Lab 01 - Parte 1 - Índice invertido e Busca Booleana

Vencimento 21 abr por 0:59 **Pontos** 30
Enviando uma caixa de entrada de texto ou uma URL de site
Disponível 13 abr em 1:00 - 23 abr em 0:59 10 dias

Esta tarefa foi travada 23 abr em 0:59.

Lab 01 - Parte 1 - Construção de Índice Invertido e Busca Booleana

Descrição

Para a primeira parte desta atividade disponibilizaremos uma **base de dados (https://drive.google.com/file/d/1YqUakSw2BLj-Hgz_oPntvHWwK6vKxjOb/view?usp=sharing)** correspondente a um conjunto de notícias políticas coletadas do **Estadão Online (http://www.estadao.com.br/)**. As notícias foram organizadas em um único arquivo para facilitar o processamento.

A atividade consiste em primeiro lugar na construção de um índice invertido (veja slides da primeira aula e capítulo 1 do livro texto) a partir do arquivo de texto disponibilizado. Como o arquivo não é muito grande, espera-se que possa ser completamente armazenado em memória RAM. Para o *dicionário use tabelas hash* e para os *postings* use *arrays de tamanho variável* ou *listas ligadas*.

Por fim, devem ser implementados algoritmos de busca booleana que utilizarão o índice invertido para responder consultas booleanas. Esses algoritmos recebem termos fornecidos pelo usuário para a realização das buscas. Três algoritmos serão exigidos:

1. 1-termo: usuário passa um termo e o algoritmo retorna os documentos associados;
2. Busca AND: todos os termos da consulta devem estar presentes na página;
3. Busca OR: algum dos termos da consulta deve estar presente na página.

Para os pontos 2 e 3 consideraremos buscas de dois termos apenas na consulta, por exemplo, "disse AND afirma".

Para avaliar a funcionalidade dos algoritmos desenvolvidos, os seus algoritmos devem estar prontos para receber consultas genéricas (de um ou dois termos), o operador booleano de interesse (AND ou OR), e retornar os ids dos documentos correspondentes. Por exemplo, se realizarmos a consulta "Campina AND Grande", a função deverá retornar uma lista de tamanho 12 com os seguintes ids: [1952, 4802, 1987, 6694, 5382, 1770, 2763, 1068, 5870, 2777, 1370, 2779]. Use esse exemplo como *sanity check* do seu algoritmo.

Rode o seu algoritmo nas seguintes consultas e imprima os resultados:

1. candidatos
2. debate, presidencial (AND e OR);
3. presidenciáveis, corruptos (AND e OR);
4. Belo, Horizonte (AND e OR)

Para cada consulta, valide utilizando os seguites asserts:

1. debate, presidenciável (AND e OR)

```
assert len(search("debate OR presidencial")) == 1770
```

```
assert len(search("debate AND presidencial")) == 201
```

2. presidenciáveis, corruptos (AND e OR)

```
assert len(search("presidenciáveis OR corruptos")) == 164
```

```
assert len(search("presidenciáveis AND corruptos")) == 0
```

3. Belo, Horizonte (AND e OR)

```
assert len(search("Belo OR Horizonte")) == 331
```

```
assert len(search("Belo AND Horizonte")) == 242
```

Descrição e Link para os Dados

Os dados podem ser obtidos **aqui** (https://drive.google.com/file/d/1YqUakSw2BLj-Hgz_oPntvHWwK6vKxjOb/view?usp=sharing) e estão estruturados em três campos, separados por vírgula, no seguinte formato:

1. titulo (string): contém o título da notícia.
2. conteudo (string): conteúdo em texto publicado na notícia.
3. idNoticia (int): identificador da notícia. Significa que você não precisa gerar Ids para os documentos.

Forma de Entrega e Critérios de Avaliação

Antes de mais nada você deve criar um repositório no github para conter os projetos da disciplina. Feito isso, **as entregas serão os links dos projetos no seu repositório da disciplina github**. Os projetos deverão ser feitos em notebooks Jupyter com código em python comentado com a descrição textual do que foi feito. Os critérios de avaliação serão:

- Legibilidade do código e organização do notebook
- Corretude da execução dos algoritmos

Dicas

Para quem não conhece, o **Jupyter** (http://jupyter.org/) é uma aplicação *web* que permite que você combine textos explicativos, equações matemáticas, código e visualizações em um único documento. **Aqui** (https://www.youtube.com/watch?v=HW29067qVWk) tem um tutorial introdutório sobre o Jupyter.

Também existem algumas bibliotecas com algumas funções prontas que podem ser úteis nesta atividade e ao longo do curso, tais como: **numpy** (http://www.numpy.org/), **pandas** (https://pandas.pydata.org/) e **nltk** (https://www.nltk.org/). Não deixem de dar uma olhada.

Entrega com atraso

Será descontado 1 ponto por dia de atraso na entrega

Bônus

Como tarefa bônus você deverá implementar o algoritmo genérico para consultas conjuntivas. Lembre que esse algoritmo recebe *n* termos de consulta como entrada e retorna os documentos que satisfazem o AND entre todos os termos. Lembre também que a ordem de execução dos ANDs intermediárias é baseada na frequência dos termos.