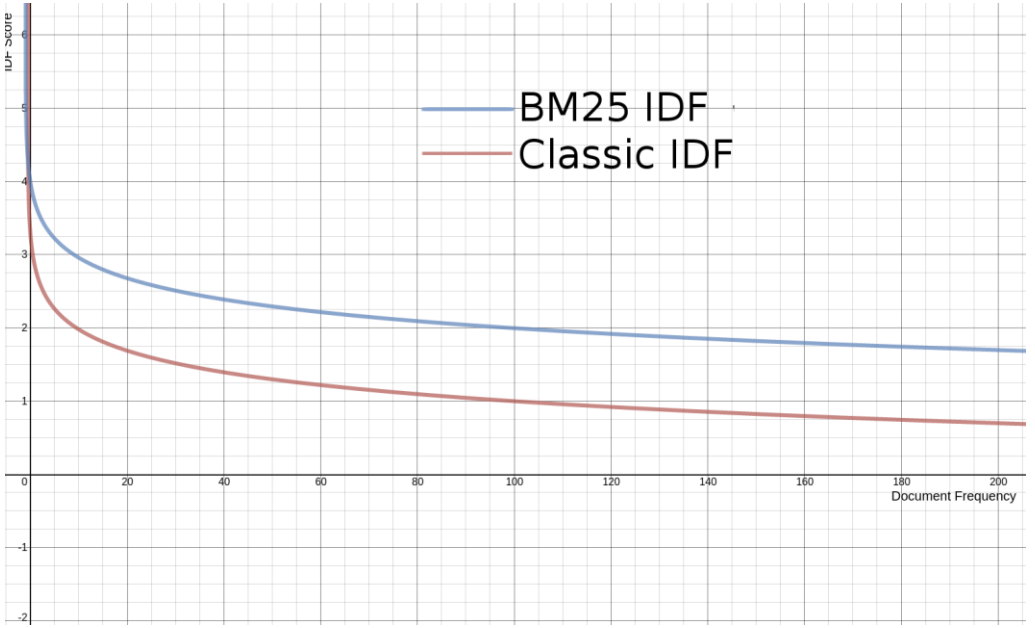
BM25 IMPLEMENTATION

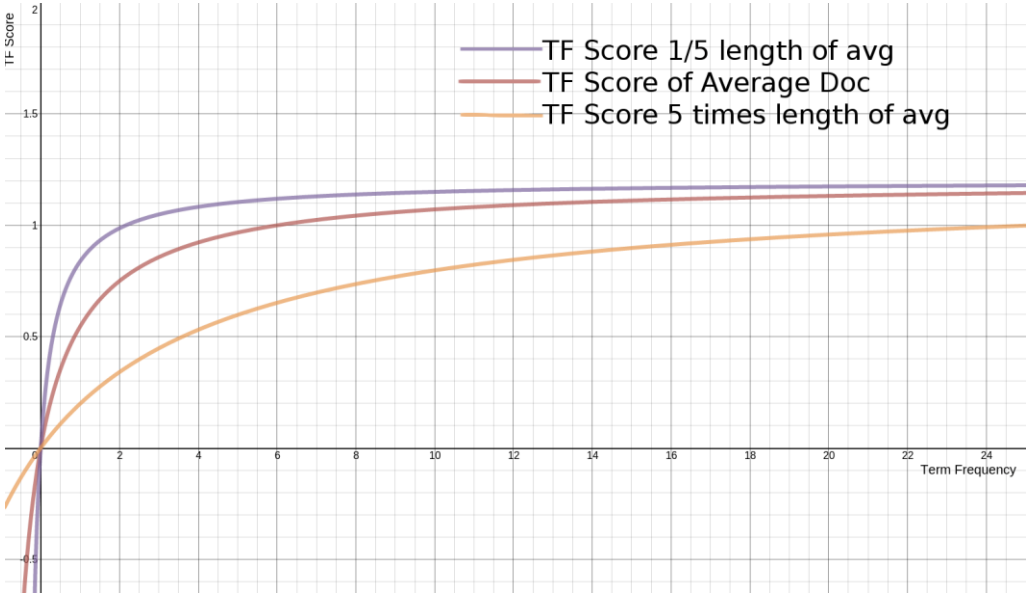
- N — Size of the Collection of documents
- nt — Number of documents in the collection containing query term ti
- f(t,D)— The frequency of term t in the document
- k1 — Determines how the tf(term frequency)component of the term weight changes as f increases.
- *b* — is a parameter to amplify saturation due to document length
- D — is the length of the document
- *avgd* — is the average length of a document in the collection



