UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
Curso de Especialização em Big Data e Data Science

CMP269 – Recuperação de Informações

Professora: Viviane Moreira

Uso de Uma Ferramenta de RI para Realizar Experimentos

Introdução

O Zettair é uma ferramenta de Recuperação de Informações desenvolvida pelo RMIT na Austrália. Ela realiza a construção de índices e permite consultar e recuperar documentos de uma coleção já indexada. O Zettair é *open source* e suporta tanto arquivos HTML, quanto com arquivos no formato TREC (*Text Retrieval Conference*), que é uma campanha de avaliação patrocinada pelo departamento de defesa norte-americano e pelo *National Institute of Standards and Technology* (NIST).

Como vimos em aula, existem muitas outras ferramentas de recuperação de informação, algumas bem mais poderosas que o Zettair. Como o nosso objetivo nessas primeiras aulas é compreender o funcionamento das principais funcionalidades de um sistema de IR, optou-se por um sistema simples que indexa um volume considerável de documentos em poucos segundos.

Neste laboratório, iremos ver um tutorial passo a passo para indexar e consultar uma coleção de documentos. **Responda os exercícios em um arquivo texto que contenha o comando, a saída e a resposta às perguntas.**

Instalação

Há duas opções para a instalação do Zettair: (i) baixar os programas executáveis para Windows ou (ii) o código-fonte. Caso seja feito o download do executável, não é necessário fazer nenhum tipo de instalação no ambiente Windows. Os arquivos executáveis podem ser baixados na *homepage* oficial do Zettair

No Linux, devem-se baixar os fontes compactados (no formato *zip*, *gz* ou *bz2*) no site http://www.seg.rmit.edu.au/zettair/download.html e descompactá-los em uma pasta específica que deve ser criada pelo usuário. Após isso, deve-se realizar a compilação e instalação do Zettair, através dos seguintes comandos na pasta raiz do Zettair descompactado:

1 - ./configure --prefix=$HOME/local/zettair-0.9.3
2 - make
3 - make install

No passo 1, o parâmetro "-- prefix" define o caminho onde o Zettair será instalado, que pode ser diferente de onde o código fonte do mesmo foi descompactado. Mais informações disponíveis em http://www.seg.rmit.edu.au/zettair/quick_start.html

A Coleção de Documentos

Iremos utilizar a coleção de documentos Glasgow Herald que possui todas as notícias desse jornal no ano de 1995. Os documentos estão no formato TREC armazenados em arquivos em texto puro (.txt). São 311 arquivos que ocupam 150 MBytes. Cada arquivo refere-se a um dia do jornal e contém vários documentos (cada notícia é um documento). Há um total de 56.472 documentos/notícias na coleção. Os nomes dos arquivos indicam o ano, mês e dia da edição do jornal. Os arquivos possuem a extensão sgml que é uma linguagem de marcação que utiliza tags para indicar as diferentes seções dos documentos.

Os documentos devem ficar na pasta `C:\Zettair\GH95`.

A Figura abaixo mostra um exemplo de documento do Glasgow Herald que se refere a uma notícia publicada em 14 de janeiro de 1995. As partes de interesse para a nossa tarefa são os campos <DOCNO> e <TEXT> que indicam o identificador e o texto do documento, respectivamente.

```
<DOC>
<DOCNO>GH950114-000049</DOCNO>
<DOCID>GH950114-000049</DOCID>
<DATE>950114</DATE>
<HEADLINE>Argos sales power ahead</HEADLINE>
<EDITION>3</EDITION>
<PAGE>21</PAGE>
<RECORDNO>980369555</RECORDNO>
<TEXT>
ARGOS, which is breaking new ground with involvement in the power
generators' privatisation share sale, yesterday revealed a healthy
sales
boost in the run-up to Christmas.
The catalogue-shopping chain said pre-Christmas trading had been 17%
better than in 1993, while there had been a steady growth of 6% in
sales
during the whole of last year on a comparable basis.
An increase in store openings led to overall sales 13% above the
#1100m worth of turnover achieved in 1993.
Chairman David Donne said: ''This strong performance reflects our
strategy of providing the consumer with a wide range of competitively
priced merchandise, supported with more catalogues and a successful
advertising and promotions programme.''
Argos, which has been chosen as a share shop for the PowerGen and
National Power share offer, will be announcing its results for 1994 on
Monday, March 20.
</TEXT>
</DOC>
```

Indexando a coleção de documentos

Nessa seção, veremos como construir um índice com o Zettair. Para testar os comandos, é necessário:

1. Abrir uma janela de comando (tecla Windows + R e digite "cmd" na caixa de texto).
2. Navegar até a pasta C:\Zettair digitando o comando `cd C:\Zettair\GH95` no prompt de comando.

