Indexação e Tratamento de Dados Heterogêneos: Variedade

Mecanismos de Busca

Apresentação

Raul Sena Ferreira

Mestrando - PESC / UFRJ

Coordenador técnico - IPEA

raulsf@cos.ufrj.br

www.raulferreira.com.br

https://br.linkedin.com/in/raulsenaferreira

Dicas importantes

Nota composta por:

- Entrega de exercícios semanais até a meia noite do dia anterior à aula.
- Ex: aulas nas quintas, trabalhos entregues por email até a meia noite de quarta
- Cada dia de atraso, menos 15% no valor do trabalho

Conteúdo dividido entre parte teórica e prática

- Não faltar aulas teóricas. Programar é a melhor maneira de praticar a teoria

Dicas importantes

Ritmo corrido

- Estudar além do horário de aula é fundamental
- Material de apoio: http://www-nlp.stanford.edu/IR-book/

Exercícios práticos em python ou java (preferencialmente python)

- Ao longo do curso construíremos algumas partes de um sistema de busca

Conteúdo do Bloco

Indexação

Recuperação de Informação

Solr

Elasticsearch

Busca e recuperação da informação

Também conhecido como Information Retrieval

Um sistema de information retrieval é composto essencialmente por lista invertida, indexador, processador de consulta e buscador

O que é lista invertida? O que é um indexador? Como eles estão ligados?

Como construir uma lista invertida e um indexador?

Mecanismo de busca

Segundo [Silveira, 2002], o mecanismo de busca é “um banco de dados que ajuda as pessoas a encontrar informações na Internet de acordo com palavras ou termos digitados pelos usuários”.

Motores de busca

Top 15 most popular Search Engines:
http://www.ebizmba.com/articles/search-engines

Lista invertida

Serve para ajudar a mapear em qual documento cada termo ocorre e o número de vezes que esse termo aparece

É o primeiro processo de indexação

Existem algumas variantes de lista invertida e várias técnicas para se construir a lista invertida

Lista invertida

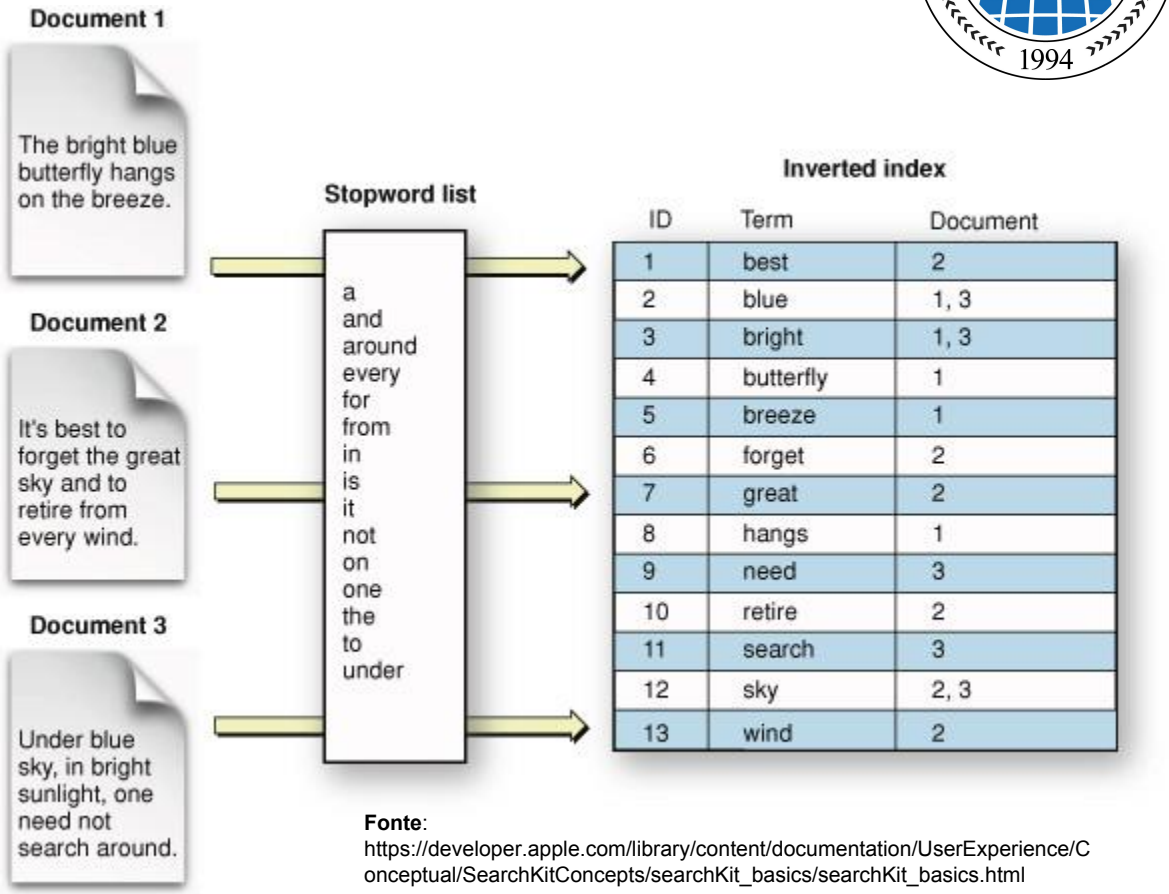
Stopwords:

