Josué Santos Silva

Relatório Técnico RI Challenge

Brasil

2016, v-1.0.0

Josué Santos Silva

Relatório Técnico RI Challenge

Relatório técnico sobre o trabalho da disciplina de Recuperação da Informação que teve com objetivo de implementar e avaliar algoritmos capazes de interpretar e enriquecer consultas de usuários, provendo resultados mais relevantes que modelos de recuperação de informaçao (BM-25) e de pseudo-relevance feedback (Rocchio) de referência.

Pontifícia Universidade Católica de Minas Gerais – PUC-MG

Ciência da Computação

Brasil

2016, v-1.0.0

Resumo

O relatório apresenta informações sobre a implementação algoritmos básicos para a execução do trabalho assim como técnologias e ferramentas utilizadas no processo. Na sequencia é apresentado a posposta do módulo Expansor de Consultas para a máquina de busca. Na conclusão seram apresentados os resutltados comparativos de acordo com as métricas estabelecidas entre os algoritmos básicos e o módulo proposto.

Palavras-chaves: recuperação da informação. relevance feedback. rocchio. bm25.

Sumário

Introdução

Tendo como ojetivo do trabalho a implementação de uma máquina de busca porém com um foco no desenvolvimento de um módulo expansor de cosulta, o projeto é baseado na máquina de busca de código aberto Lucene, pelo fato de a mesma já possuir uma base sólida (LUCENE, 2010; GOSPODNETIC; HATCHER, 2005; STORE. . . ,) para implemtação do módulo propostos.

Todo o código fonte do projeto segue em licensa GPV v3[1] e está hospedado no site do github em `https://github.com/josuesasilva/jsearch`. Neste repositorio estão contidos informações para a execução do projeto assim como instruções pra compilação e demais comandos [2].

Também seram mostrados a implemetação de ranking utilizando modelo BM25 e modelo Rocchio de pseudo-relevance feedback, que serviram de base para a comparação de resultados obitidos no módulo proposto.

O projeto está separado logicamente em componentes Indexador, Componente Ranqueador, Componente Expansor de Consultas, Componente Gerador de Log. Ao final aspectos de efetividade (qualidade do ranking gerado em MAP, P@5 e nDCG)(ROBERTSON; HULL, 2000; VOORHEES; HARMAN, 2001) e eficiência (tempo de resposta) seram reportados em relação a cada componente.

[1] `https://www.gnu.org/licenses/gpl-3.0.de.html`
[2] `https://github.com/josuesasilva/jsearch/blob/master/README.md`

Parte I

Desenvolvimento

1 Ferramentas

1.1 Apache Lucene

Criado por Doug Cutting em 2000, o Lucene é uma das mais famosas e mais usadas bibliotecas para indexação e consulta de textos, disponível em código aberto. Sob o domínio da Apache Foundation, a biblioteca, escrita em java, pode ser utilizada em qualquer aplicativo J2SE ou J2EE, de código aberto ou não. Outras linguagens como Delphi, Perl, CSharp, C++, Python, Ruby e PHP devem usar os ports do Lucene para as referidas linguagens.

A biblioteca é composta por duas etapas principais: indexação e pesquisa. A indexação processa os dados originais gerando uma estrutura de dados inter-relacionada eficiente para a pesquisa baseada em palavras-chave. A pesquisa, por sua vez, consulta o índice pelas palavras digitadas em uma consulta e organiza os resultados pela similaridade do texto com a consulta.

Os índices podem ser criados em ambientes distribuídos, aumentando a performance e a escalabilidade da ferramenta.

Em particular no Lucene 4.1, o codec mudou para comprimir automaticamente o armazenamento de documentos. Ele funciona através do agrupamento de documentos em blocos de 16KB e depois comprime-los em conjunto, utilizando LZ4, um algoritmo de compressão leve. A vantagem dessa abordagem é que ela também ajuda a comprimir documentos curtos uma vez que vários documentos seriam comprimidas em um único bloco. No entanto, a fim de ler um documento único, que você precisa para descomprimir todo o bloco. De modo geral, não importa como descompressão de 16KB para 100 documentos com LZ4 é ainda mais rápido do que executar uma consulta não-trivial ou mesmo apenas buscando em um disco giratório para esses 100 documentos.

Neste projeto é foi utilizada a última versão estável do Lucene, 6.2.1. Embora já disponivel compressão de documentos por DEFLATE ainda segue utilizando LZ4 que é o método padrão desde o Lucene 4.