BM25 IMPLEMENTATION

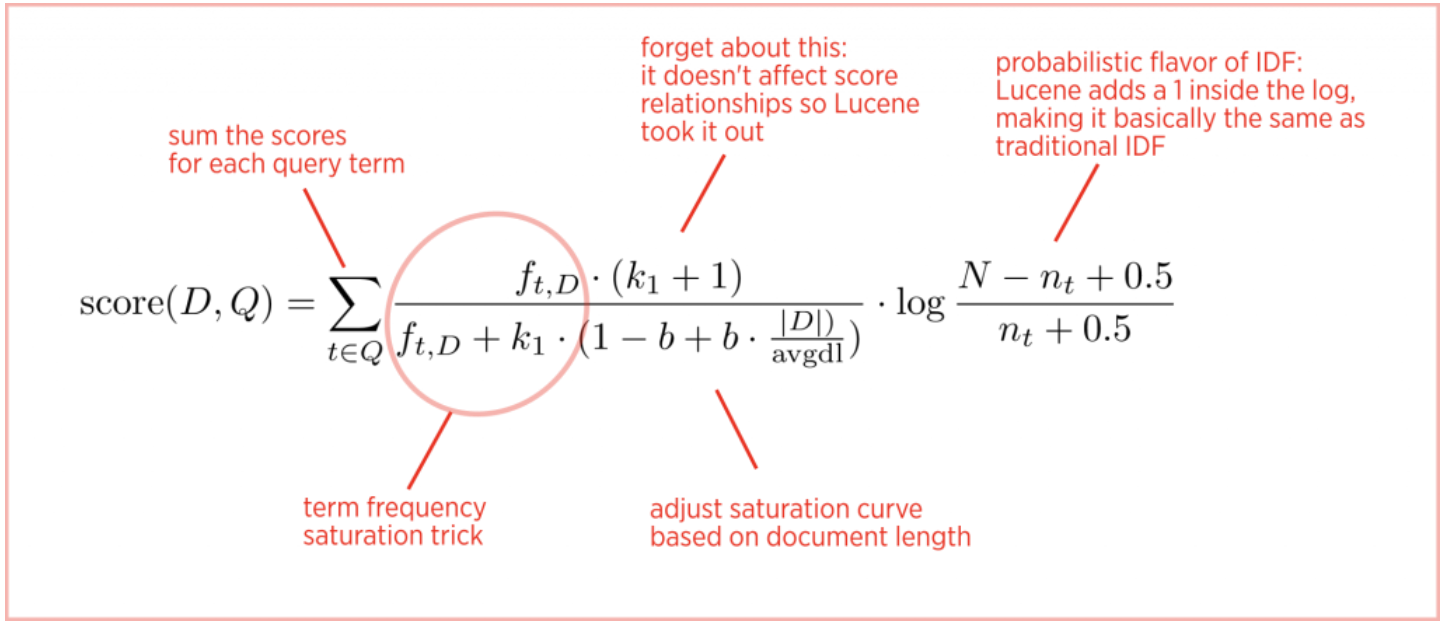




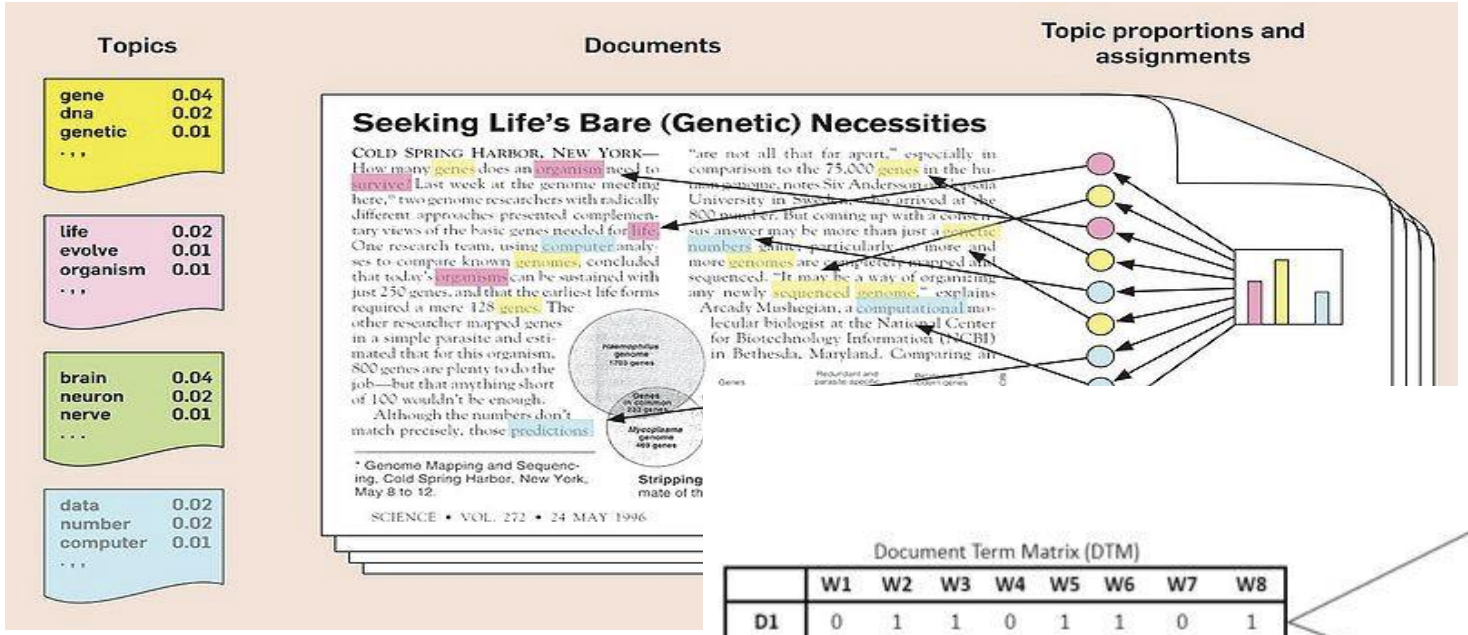


BM25 IMPLEMENTATION

- N — Size of the Collection of documents
- nt — Number of documents in the collection containing query term ti
- f(t,D)— The frequency of term t in the document
- k1 — Determines how the tf(term frequency)component of the term weight changes as f increases.
- *b* — is a parameter to amplify saturation due to document length
- D — is the length of the document
- *avgd* — is the average length of a document in the collection



LANGUAGE MODEL – DIRICHLET SIMILARITY



Document Term Matrix (DTM)

