Universidade Federal do Amazonas
Instituto de Computação

Aluno: Daniel da Costa Xavier
Matrícula: 21201529

Trabalho Prático de Recuperação de Informação – 2015.2

• Objetivos do trabalho

Este trabalho prático tem como objetivo o desenvolvimento de um sistema de recuperação de informação que implemente o modelo vetorial para a coleção de documentos CFC (Cystic Fibrosis Collection), que consiste de 1.239 documentos publicados entre 1974 e 1979 sobre a doença genética Fibrose Cística. Sendo assim, o sistema tem o objetivo de processar as 100 consultas da coleção, guardando as respostas retornadas em função do Record Number dos documentos.

• Implementação

O sistema foi implementado utilizando a linguagem Java.

Primeiramente os dados são extraídos dos arquivos disponibilizados e armazenados na Classe Documento. Após a extração, os documentos são armazenados em um ArrayList.

Após isto, é gerado a lista invertida dos termos presentes em todos os documentos. A lista invertida é armazenada em um HashMap, onde a chave é a string do termo e o conteúdo é uma instancia da classe TermoDocumentos. A classe TermoDocumentos é utilizada para armazenar o a string do termo, A frequencia do termo em cada um dos documentos da coleção e o valor do IDF para o termo em relação aos documentos. Na criação da lista invertida, são removidas as *stop words.*

O próximo passo é a extração das consultas, que são extraídos do arquivo de consultas disponibilizado. Os dados são extraídos e armazenados na classe Consulta, após isto, as consultas são armazenadas em um ArrayList.

Ao fim da extração dos dados, o sistema processa todas as consultas retornando o ranking de documentos ordenados pela similaridade em relação à consulta. Este processo se inicia com a geração da lista invertida dos termos da consulta.

Após isto, são gerados os vetores de cada um dos documentos e armazenados ordenadamente em relação ao Record Number em um ArrayList. É gerado também o vetor da consulta.

O próximo passo do sistema é calcular a similaridade dos documentos em relação à consulta. As similaridades dos documentos são armazenadas em um ArrayList que é atributo da classe Consulta. Por fim, o ArrayList é ordenado em ordem decrescente em relação à similaridade, gerando então o ranking de similaridade.

Com o ranking de similaridade feito, é calculado então a precisão no ponto 10 (P@10) e o MAP da consulta, levando em consideração os documentos citados como relevantes no arquivo de consulta da coleção.

Durante o processamento de cada consulta, é calculado o tempo de execução do processamento.

Todos dados solicitados (ranking de similaridades, P@10, MAP e tempo de execução das consultas) são gerados e exibidos através de um arquivo .txt gerado ao fim do processamento de todas as consultas

Na implementação foram utilizadas as seguintes bibliotecas:

- java.io → para ler e gravar os arquivos .txt;
- java.util → biblioteca que implementa as estruturas de dados utilizadas no sistema.

- **Como executar**

O link do projeto no GitHub é o seguinte:

https://github.com/danieelxavier/tp_ri.git

Após clonar, entre no diretório baixado com o nome “tp_ri”, neste repósitório será possível executar o programa. Como o programa é em Java, haverá no diretório um arquivo .jar com o nome “trabalhoRI.jar”, este arquivo que será executado.

No diretório, o seguinte comando deve ser executado:

```
java -jar trabalhoRI.jar <PATH DO DIRETORIO ONDE ESTÃO OS ARQUIVOS DA COLEÇÃO CFC>
```

Após o processamento das consultas, será aberto e um arquivo .txt chamado “resultados.txt”, gerado no diretório “tp_ri”. Este arquivo contém o relatório geral da execução do programa, com a lista de todos os documentos retornados para cada consulta com sua respectiva similaridade em relação à consulta, com o P@10, MAP e tempo de execução de cada consulta e, ainda, a média de P@10, MAP, a média de tempo das consultas,o tempo de todas as consultas e o tempo total de todo o processo.

Apesar de não ser necessário compilar o código, ele se encontra no diretório caso seja necessária sua avaliação.

- **Resultados obtidos**

Tempo de processamento das 100 consultas: aproximadamente 3.529 segundos

Média de tempo de todas as consultas: aproximadamente 0.03529 segundos

Média do MAP de todas as consultas: 0.25734744993826264

Média do P@10 de todas as consultas: 0.4119999999999998

Mais informações podem ser obtidas no arquivo gerado após a execução do programa.