O comando a ser executado é o **`zet`**, o qual possui os seguintes parâmetros:

- Uso: `zet -i file1 ... fileN`

- Opções de parâmetros para a geração de índices:

  - `-i`

  Coloca o Zettair no modo de construção de índice (em oposição ao modo de pesquisa dentro de uma coleção já indexada).

  - `file1 ... fileN`

  Lista de arquivos na qual cada um deles contém um ou mais documentos a serem indexados. De forma alternativa, pode-se também criar um arquivo *txt* único (exemplo: *lista_arquivos.txt*), contendo uma lista com os nomes dos arquivos que devem ser indexados. Neste caso, devemos usar a opção:
  `--file-list lista_arquivos.txt`

  - `-f <nome do índice>`

  Especifica qual será o nome do índice a ser criado. Se não for especificado, o nome do índice será *index*. Caso esse índice já exista, por padrão, o programa é encerrado e um erro é informado ao usuário.

  - `--big-and-fast`

  Faz com que o Zettair use cerca de 500MB de memória durante a indexação (por padrão, cerca de 20MB são usados para isso).

  - `--stem{ none | eds | light | porters }`

  Usa um algoritmo de stemming durante a construção do índice. `none` indica que não deve ser usado stemming. `eds` remove as terminações e, ed e s. `light` é um stemmer que remove menos sufixos, porém menos efetivo que o Porter. `porters` é o Porter stemmer que retira o sufixo das palavras.

Vamos criar o índice chamado "`completo`" em que os documentos serão indexados sem nenhuma forma de stemming. Para isso, utilizaremos o comando:

```
..\zet -i -f completo --big-and-fast --stem none --file-list
.\lista_arquivos.txt
```

Se tudo funcionar corretamente, ao final da execução, o Zettair apresentará as estatísticas do índice criado. Neste caso:

```
summary: 56472 documents, 292024 distinct index terms, 24480803 terms
```

Além disso, os arquivos abaixo serão criados. Tratam-se de arquivos binários em um formato definido pela ferramenta para armazenar o vocabulário e a lista de postings. Para visualizar o conteúdo do índice em formato textual, o comando **zet_cat** pode ser usado.

```
completo.map.0
completo.param.0
completo.v.0
completo.vocab.0
```

Exercício 1) Criar um outro índice, chamado de "porter", que utilize o Porter stemmer. A seguir compare as estatísticas desse índice com as do índice completo. O que podemos observar?

Realizando Consultas no Modo Interativo

O comando para realizar consultas é:

```
zet -f <nome_do_índice> <consulta>
```

Algumas opções para a execução de consultas são:

- `-n <número de resultados por consulta>`

Permite escolher o número de resultados por consulta. O default é 20.

Exercício 2) Pesquisar pela palavra "housing" em ambos os índices criados. O que podemos observar quanto ao número de documentos que contêm esse termo? O que justifica essa diferença?

- `--summary={ plain | capitalise | tag | none }`

Permite escolher o tipo de sumarização (snippet) que aparece quando os resultados são mostrados. none significa que nenhum snippet deve ser mostrado e é o default. As outras alternativas especificam como ressaltar os termos da busca nos snippets: plain especifica que os termos da busca não devem ressaltados; capitalise coloca os termos da busca em maiúsculo e tag coloca tags <b> em torno dos termos de busca.

Exemplos de consulta:

```
..\zet -f completo -n 10 "earthquake"
```

Pesquisa no índice "completo" pelas ocorrências do termo earthquake e retorna os 10 primeiros resultados.

```
..\zet -f porter -n 2 --summary capitalise "house cat"
```

Pesquisa no índice "porter" pelos termos house e cat, retorna os dois primeiros resultados e mostra as ocorrências das palavras pesquisadas em maiúsculo.

Quando fazemos uma consulta com dois ou mais termos, o Zettair subentende que desejamos que os termos sejam combinados com "OR", ou seja, ele retorna a união dos documentos que contêm cada termo. É possível fazer consultas em que os termos sejam combinados com "AND" na qual a interseção dos documentos contendo os dois (ou mais) termos é retornada. Para isso, o AND precisa ser especificado. Por exemplo:

```
..\zet -f porter -n 2 --summary capitalise "house AND cat"
```

Exercício 3) Pesquisar pelas palavras "apple" e "banana" (em qualquer um dos índices criados) separadamente e combinadas com e sem o operador AND. O que podemos observar quanto ao número de documentos retornados nas quatro consultas realizadas? Por que o número de documentos na consulta "apple banana" (sem AND) não é igual à soma dos documentos que contêm apple e banana?

Exercício 4) Repita o exercício anterior para os termos "banana" e "ak47". O que justifica os resultados obtidos?

