O BACO e a sua API

Luís Sarmento
FEUP & Linguateca

Mini-Jornadas de PLN - Univ. Minho
8 de Novenbro de 2006

Motivação (1)

- Muitas sub-tarefas e aplicações de PLN podem usar técnicas que recorrem intensivamente a dados (i.e. texto):
  - aquisição lexical (terminologia, nomes)
  - identificação de relações / grupos semânticos
  - desambiguação de sentidos
  - resposta automática a perguntas
  - etc…

Motivação (2)

- Acesso a grandes quantidades de dados faz-se:
  - tradicionalmente recorrendo a grandes corpora (Cetempúblico, BNC…)
    - 100-200M palavras para pesquisa orientada linguisticamente.
    - alguns marcados POS. Normalmente especializados num género
    - com interface web mas normalmente sem API
  - Mais recentemente usando as API's dos motores de pesquisa (Google e Yahoo):
    - Web "portuguesa" + "brasileira". Milhares de M palavras
    - Texto cru. "Todos" os géneros. Orientados para RI
    - API próprio com acesso limitado (ex: 1000 pesquisas / dia)

Motivação (3)

- Têm surgido grandes colecções de docs em português:
  - WBR99, WPT03 (WPT05 em breve)
  - dezenas de Gb de texto para pesquisa (local)
  - sem limitações de acesso
  - Permitem a marcação linguistica (havendo ferramentas)
- Mas não pesquisáveis directamente:
  - não "limpos": divisão e remoção de duplicados
  - sem indexação nem métodos de acesso simples
- Isto afasta os desenvolvedores que já têm muito em que pensar: manipular 15Gb de texto dá trabalho!!

BACO - Base de Co-Ocorrências

- Uma BD de texto produzida a partir do WPT03
  - Pesquisa fácil sobre 6Gb de texto (1000M pals.)
- Dados preparados para facilitar certo tipo de pesquisas com interesse linguístico:
  - Pesquisa directa sobre "frases"
  - Pesquisa de contextos (em janela)
  - Pesquisa de co-ocorrências
- Usa um SGBD MySQL para armazenamento, indexação e processamento de queries…
- API Perl simples de usar!

A tabela das frases e metadados

- WPT03 distribuído em XML:
  - 15GB -> 6Gb (remoção de duplicados Nuno Seco)
- Divisão do XML em dois ficheiros tabulares:
  - metadados dos documentos: 1,529,758 docs
  - "frases" - para cada documento foi "fraseado" usando as ferramentas PT::PLN: 35,575,103 frases
- Dados tabulares carregados para a BD
- Índices:
  - "full-text" para o campo texto da tabela frases
  - "B-tree" para os restantes campos relevantes

Necessidades de outras tabelas

- Para muitas tarefas, a pesquisa sobre documentos / texto não é a mais útil
- É também importante ter informação acerca da frequências de sequências de palavras
  - Palavras simples (dicionário): p_1 f d
  - Bigramas: p_1 p_2 f d
  - Trigramas: p_1 p_2 p_3 f d
  - Tetragramas: p_1 p_2 p_3 p_4 f d
- Permite por exemplo responder questões como:
  - Que contextos de 3 palavras que antecedem Y?

Cálculo das frequências de n-gramas

- Processamento de cada frase da tabela
- Recursos a hash Perl para obter as contagens:
  - 1-gramas: 6.7 M
  - 2-gramas: 54.6 M
  - 3-gramas: 173.6 M
  - 4-gramas: 293.1 M
- Para 2, 3 e 4-gramas foi feito um processamento por etapas com a fusão final (external merge-sort)
  - muitas centenas de horas de CPU, muitos GB em disco
  - alguns erros mas não muito graves…

Armazenamento e Indexação

- Armazenamentos em tabelas simples

```
CREATE TABLE `wpt_3_gramas` (
        `p1` varbinary(255) NOT NULL default '',
        `p2` varbinary(255) NOT NULL default '',
        `p3` varbinary(255) NOT NULL default '',
        `f` int(10) default NULL,
        `d` int(10) default NULL,
        KEY `idx_wpt_3g_p1` (`p1`),
        KEY `idx_wpt_3g_p2` (`p2`),
        KEY `idx_wpt_3g_p3` (`p3`)
        )
                ENGINE=MyISAM
                DEFAULT CHARSET=latin1
                MAX_ROWS=176000000
                AVG_ROW_LENGTH=40
```

- Índices B-tree: um por coluna

A Tabela de Co-ocorrências

- Em muitas tarefas de PLN (por exemplo DSP) torna-se necessário saber que palavras ocorrem (antes ou depois) de determinada palavra.
  - No mesmo documentos, parágrafo, frase
- Por isso, optou-se por obter essa informação para todas as "palavras" do WPT (com mais de 4 caracteres) e armazená-la
  - Âmbito: co-ocorrências no interior da frase

Processo

- Para cada frase, obter todos os pares de palavras existentes mantendo a informação da ordem: p_1 p_2 f_{frase}
- Repetir o processo para as 1.5 M de frases
- Acumular os resultados: 761M pares/tuplos
- Processamento por etapas
  - merge sort externo
  - centenas de horas de CPU e vários GB disco

Estatísticas Globais

tabelas	# tuplos (milhões)	Tamanho dados (GB)	Tamanho Índice (GB)
Metadados	1.529	0.2	0.05
frases	35.575	6.55	5.90
1-gramas	6.834	0.18	0.27
2-gramas	54.610	1.50	0.92
3-gramas	173.608	5.43	2.97
4- gramas	293.130	10.40	6.35
co-ocorrências	761.044	20.10	7.56
BACO total	-	44.4	~ 24