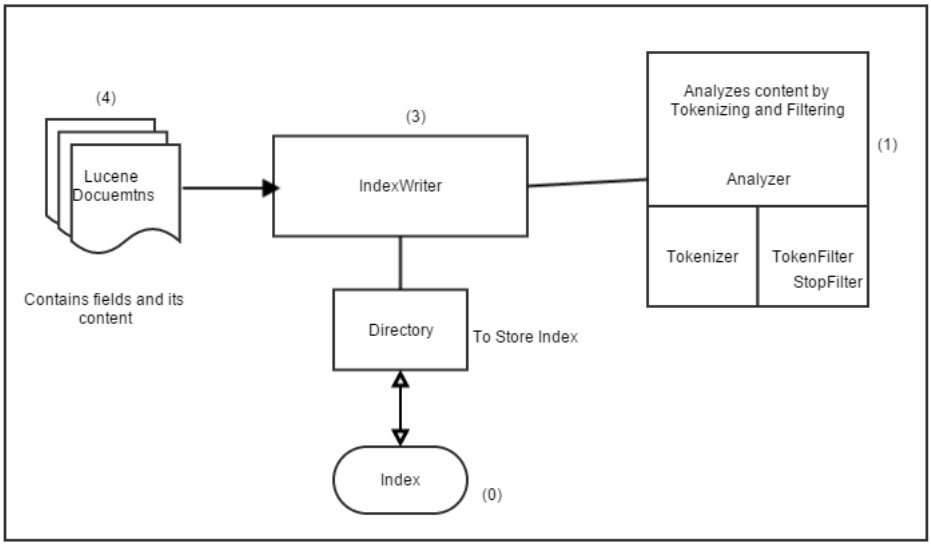


Figura 1: Componentes do Apache Lucene

2 Design da máquna de busca

2.1 Componente Indexador

Para este componente foram utilizados os recursos padrão do Apache Lucene 6, nenhum parametro interno como por exemplo compactação de documentos, que utiliza LZ4 desde a versão 4 do Lucene e parametros do BM25 não foram modificados.

2.2 Componente Ranqueador

O rannking da máquina de busca proposta utiliza a função de ranking BM25(PÉREZ-IGLESIAS et al., 2009) na qual o Lucene possui uma implementação em sua biblioteca. Todos os parametros da função foram deixadas no padrão estabelecido pela função(k1 = 1.2 e b = 0.75) que é bem indicado no caso de coleção heterogênea.

O componente de ranking também possuiu um mecanismo de expansão de de consulta, no qual foi implementado o algoritmo Rocchio(ROCCHIO, 1971; JOACHIMS, 1996), utilizando como parametros "alpha = 1.0"e "beta = 0.8". A primeira consulta foi realizado utilizandoo o ranking BM25, foram selecionados os 10 melhores resultados como os resultados relevantes para a execução do restante do algoritmo, além de que a quantidade de termos em que a consultada foi expnaida foi considerado um valor de no máximo 10 termos.

Na seção de resultado segue dados a respeito do desempenho do componente.

2.3 Componente Expansor de Consultas

2.4 Componente Gerador de Log

Parte II

Resultados

Resultados

2.5 Componente Indexador

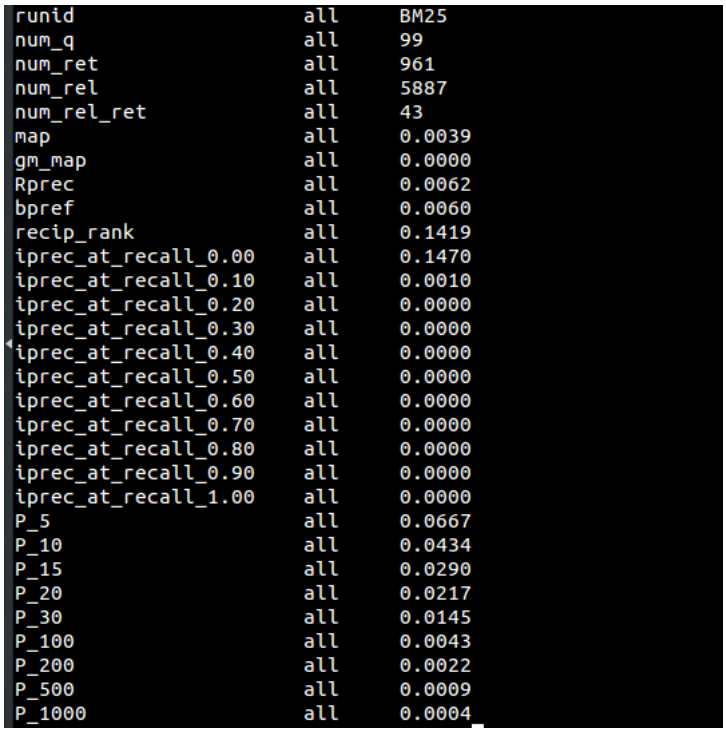
Segue abaixo dados a respeito do desempenho do indexador:

Tabela 1: Dados de desempenho

Documentos indexado	17123
Tempo de indexação	20,31 seg.
Documentos indexados por segundo	843
Tempo de resposta a consultas ao índice	0,007 seg.
taxa de compressão	25%