Executando consultas em lote

Para avaliar a qualidade de um sistema de IR, é necessário rodar um número significativo de consultas (*i.e.*, pelo menos 30) e calcular as métricas de avalição para as mesmas. As consultas utilizadas em campanhas de avaliação são comumente chamadas de *tópicos*. Um tópico é representado por uma estrutura que possui um número de identificação, um título, uma descrição e uma narrativa. O título é uma descrição bastante sucinta do tópico. A descrição fornece um pouco mais de detalhe e a narrativa auxilia as pessoas que produzem os julgamentos de relevância a distinguir documentos relevantes de não relevantes.

A seguir, apresentamos um exemplo de tópico:

```
<top>
<num> 254 </num>
<title> Earthquake Damage <title>
<desc> Find documents describing damage to property or persons caused
by an earthquake and specifying the area affected. <desc>
<narr> Relevant documents will provide details on damage to buildings
and material goods or injuries to people as a result of an earthquake.
The geographical location (e.g. country, region, city) affected by the
earthquake must also be mentioned. <narr>
</top>
```

As consultas que serão submetidas ao sistema de IR devem ser construídas automaticamente a partir dos tópicos. A forma mais comum é usar as palavras do título e da descrição excluindo algumas palavras comuns (ex: find, documents, etc) que estarão em uma lista de stopwords.

O comando para realizar consultas em lote é o `zet_trec`.

- Uso: `zet_trec -f <arquivo_de_topicos> <nome_do_indice>`

Exemplo: `..\zet_trec -f topicos05.txt porter`

A saída impressa na tela segue o formato do trec_eval que é um software bastante utilizado em campanhas de avaliação de sistemas de RI. Um extrato da saída em resposta ao comando do exemplo é apresentado abaixo.

```
251     Q0      GH950321-000003 1       9.180419        zettair
251     Q0      GH950316-000151 2       7.564804        zettair
251     Q0      GH950126-000087 3       7.284443        zettair
251     Q0      GH950511-000141 4       6.400102        zettair
251     Q0      GH950904-000067 5       6.100887        zettair
251     Q0      GH951017-000149 6       5.947077        zettair
251     Q0      GH951221-000092 7       5.730589        zettair
251     Q0      GH950902-000012 8       5.527955        zettair
                                .
                                .
                                .
300     Q0      GH950907-000043 991     1.352079        zettair
300     Q0      GH950225-000167 992     1.352079        zettair
300     Q0      GH950918-000039 993     1.351701        zettair
300     Q0      GH950302-000048 994     1.351568        zettair
300     Q0      GH951226-000106 995     1.349098        zettair
300     Q0      GH950420-000165 996     1.349098        zettair
300     Q0      GH950413-000060 997     1.347850        zettair
300     Q0      GH950424-000084 998     1.346885        zettair
300     Q0      GH950525-000210 999     1.345945        zettair
300     Q0      GH950415-000039 1000    1.345232        zettair
```

Dica: para redirecionar a saída para um arquivo, utilize o sinal > seguido pelo nome do arquivo.

Exemplo: `..\zet_trec -f topicos05.txt porter >saída.txt`

O conteúdo das colunas do arquivo é o seguinte:

- a primeira coluna possui o número da consulta;
- a segunda coluna possui o valor "Q0" para todas as linhas indicando que essa é a primeira iteração da consulta;
- a terceira coluna mostra o identificador do documento recuperado;
- a quarta coluna mostra a posição do ranking em que o documento aparece;
- a quinta coluna traz o escore de similaridade atribuído pela função de ranking; e
- a sexta coluna mostra uma string que identifica a execução do experimento.

- **Opções de parâmetros para o zet_trec:**

  - `-n <num>`

  Permite especificar o número de resultados por consulta desejados (o default é 1000).

  - `--cosine, --okapi`

  Troca a função de ranking para cosseno ou okapi. O default é o Dirichlet-smoothed language modelling.

  - `--query-stop <nome do arquivo com stopwords>` permite especificar um arquivo com stopwords que serão excluídas da busca
  - `-t` utiliza o campo <title> do tópico na consulta
  - `-d` utiliza o campo <description> do tópico na consulta
  - `-a` utiliza o campo <narrative> do tópico na consulta
  - `--big-and-fast` Faz com que o Zettair use cerca de 500MB de memória durante a execução das consultas (por padrão, cerca de 20MB são usados para isso).

Vamos executar as consultas do arquivo `topicos05.txt` sobre o índice `porter` (criado no Lab01, usando o título e a descrição para construir as consultas, `okapi` como função de ranking, excluir da busca as palavras do o arquivo `stopwords.txt` e redirecionar a saída para o arquivo `saida_okapi.txt`. Digite o comando:

```
..\zet_trec -t -d -f topicos05.txt -n 100 --okapi --query-stop
stopwords.txt --big-and-fast -r okapi porter >saida_okapi.txt
```

Se tudo funcionar corretamente, ao final da execução, o arquivo `saida_okapi.txt` será criado. Vamos abrir e explorar esse arquivo para entender a sua estrutura.