	W1	W2	W3	W4	W5	W6	W7	W8
D1	0	1	1	0	1	1	0	1
D2	1	1	1	1	0	1	1	0
D3	1	0	0	0	1	0	0	1
D4	1	1	0	1	0	0	1	0
D5	0	1	0	1	0	0	1	0

Shape: 5 * 8

Document Topic Matrix

	K1	K2	K3	K4	K5	K6
D1	1	0	0	0	0	0
D2	0	1	0	0	1	1
D3	1	1	0	0	0	0
D4	1	0	0	1	0	1
D5	0	0	1	1	0	0

Shape: 5 * 6

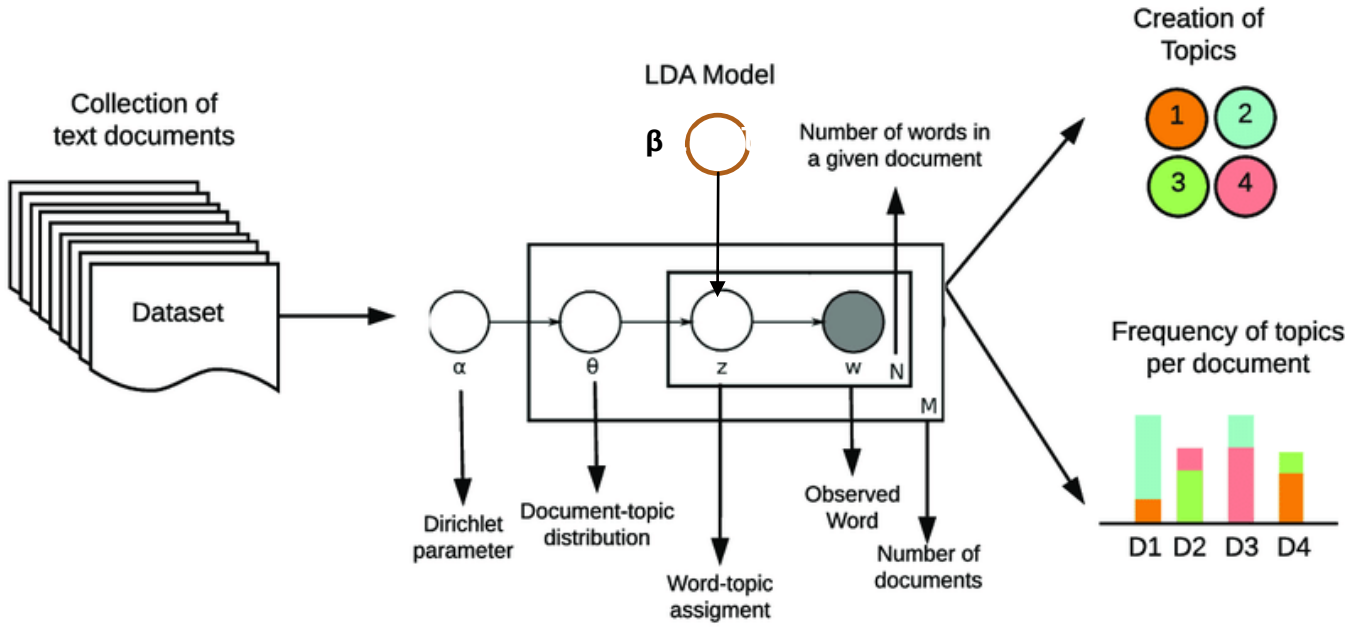
Topic Word Matrix

	W1	W2	W3	W4	W5	W6	W7	W8
K1	0	1	1	0	1	0	1	0
K2	1	1	1	1	0	1	1	1
K3	1	0	0	0	0	1	0	0
K4	1	1	0	1	1	0	0	1
K5	0	0	1	1	0	1	1	1
K6	1	0	1	1	1	0	0	1

Shape: 6 * 8

LANGUAGE MODEL – DIRICHLET SIMILARITY

- The LDA model has two parameters that control the distributions:
  - Alpha (α) controls per-document topic distribution, and
  - Beta (β) controls per topic word distribution
- To summarize:
  - M: is the total documents in the corpus
  - N: is the number of words in the document
  - w: is the Word in a document
  - z: is the latent topic assigned to a word
  - theta (θ): is the topic distribution



RESULTS (STANDARD ANALYZER)

	Precision @5	Recall @5	F-1 @5	Avg Retrieval time
TF-IDF	0.5307	0.6052	0.5451	0.001169
Boolean Model	0.5263	0.4824	0.4982	0.0004878
LM Dirichlet	0.3815	0.4210	0.3959	0.001015
BM25(default)	0.4868	0.5526	0.5012	0.001868
BM25(k=1, b=0.5)	0.5131	0.5526	0.5187	0.0004683