2.6 Componente Ranqueador

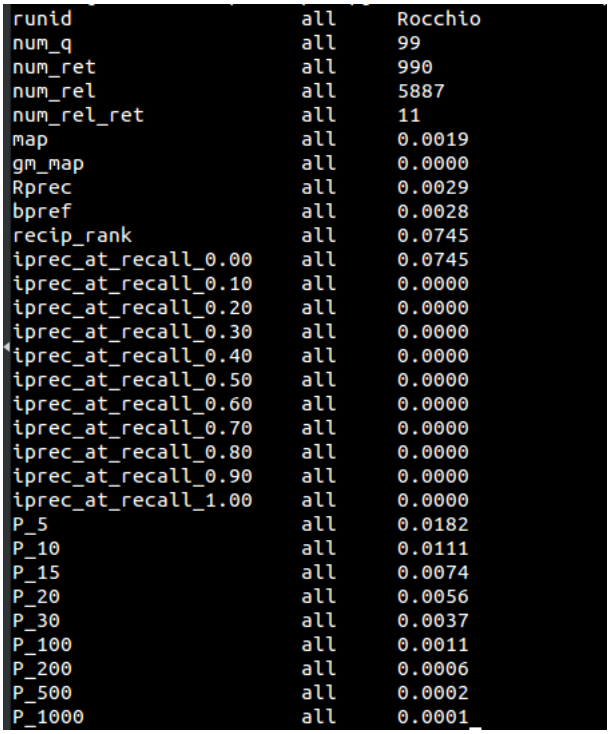
Desempenho de ranqueador BM25, considerando um ranking entre os 10 melhores resultados.



```
runid                   all     BM25
num_q                   all     99
num_ret                 all     961
num_rel                 all     5887
num_rel_ret             all     43
map                     all     0.0039
gm_map                  all     0.0000
Rprec                   all     0.0062
bpref                   all     0.0060
recip_rank              all     0.1419
iprec_at_recall_0.00    all     0.1470
iprec_at_recall_0.10    all     0.0010
iprec_at_recall_0.20    all     0.0000
iprec_at_recall_0.30    all     0.0000
iprec_at_recall_0.40    all     0.0000
iprec_at_recall_0.50    all     0.0000
iprec_at_recall_0.60    all     0.0000
iprec_at_recall_0.70    all     0.0000
iprec_at_recall_0.80    all     0.0000
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.0667
P_10                    all     0.0434
P_15                    all     0.0290
P_20                    all     0.0217
P_30                    all     0.0145
P_100                   all     0.0043
P_200                   all     0.0022
P_500                   all     0.0009
P_1000                  all     0.0004
```

Figura 2: Resultado BM25

Desempenho de ranqueador BM25 executando a query expandida pelo Rocchio, considerando um ranking entre os 10 melhores resultados.



```
runid                   all     Rocchio
num_q                   all     99
num_ret                 all     990
num_rel                 all     5887
num_rel_ret             all     11
map                     all     0.0019
gm_map                  all     0.0000
Rprec                   all     0.0029
bpref                   all     0.0028
recip_rank              all     0.0745
iprec_at_recall_0.00    all     0.0745
iprec_at_recall_0.10    all     0.0000
iprec_at_recall_0.20    all     0.0000
iprec_at_recall_0.30    all     0.0000
iprec_at_recall_0.40    all     0.0000
iprec_at_recall_0.50    all     0.0000
iprec_at_recall_0.60    all     0.0000
iprec_at_recall_0.70    all     0.0000
iprec_at_recall_0.80    all     0.0000
iprec_at_recall_0.90    all     0.0000
iprec_at_recall_1.00    all     0.0000
P_5                     all     0.0182
P_10                    all     0.0111
P_15                    all     0.0074
P_20                    all     0.0056
P_30                    all     0.0037
P_100                   all     0.0011
P_200                   all     0.0006
P_500                   all     0.0002
P_1000                  all     0.0001
```

Figura 3: Resultado Rocchio

2.7 Componente Expansor de Consultas

2.8 Componente Gerador de Log

Referências

GOSPODNETIC, O.; HATCHER, E. *Lucene*. [S.l.]: Manning, 2005. Citado na página 5.

JOACHIMS, T. *A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization*. [S.l.], 1996. Citado na página 11.

LUCENE, A. *Apache Lucene-Overview*. 2010. Citado na página 5.

PÉREZ-IGLESIAS, J. et al. Integrating the probabilistic models bm25/bm25f into lucene. *arXiv preprint arXiv:0911.5046*, 2009. Citado na página 11.

ROBERTSON, S. E.; HULL, D. A. The trec-9 filtering track final report. In: *TREC*. [S.l.: s.n.], 2000. p. 25–40. Citado na página 5.

ROCCHIO, J. J. Relevance feedback in information retrieval. Prentice-Hall, Englewood Cliffs NJ, 1971. Citado na página 11.

STORE compression in Lucene and Elasticsearch. https://www.elastic.co/blog/store-compression-in-lucene-and-elasticsearch. Accessed: 2016-09-26. Citado na página 5.

VOORHEES, E. M.; HARMAN, D. Overview of trec 2001. In: *TREC*. [S.l.: s.n.], 2001. Citado na página 5.