termos que podem ser

dispensados para diminuir

o espaço de termos a serem

considerados pelo buscador



Fonte:
https://developer.apple.com/library/content/documentation/UserExperience/Conceptual/SearchKitConcepts/searchKit_basics/searchKit_basics.html

Lista invertida

Stemming

- Corte de sufixos de um termo
  - car, car’s, cars, cars’ -> car

Lemmatization

- Reduz termo considerando a morfologia do termo
  - saw -> see, saw
  - am, are, is -> be

Lista invertida

Stemming

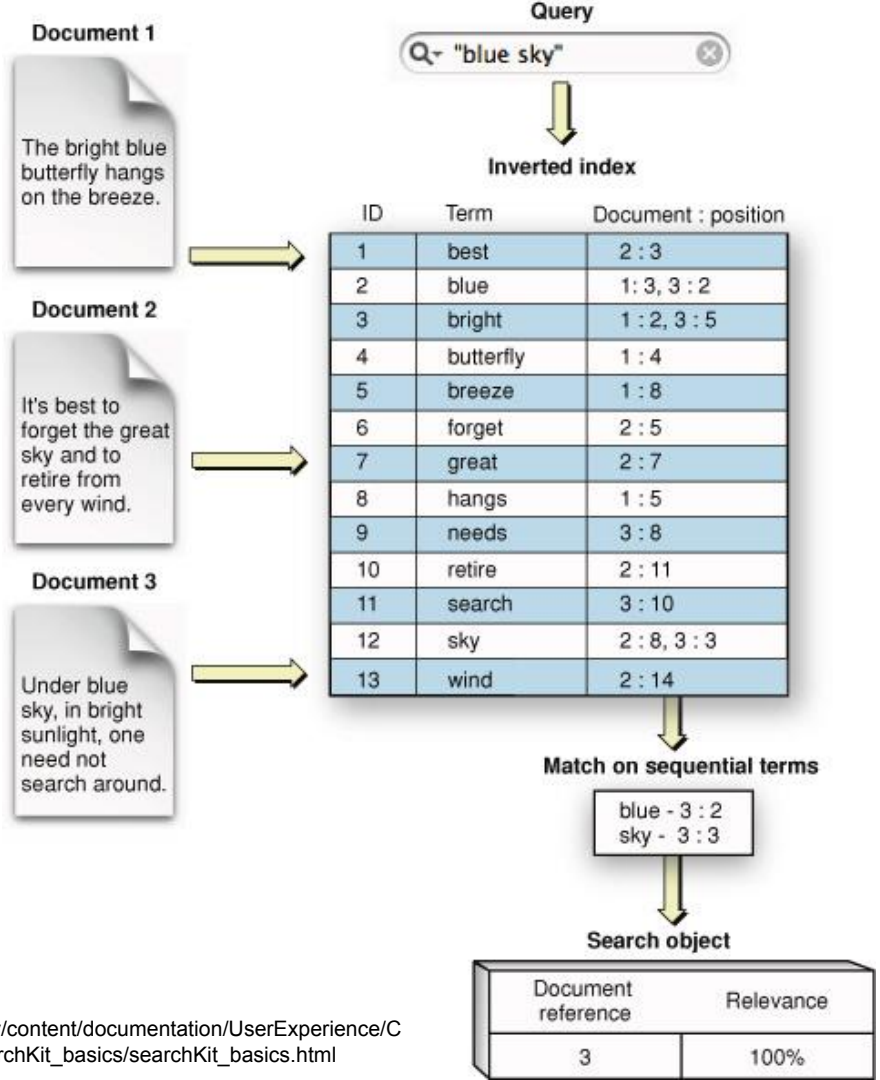
- Algoritmo de Porter
- Basicamente possui 5 fases de redução de um termo

