Project 2: MuGle

**Members**

1. Krittin Chatrinan ID 6088022  
2. Anon Kangpanich ID 6088053  
3. Tanawin Wichit ID 6088221

SUMMARY REPORT

# OVERVIEW

This reports shows backgrounds of our Jaccard, TF-IDF and BM25 (Bonus Credit) implementation.

# Members

1. **Krittin Chatrinan** ID 6088**022**
2. **Anon Kangpanich** ID 6088**053**
3. **Tanawin Wichit** ID 6088**221**

# ANSWERS

## QUESTION 1

### TF-IDF w/ Cosine Similarity VS. Jaccard Similarity

From the *Figure 1* and *Figure 3*, it can be observed that at the same k and recall value, TF-IDF yields greater precision and F1 values. This means that foreach k value TF-IDF search engine can retrieve more relevant documents proportional to non-relevant documents. Moreover, the average of interpolated Precision Curve of TF-IDF, in table 1, stays at 0.2972108 while Jaccard stays at 0.1759865 which is significantly lower; therefore, with both indications, TF-IDF search engine tends to serve the information need of the user better in terms of relevance than the Jaccard one. The statement is bold but it is true because the benchmark data provides the ground truth of relevant documents for each query which is done by human. Search engines are tested by using benchmark queries. The result will be tested against the ground truth using precision, recall, and F1 values.

The reason why TF-IDF with cosine similarity is better than Jaccard Similarity lies with in its characteristic. TF-IDF with cosine similarity does not take each term equally by using Inverted Document Frequency (IDF). In another words, some terms may have higher weight because of its rarity in a corpus.

The other reason is that TF-IDF with cosine similarity also consider weight of each document based on the frequencies of terms that occur in the query. Multiplying both TF and IDF creates a score for a term within a document. Lastly, by treating each document as a vector, TF-IDF with cosine similarity is also advantageous when it comes to comparing document and query. If we construct both document and query vectors, we can calculate its norm and calculate the angle between those vectors which is a similarity score between document and query.

However, in Jaccard Similarity, it only works on binary values indicate whether a term is present or not present in a document. Moreover, Jaccard Similarity does not take term frequency and document length into account; therefore, every term and document are the same if the term presents in the query. Thus, Jaccard Similarity is inferior when it comes to relevancies returned documents because it cutting too much corners.

**Figure 1** Average Precision Curves of our implementation of TF-IDF, Jaccard and BM25 limited at K = 50. Tested by using LISA dataset. Showing differences in the precision decrease rate of three systems.

**Figure 2.** Average Recall Curves of our implementation of TF-IDF, Jaccard and BM25 limited at K = 50. Tested by using LISA dataset.

**Figure 3.** Average F1 Curves of our implementation of TF-IDF, Jaccard and BM25 limited at K = 50. Tested by using LISA dataset.

## QUESTION 2

### Precision, Recall and F1 Curve

Here are our observations from Precision curves in *Figure 1*, Recall curves in *Figure 2*, F1 curves in *Figure 3* from all three search systems.

1. **K value is a direct variation to Recall value** – As illustrated in *Figure 2,* the recall value increases as K value increases. This implies that both value is direct variation to each other. Likewise, in the definition of Recall which is a proportion of relevant documents to all relevant documents, retrieving more documents could increase the chances to get relevance documents; therefore, Recall always increase when retrieving more documents.
2. **Different Precision at the same Recall** – As illustrated in both *Figure 4* and *Figure 5*, it can be observed that the precision values at the same recall is significantly different between Jaccard and TF-IDF. Moreover, both also show that TF-IDF consistently yields better precision in every recall value which means that TF-IDF’s search results has more relevant documents proportional to all retrieved documents.
3. **TF-IDF’s F1 is consistently greater than Jaccard’s F1** – Due to the fact that F1 is a harmonic mean between precision and recall, in another word, it quantify both precision and recall into a value, larger F1 value means larger precision and recall. Therefore, from *Figure 4* alone, it can conclude that TF-IDF search results yield higher F1 values.
4. **Decreasing Precision when k is increased** – From *Figure 1*, it can be observed that the precision curve of Jaccard is initially steep which means that it has higher decrease rate than TF-IDF. Similarly to Recall, both TF-IDF and Jaccard, in high k values, has a flatter slope which can be inferred that the decreasing rate is diminishing with higher K value or Recall.
5. **Increasing Recall when k is increased** – From *Figure 2*, it can be observed that recall curves from all search systems are initially low but later higher due to more document retrieved. The slop of all recall curves is initially high; however, at higher K values, the slope of all curves is going closing to 0 or flatter.

