Information Retrieval (CS F469)

Assignment – 1

Text-Based Search Engine

Documentation

A Sai Rahul	2013B1A7687P
Akshith Reddy S	2013B3A7673P
Sai Teja K	2013A7PS039P
Akhil Reddy P	2013A7PS152P

Objectives:

- To build an indexing component, which will take a large collection of text and produce a searchable, persistent data structure.
- To add a searching component, according to the vector space model.

Introduction:

The text-based search engine is based on the vector space model. It accepts the following queries:

- Single word
- Free text
- Wildcard
- Proximity

Vector Space Model:

- Vector Space Model represents documents and user queries as vectors.
- Dimensions correspond to terms occurring in the document/user query.
- "tf-idf" weighting factor has been used in the retrieval system using the formula "$w_{t,d} = \log_{10}(1 + tf_{t,d})*\log_{10}(N/df_t)$".
- The cosine similarity is calculated between the query and the documents.

Implementation:

1. Indexing
    - The Data structure used for indexing is “postings_list”.
    - In the initial step the corpus is parsed.
    - The corpus is parsed document by document.
    - The parsed text from each document is then subjected to case folding, tokenization and normalization.
    - TreebankWordTokenizer(imported from nltk library) have been used for tokenizing words.
    - PorterStemmer is used for stemming.

2. Calculating weight matrix for documents
    - The postings_list created is traversed to create the weight matrix for documents using the tf-idf weighting formula “$w_{t,d} = \log_{10}(1 + tf_{t,d})*\log_{10}(N/df_t)$” in “calculatetfidfdocs” function.

3. Query Processing
    - The search engine accepts single word, free text, wildcard and positional queries.
    - The user queries are spell checked using the “difflib” library.
    - For single word queries, the cosine similarity is calculated between the query and the documents. The first 20 documents ranked in decreasing order of cosine similarity are returned by the “search” function.
    - For free text queries, the query is subjected to tokenization and normalization. The cosine similarity is calculated between the individual query terms and the documents. The cosine similarity is calculated between the new query and the documents. The first 20 documents ranked in decreasing order of cosine similarity are returned by the “search” function.
    - For wildcard queries, the “re” library generates terms similar to the user query, which are then displayed to the user. The user is prompted to search for any one

of the displayed terms. The first 20 documents ranked in decreasing order of cosine similarity are returned by the “search” function.

- For proximity queries, the query is subjected to tokenization and normalization. The cosine similarity is calculated between the individual query terms and the documents. The first 20 documents (containing the query terms separated by the mentioned distance) ranked in decreasing order of cosine similarity are returned by the “proximitysearch” function.

Performance Metrics

Queries: exchange, random, credit, physical rubber, trade ministry and exchange, palm oil, malaysia, savings bank, purchase authorizations, official, decertification, cotton exchange, barley, european currency, stocks, central bank, acquisition.

Mean Reciprocal Rank: 0.25

Mean Average Precision: 0.658

Github Repo : https://github.com/akshith96/IR---Vector-Space-Model