Exercício 5) Execute mais duas runs de consultas uma utilizando o cosseno como função de ranking e a outra utilizando a métrica default. Cada run deve ser identificada com uma string indicativa (ex: cosseno, default) e salva em um arquivo separado.

Exercício 6) Faça uma inspeção visual dos rankings gerados para as consultas nos três arquivos. O que pode ser observado?

Avaliando os resultados das consultas

Para avaliar quão bem o sistema de IR respondeu as consultas, é necessário conhecer as respostas "esperadas", conhecidas como **julgamentos de relevância**. A lista de documentos que deveriam ter sido recuperados para o conjunto de consultas do arquivo `TopEN05.txt` está no arquivo **qrels_GH05.txt**. Um pequeno trecho deste arquivo é ilustrado abaixo. O trecho mostra documentos que foram avaliados para o tópico de consulta `251`. O documento `GH950126-000087` foi considerado relevante para esta consulta (o que é indicado pelo número 1 na última coluna).

```
251 0 GH950107-000074 0
251 0 GH950114-000076 0
251 0 GH950124-000109 0
251 0 GH950126-000087 1
251 0 GH950202-000076 0
251 0 GH950206-000123 0
251 0 GH950208-000060 0
```

Para calcular as métricas de qualidade para um conjunto de consultas, iremos utilizar o programa **trec_eval**.

Uma versão binária para Windows encontra-se compilada no diretório em que estamos trabalhando.

Para chama-lo, o comando é:

```
treceval8.1 –c <arquivo_de_julgamentos> <arquivo_de_resultados>
```

Alguns parâmetros para o trec_eval:

-q mostra o resultado de todas as consultas (sem o –q, apenas a média para todas as consultas é mostrada)

-o mostra o resultado num formato mais descritivo (sem o –o, os resultados aparecem em um formato tabular)

O trec_eval precisa de dois arquivos de entrada: o arquivo que tem os julgamentos de relevância (qrels) e um arquivo com o ranking gerado pelo sistema de IR. Ambos os arquivos têm de estar em formato UNIX. Desta maneira, antes de usar o trec_eval, é necessário adaptar o formato. Podemos usar o programa `adapta` para fazer a conversão. A sintaxe é `adapta <nome_do_arquivo>`

Vamos então adaptar os arquivos de resultados gerados.

```
..\adapta saida_okapi.txt
```

E então avaliá-lo com o trec_eval com o comando abaixo onde `qrels_GH05.txt` é o arquivo de julgamentos e `saída_okapi.txt` é o arquivo de resultados.

```
..\treceval8.1.exe -c -o qrels_GH05.txt saida_okapi.txt
```

```
Queryid (Num):  All
Total number of documents over all queries
    Retrieved:     5000
    Relevant:       688
    Rel_ret:        564
Interpolated Recall - Precision Averages:
    at 0.00       0.6993
    at 0.10       0.6684
    at 0.20       0.6134
    at 0.30       0.5457
    at 0.40       0.4482
    at 0.50       0.4111
    at 0.60       0.3656
    at 0.70       0.3005
    at 0.80       0.2308
    at 0.90       0.1831
    at 1.00       0.1332
Average precision (non-interpolated) for all rel docs(averaged
over queries)
                  0.4017
Precision:
  At    5 docs:   0.4360
  At   10 docs:   0.3820
  At   15 docs:   0.3360
  At   20 docs:   0.3030
  At   30 docs:   0.2500
  At  100 docs:   0.1128
  At  200 docs:   0.0564
  At  500 docs:   0.0226
  At 1000 docs:   0.0113
R-Precision (precision after R (= num_rel for a query) docs
retrieved):
    Exact:        0.3619
```

Número de documentos recuperados, relevantes e relevantes recuperados

Precisão para os 11 níveis de revocação padrão (pontos da curva de revocação e precisão)

Mean Average Precision (principal métrica de avaliação de rankings)

Precisão em diversos pontos do ranking

Precisão em diversos pontos do ranking

Exercício 7) Em uma planilha (Excel, GoogleSheets, LibreOffice, etc) :

A) Crie um gráfico com as curvas de precisão e revocação para as 3 funções de ranking testadas.

B) Coloque as médias de cada uma das 50 consultas para as 3 funções de ranking (use a opção –q no trec_eval) na planilha. Em seguida, realize testes-t pareados bicaudais entre todos os pares de funções de ranking. O que podemos concluir a partir desses resultados?