(F)

Rule			Example		
SSES	→	SS	caresses	→	caress
IES	→	I	ponies	→	poni
SS	→	SS	caress	→	caress
S	→		cats	→	cat

Fonte:
http://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html

Lista invertida



Fonte:
https://developer.apple.com/library/content/documentation/UserExperience/Conceptual/SearchKitConcepts/searchKit_basics/searchKit_basics.html

Lista invertida

Tokenization

- Dado uma frase em um documento, deve-se construir tokens, onde estes são cada termo da frase
  - ciência de dados é o emprego do momento, os salários são altos!
    - [ciência, de, dados, é, o, emprego, do, … , altos!]
- Qual a forma correta de se “tokenizar”?
  - [..., altos!] ou [..., altos] ou [..., altos, !] ?

Lista invertida

Snowball: Processamento de palavras voltados para stemming

- http://snowballstem.org/

Pode ser usado facilmente com a ferramenta NLTK

Lista invertida

Normalização

- Retirada de acentos
- Maiúsculas/minúsculas
- outros tratamentos

Existem ferramentas muito úteis para fazer de forma automática os tratamentos descritos nos slides anteriores

- NLTK

NLTK

Ferramenta em python de código aberto bastante usada para pré processamento para problemas de linguagem natural (NLP) ou information retrieval

- http://www.nltk.org/book/

Um jeito fácil de praticar é usá-lo em conjunto com notebooks em python de forma online como o Jupyter

- https://try.jupyter.org/

Indexação

Peso dos termos

- tf-idf (term frequency-inverse document frequency)
  - TF(t) = Número de vezes que um termo t aparece em um documento / total de termos existentes no documento
    - Quão frequente um termo ocorre?
  - IDF(t) = log_e(Número total de documentos / Número de documentos contendo o termo t)
    - Quão importante é o termo?
  - Exemplo
    - Um documento contém 100 palavras onde 6 são “infnet”
      - tf(infnet) = 6 / 100 = 0.06
    - Em 1 milhão de documentos a palavra infnet aparece em 1000
      - idf(infnet) = log(1.000.000 / 1.000)
    - TF-IDF(infnet) = **tf*idf**

Modelo vetorial

Link simplificado de cálculo do modelo vetorial

http://www.site.uottawa.ca/~diana/csi4107/cosine_tf_idf_example.pdf

Modelo vetorial

Documentos podem ser representados dentro do espaço vetorial

- Modelo simples baseado em álgebra linear
- Pesos dos termos não são binários
- Permite computar um grau contínuo de similaridade entre consultas e documentos
- Permite ranquear documentos de acordo com sua possível relevância
- Permite “casamento” parcial em relação a consulta

Similaridade

Para medir a similaridade entre a consulta feita e o documento retornado usa-se algumas métricas

- Cosseno é a mais comum

$$\text{sim}(d_j, q) = \frac{\mathbf{d_j} \cdot \mathbf{q}}{\|\mathbf{d_j}\| \, \|\mathbf{q}\|} = \frac{\sum_{i=1}^{N} w_{i,j} w_{i,q}}{\sqrt{\sum_{i=1}^{N} w_{i,j}^2} \sqrt{\sum_{i=1}^{N} w_{i,q}^2}}$$

Bases de dados que podem ser usados

CSTNews corpus

- http://conteudo.icmc.usp.br/pessoas/taspardo/sucinto/cstnews.html

Cystic fibrosis

- http://people.ischool.berkeley.edu/~hearst/irbook/cfc.html

Gutenberg Corpus

- http://www.nltk.org/book/ch02.html

Wikipedia

- https://en.wikipedia.org/wiki/Wikipedia:Database_download

Próxima aula

Entrega do primeiro exercício

Métricas em information retrieval e aprofundamento do modelo vetorial de busca

Implementando algumas partes em python

Exercício

Resumo de duas páginas sobre o TED Talk de Andreas Ekström:

The moral bias behind your search results

- Escrever com suas próprias palavras sobre o que foi explicado e quais as suas impressões e opiniões sobre o que o autor disse
- Entrega na próxima semana