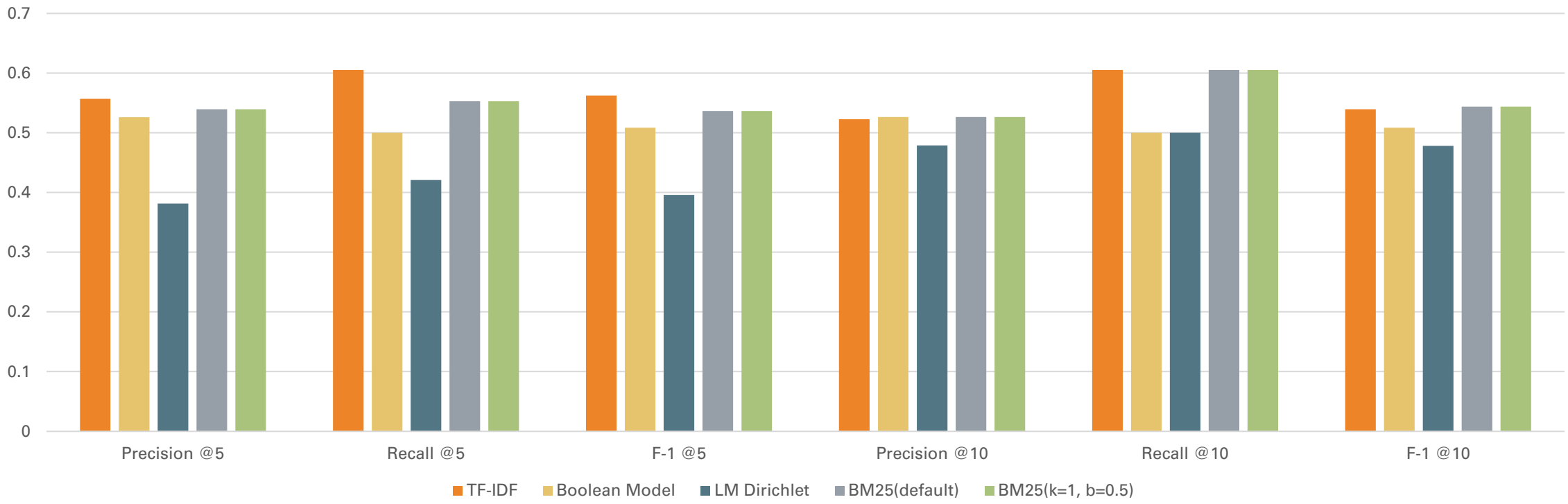
	Precision @10	Recall @10	F-1 @10	Avg Retrieval time
TF-IDF	0.5175	0.6052	0.5350	0.001056
Boolean Model	0.5131	0.5	0.5012	0.0005195
LM Dirichlet	0.4210	0.4473	0.4210	0.001832
BM25(default)	0.5175	0.6052	0.5350	0.0007659
BM25(k=1, b=0.5)	0.5175	0.6052	0.5350	0.0009924

RESULTS (ENGLISH ANALYZER)

	Precision @5	Recall @5	F-1 @5	Avg Retrieval time
TF-IDF	0.5570	0.6052	0.5626	0.0007008
Boolean Model	0.526	0.5	0.5087	0.0005468
LM Dirichlet	0.3815	0.4210	0.3959	0.001710
BM25(default)	0.5394	0.5526	0.5363	0.0009391
BM25(k=1, b=0.5)	0.5394	0.5526	0.5363	0.0007241

	Precision @10	Recall @10	F-1 @10	Avg Retrieval time
TF-IDF	0.5228	0.6052	0.5394	0.001544
Boolean Model	0.5263	0.5	0.5087	0.0009108
LM Dirichlet	0.4789	0.5	0.4780	0.001260
BM25(default)	0.5263	0.6052	0.5438	0.001309
BM25(k=1, b=0.5)	0.5263	0.6052	0.5438	0.0007389

BAR PLOT FOR THE ENGLISH ANALYZER



CONCLUSION

English Analyzer outperform the Standard analyzer by an average of 2.22% on overall accuracy and had a shorter average retrieval time of 4.66%.

TF-IDF had the best overall performance when our evaluation took only into consideration the top 5 retrieve documents

BM25 with k=1 and b=0,5 had the best overall performance when our evaluation took only into consideration the top 10 retrieve documents and when used with the English analyzer had the shortest retrieval time.

If we can use a model which also incorporate semantic and hidden relations in addition to the main bag of words, we might be able to improve the retrieval even better.

ROLES AND RESPONSIBILITIES

Team member	Role
Nikolas Sokratous	Analysis/research on data collection, Evaluation/analysis of results, repo setup/managing. (25%)
Neha Mundhe	Data cleaning/joining, Build pre-processing scripts, Analysis/research on data. (25%)
Shiwang Jaiswal	Build post-processing scripts, Retriever analysis, Implementing BM25 in the framework. (25%)
Damith Senadeera	Project structure, PyLucene engine evaluation, Indexer building, PyLucene Language Model implementation. (25%)

DEMO

REFERENCE

- [1] Xue, X., Jeon, J., & Croft, W. B. (2008, July). Retrieval models for question-and-answer archives. In Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (pp. 475-482).
- [2] Liu, X., Wang, C., Leng, Y., & Zhai, C. (2018, November). Linkso: a dataset for learning to retrieve similar question answer pairs on software development forums. In Proceedings of the 4th ACM SIGSOFT International Workshop on NLP for Software Engineering (pp. 2-5).
- [3] https://archive.org/details/stackexchange
- [4] https://lucene.apache.org/pylucene/