## QUESTION 3

### Precision-Recall plot and Search Engine Performance Evaluation

Normally, the performance test of a search engine is observed by the same precision values at the same recall values. From figure 4, it can be observed that the precision values of the TF-IDF model are higher than the precision values of the Jaccard model. Therefore, it can be summarized from the graph that the search engine using the TF-IDF model can retrieve more relevant documents than the search engine that uses the Jaccard model at the same rate of retrieving documents.

**Figure 4.** Average Precision Curves of our implementation of TF-IDF, Jaccard and BM25 limited at K = 50.

**Figure 5.** Interpolated Average Precision Curves of our implementation of TF-IDF, Jaccard and BM25 limited at K = 50.

Table 1. Interpolated average precision with the average of the interpolations.

|  |  |  |  |
| --- | --- | --- | --- |
| Recall | **Ranking Methods** | | |
| BM25 | TF-IDF | Jaccard |
| 0 | 0.428571 | 0.428571 | 0.371429 |
| 0.1 | 0.471429 | 0.414286 | 0.146939 |
| 0.2 | 0.447619 | 0.35 | 0.11619 |
| 0.3 | 0.330612 | 0.289796 | 0.069388 |
| 0.4 | 0.257143 | 0.175714 |  |
| 0.5 | 0.208163 | 0.124898 |  |
| 0.6 | 0.155844 |  |  |
| **AVG** | **0.328483** | **0.2972108** | **0.1759865** |

## EXTRA CREDIT QUESTION

### MyCoolSearcher

Our implementation of MyCoolSearcher is based on BM25, a probablisitic ranking appoarch. Our implementation starts from Indexer class, resides in TFIDFSearcher.java, which involves indexing all terms in all documents as well as collecting term frequency. ProbabilisticIndexer class, a subclass of Indexer class, which involve pre-calculating IDF. The MyCoolSearcher class will use indexed term incidence matrix from the indexer object to calculate Retrieval Status Value (RSV) for every document. With RSV values, the Searcher then sort documents by its RSV. The higher the RSV, the greater. To calculate BM25, we specify 3 tuning variables as follows: k1 = 1.2, b = 0.75, k2 = 2.0.

### Why is it better than TF-IDF with cosine similarity and Jaccard similarity?

According to the similarity model, the Jaccard model looks at both documents and queries in the form of the set and then calculates the rate of the term contained in both documents and queries and total term in both documents and queries. In the case of the TF-IDF model, it looks at both the documents and queries in the form of vector space and calculates the cosine from the angle between the documents vector and the queries vector because both term frequency and document frequency are also taken into consideration. So, it has more efficient than the Jaccard model.

However, the model that we use in MyCoolSearcher is a probabilistic model named BM25. BM25 looks at both documents and queries in the form of probability and uses criteria based on the existence of each term in documents and documents relevancy. So that is possible to create a condition in the rank calculation more than the TF-IDF Model. In addition, although both models will have the same TF and IDF calculations, the BM25 has additional document length considerations that are different from the TF-IDF model which does not consider document length. Therefore, it can be concluded that BM25 has the highest efficiency because it has more conditions that are considered than the TF-IDF model and the Jaccard model.