# Implement an Information Retrieval (IR) System

## Data Selected For Processing

---

Four Stories have been taken for testing the system and in (.txt) format:

- AIR CONDITIONING
- THE FABLE OF THE ANT AND THE CRICKET
- THE SEVEN OLD SAMURAI
- THE THREE WISHES

| Data Structure | Story Name | Text for Query 1 | Story Name | Text for Query 2 |
|---|---|---|---|---|
| **Inverted Index** | • AIR CONDITIONING<br>• The Fable of the Ant and the Cricket<br>• THE SEVEN OLD SAMURAI<br>• THE THREE WISHES | Query: last | • The Fable of the Ant and the Cricket | Query: cricket |

| | | | | |
|---|---|---|---|---|
| **Boolean Retrieval** | THE FABLE OF THE ANT AND THE CRICKET | Query: green AND autumn | • AIR CONDITIONING | Query: local AND supply |
| **Jaccard coefficient** | THE FABLE OF THE ANT AND THE CRICKET | Query = "Who cares about seven beggars" | THE SEVEN OLD SAMURAI | Query = "An early frost tinged the fields with white and turned the last of the green leaves brown" |
| **TF-IDF** | THE FABLE OF THE ANT AND THE CRICKET | Query = "Who cares about seven beggars" | THE SEVEN OLD SAMURAI | Query = "An early frost tinged the fields with white and turned the last of the green leaves brown" |
| **Cosine Similarity** | THE FABLE OF THE ANT AND THE CRICKET | Query = "Who cares about seven beggars" | THE SEVEN OLD SAMURAI | Query = "An early frost tinged the fields with white and turned the last of the green leaves brown" |

# Documentation

---

1. Reading four (.txt) files
2. Processing of all four files (lowercase, punctuation, stop words etc.)
3. Saving the respective data tokens in specific list for processing;
    - [data1 = tokens of AIR CONDITIONING]
    - [data2 = tokens of THE FABLE OF THE ANT AND THE CRICKET]
    - [data3 = tokens of THE SEVEN OLD SAMURAI
    - [data4 = tokens of THE THREE WISHES]
4. Saving all text data (words/tokens) at one list ; data = [data1, data2, data3, data4]
5. Processing with the prepared data with all five data structures
6. Inverted Index:
    - ✓ predicting single term
    - ✓ returning all the documents consisting the term
    - ✓ returning frequency of the term within each document
7. Boolean Retrieval:
    - ✓ take query from user
    - ✓ predicting outcome of single term
    - ✓ predicting outcome of multiple terms (will be complex and slow due to computation)
    - ✓ processing operations (AND, OR, NOT)
    - ✓ processing query based on all four files data
    - ✓ returning all the documents satisfying query

8. Jaccard Coefficient:
    - ✓ predicting single term
    - ✓ predicting phrase queries with more than three words
    - ✓ calculating similarity score based on query and all four documents
    - ✓ scoring relevant documents with higher score
9. TF-IDF
    - ✓ Process single terms
    - ✓ process short query of two to three terms
    - ✓ process long queries with more than three terms
    - ✓ returns an array; scoring all four documents
10. Cosine Similarity
    - ✓ Process single terms
    - ✓ process short query of two to three terms
    - ✓ process long queries with more than three terms
    - ✓ returns an array; scoring all four documents

# Information Retrieval System Output

---

| Data Structures | Query | Output | Query | Output |
|---|---|---|---|---|
| **Inverted Index** | Query: last | The given term occurs in the document number:<br>{0, 1, 2, 3}<br><br>Occurrence of term in document number and their frequency:<br>{0, 1, 2, 3}<br>0 : 1<br>1 : 2<br>2 : 3<br>3 : 1 | Query: cricket | The given term occurs in the document number:<br>{1}<br><br>Occurrence of term in document number and their frequency:<br>{1}<br>1 : 10 |

| | | | | |
|---|---|---|---|---|
| **Boolean Retrieval** | Enter your query: green AND autumn<br>['and']<br>green<br>autumn | [[0, 1, 0, 0], [0, 1, 0, 0]]<br>[[0, 1, 0, 0], [0, 1, 0, 0]]<br>Present in Document No: 2 | Enter your query: local AND supply<br>['and']<br>local<br>supply | [[1, 0, 0, 0], [1, 0, 0, 0]]<br>[[1, 0, 0, 0], [1, 0, 0, 0]]<br>Present in Document No: 1 |
| **Jaccard coefficient** | Query = "Who cares about seven beggars" | jaccard_similarity(query, data3)<br>0.017543859649122806<br><br>jaccard_similarity(query, data4)<br>0.0 | Query = "An early frost tinged the fields with white and turned the last of the green leaves brown" | jaccard_similarity(query, data3)<br>0.0<br><br>jaccard_similarity(query, data4)<br>0.0 |
| **TF-IDF** | Query = "Who cares about seven beggars" | Query result:<br><br>Array ([0, 0, 0.20997019, 0.] | Query = "An early frost tinged the fields with white and turned the last of the green leaves brown" | Query result:<br><br>Array ([0, 0.22125918, 0, 0.] |

| | | | | |
|---|---|---|---|---|
| **Cosine Similarity** | Query = "Who cares about seven beggars” | array([[1. , 0. , 0. , 0.06972064, 0. ]]) | Query = "An early frost tinged the fields with white and turned the last of the green leaves brown" | array([[1. , 0. , 0.05434862, 0. , 0. ]]) |

**Analysis:**

**TF-IDS and Cosine Similarity measure performs best in searching for long queries as the both methods account for contextual information other than just matching the terms in the document like Inverted Index, Boolean and Jaccard. On the other hand, the first three data structures works best when the length of the query is short such as comprised of one or two terms. TF-IDF and Cosine Similarity consider more relevance of rare terms in the document.**

**Inverted Index Result**

Inverted index correctly predicted the occurrence of the term (Query: “last”) in the respective documents {0, 1, 2, 3} along with their frequency
Story 1 = 0: 1
Story 2 = 1: 2
Story 3 = 2: 3
Story 4 = 3: 1

## Boolean Retrieval

Boolean Retrieval correctly predict the outcome of terms (Query:”green AND autumn”) in the respective document and also returns the output in vector form for both the terms.
[[0, 1, 0, 0], [0, 1, 0, 0]]
[[0, 1, 0, 0], [0, 1, 0, 0]]
Present in Document No:
2

## Jaccard Co-efficient

Jaccard Co-efficient correctly predicts the outcome of query in all four documents returning the higher score for the document in which the phrase (query: “"Who cares about seven beggars") occurred i-e 0.017543859649122806 for document 3/ story 3; “"THE SEVEN OLD SAMURAI".

## TF-IDF Result

TF-IDF predict correctly with score 0.209, the text from story 3, "THE SEVEN OLD SAMURAI" and also TF-IDF predict correctly with score 0.221, the text from story 2, "The Fable of the Ant and the Cricket"

## Cosine Similarity Result

"The Fable of the Ant and the Cricket" is the second story which is correctly predicted by Cosine Similarity with a similarity score = 0.0543 and the third story "THE SEVEN OLD SAMURAI" which is also correctly predicted with a similarity score of 0.069