E então?…

- Tudo isto só é interessante se as pesquisas forem eficientes:
  - Esperamos ter trocado a espaço em disco por eficiência de pesquisa: 6GB -> ~70 Gb
  - Muita informação repetida / redundante permite pesquisas mais eficientes (supostamente)
    - Mas há algumas idiosincrasias do MySQL…
- Se for simples aceder à informação…

Modelo de Utilização BACO

- Um servidor de MySQL como armazem de dados e como processador de queries
  - servidor local ou remoto (ou paralelo?)
- Uma API simples de utilizar que isole o programador dos detalhes de acesso ao ficheiro:
  - para já só em Perl
- Programas cliente que façam uso da informação disponível no servidor e da API

Exemplo Perl

```
#!/usr/bin/perl
use BACO::BACODB;

my $baco = new BACO::BACODB();
$baco->servidor("localhost");   ## Definir o servidor
## Nota: para estabelecer a ligacao, atencao as permissões no mysqld!
$baco->init() || die "Nao foi possivel estabelecer a ligacao ao sevidor BACO";
$baco->cache(1);  ## Se quisermos ter efeito de "caching" no mysqld.

## Uma pesquisa concreta por frases retorna uma lista de tuplos
@frases = $baco->frasesMatch('+"alvares cabral" +brasil',1,15);
for (@frases) {
    ## Cada tuplo é uma hash com campos que dependem  do metodo e tabela
    %tuplo = %{$_};
     print "Frase: ". $tuplo{frase} ."\n"
}
```

Exemplo de um método data-driven usando o BACO

- Expansão de um conjunto de co-hipónimos
  - descoberta de "elementos semelhantes"a um conjunto de exemplos dados pelo utilizador
- Observação: elementos semelhantes ocorrem em contextos léxicais semelhantes
  - "a compota de morango/laranja":
    - "a compota de X" --> X? Fruto? Vegetal?
  - "o gelado de morango/baunilha":
    - "o gelado de X" --> X? Fruto? Alimento?

Usando a tabela de 4-gramas

- Para os elementos do conjunto $S = \{s_1, s_2, ..s_n\}$
  - Procurar contextos de 3 palavras: c_1, c_2, c_3, s_i
- Para contextos "representativos" $c_{i1}, c_{i2}, c_{i3}, X$:
  - Procurar elementos X_i que ocorram no mesmo contexto
- Contextos "representativos" co-ocorrem com:
  - L sementes: garante especificidade
  - menos de M_{max} elementos do léxico (250): impede sobre-generalização
- O algoritmo "aprende" contextos "representativos".

Resultados

#	Conjunto inicial	L	# C	Resultado
1	amarelo, vermelho, azul	3	4 4	verde (26), branco (22), preto (19), cinza (14), castanho (14), ... violeta (6), prata (5), *escuro* (4), dourado (4), *fe* (4), ... *pele* (4), *cores* (3), ... *liso* (2), *carvalho* (2), marrom (2), ... *terra* (2), *iluminado* (2), *54* (2), *brasil* (2), *pobre* (2)
2	granito, mármore, basalto	3	3	*betão* (2), *vidro* (2), *papel* (2), *pedra* (2), *madeira* (2), *material* (2)
3	whiskey, rum, gin	2	6	vodka (4), vinho (3), tequila (3), porto (3), cerveja (2), licor (2), sumo (2), coca-cola (2), verdelho (2), whiskie (2), tinto (2), aguardente (2), conhaque (2), *jack* (2), *neoplast* (2), uisque (2), água (2), *coca* (2), *plástico* (2), champanhe (2), cachaça (2), champagne (2)
4	porto, braga, aveiro	3	2 1 0	coimbra (141), lisboa (137), *vila* (126), *castelo* (115), leiria (110), viseu (110),... almada (51), guimarães (49),... *cidade* (5) ... régua (2), *avaliação* (2), *recrutamento* (2), *municípios* (2), *editorial* (2), gorazde (2), *gás* (2), *coliseu* (2), alvor (2), inhambane (2)

Algumas ideias para o futuro

- Marcar o BACO sintacticamente / semanticamente:
  - Já feito para cerca de 10% com REM SIEMÊS
- Tornar mais automático e eficiente a criação de BACO's
  - Como fazer a contabilização das frequências com mais eficiência?
  - Como automatizar o processo para novas e maiores colecções?
  - Como distribuir a base para outras pessoas (~70Gb…)?
- Como distribuir do esforço de pesquisa?
  - Vários servidores BACO em paralelo?
  - Uma rede de servidores BACO espalhada por vários investigadores?
  - Uma máquina tipo Google usando micro-servidores, screen-savers e tempo de CPU em vazio (interessados? falem comigo depois…)
- API's noutras linguagens ou BACO como Web-Service?

Conclusões

- BACO: uma base de texto de 1000 M de palavras gerada a partir do WPT03
- Usa MySQL para armazenamento e indexação dos dados, e para processamento de pesquisa
- API Perl disponível (peçam-ma por mail)
  - Permite a exploração simples de métodos "data-driven"
- A base pode ser descarregada:
  - ou podem vir ao Porto buscá-la / copiá-la :)
- Bastantes possibilidades de desenvolvimento futuro

Obrigado

Luis Sarmento -> las@fe.up.pt