TEMU KEMBALI INFORMASI

JULIO ADISANTOSO
Departemen Ilmu Komputer IPB

Pertemuan 4

TOLERANCE RETRIEVAL

Tugas

- Pelajari Extended Boolean
  - Mengapa dikembangkan model ini?
  - Bagaimana prinsip dasar model ini?
  - Bagaimana teknis komputasi model ini?
- Kerjakan soal pada Manning *et al* (2008) nomor 2.9, 6.8, 6.9, 6.10, 6.11, 6.19

Extended Boolean

- Model Boolean yang baku tidak menghasilkan pemeringkatan dokumen
- Operator Boolean dalam model Boolean yang baku terlalu *strict*

Maka ...

- Dikembangkan model Boolean yang memungkinkan memberi pemeringkatan dokumen ... extended Boolean
- Ada 3 model:
  1. Mixed Min and Max (MMM) Model
  2. Paice Model
  3. P-norm Model

Extended Boolean

Diketahui matrik term-document:

$$\begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{i1} & w_{i2} & \cdots & w_{in} \\ \vdots & \vdots & \ddots & \vdots \\ w_{t1} & w_{t2} & \cdots & w_{tn} \end{pmatrix}$$

sedangkan w_{ij} adalah bobot term i pada dokumen j, untuk $i = 1, 2, \ldots, t$ dan $j = 1, 2, \ldots, n$

MMM Model

- Misalkan terdapat query:

$$\begin{aligned} Q_{OR} &= (T_1 \; or \; T_2 \; or \; \ldots \; or \; T_t) \\ Q_{AND} &= (T_1 \; and \; T_2 \; and \; \ldots \; and \; T_t) \end{aligned}$$

- Maka ukuran kemiripan dokumen j dengan query adalah:

$$\begin{aligned} sim(Q_{OR}, j) &= \alpha \times \max(w_{1j}, w_{2j}, \ldots, w_{tj}) + \\ &\quad (1-\alpha) \times \min(w_{1j}, w_{2j}, \ldots, w_{tj}) \\ sim(Q_{AND}, j) &= \beta \times \min(w_{1j}, w_{2j}, \ldots, w_{tj}) + \\ &\quad (1-\beta) \times \max(w_{1j}, w_{2j}, \ldots, w_{tj}) \end{aligned}$$

- Hasil percobaan Lee & Fox (1988): nilai $\alpha > 0.2$ dan $\beta \in [0.5, 0.8]$

Paice Model

- Ukuran kemiripan dokumen j dengan query adalah:

$$sim(Q,j) = \frac{\sum_{i=1}^{n} r^{i-1} d_i}{\sum_{i=1}^{n} r^{i-1}}$$

  sedangkan d_i adalah urutan menurun untuk query *OR* dan menaik untuk *AND*.
- Percobaan Lee & Fox (1988) menghasilkan nilai r yang optimum adalah 1.0 untuk query *AND*, dan 0.7 untuk query *OR*.

P-norm Model

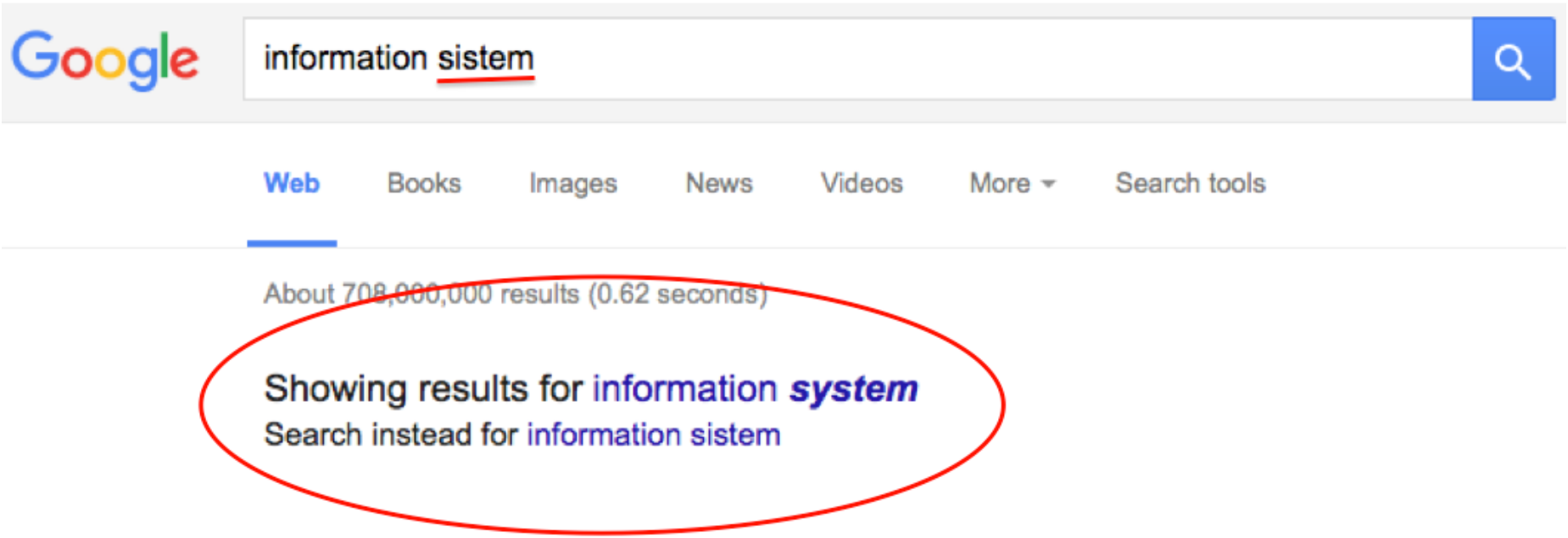
- Ukuran kemiripan dokumen j dengan query adalah:

$$
\begin{aligned}
sim(Q_{OR}, j) &= \sqrt[p]{\frac{w_{1j}^p + w_{2j}^p + \cdots + w_{tj}^p}{t}} \\
sim(Q_{AND}, j) &= 1 - \sqrt[p]{\frac{(1-w_{1j})^p + (1-w_{2j})^p + \cdots + (1-w_{tj})^p}{t}}
\end{aligned}
$$

- Misal query $Q = (T_1 \; and \; T_2) \; or \; T3$, maka untuk $p = 2$, ukuran kemiripan Q dengan dokumen j adalah:

$$
sim(Q, j) = \sqrt{\frac{\left(1 - \sqrt{\frac{(1-w_{1j})^2 + (1-w_{2j})^2}{2}}\right)^2 + w_{3j}}{2}}
$$

Tolerance Retrieval



- Bagaimana metode matching antara term pada query dengan term pada inverted index (dictionaries)? – terkait ekspansi query?

Phonetic Problem

Metode

Beberapa metode Tolerance Retrieval:

- WordNet and Thesaurus
- Wildcard queries
- Spelling correction
  1. Jarak edit (edit distance)
  2. *n*-gram
- Phonetic

Sangat situasional

Wildcard Queries

Mengapa Wildcard Queries?:

- User tidak tahu pasti spelling dari suatu term. Misalnya Sydney vs Sidney, University Stuttgart vs Universitet Stuttgard, etc.
- User khawatir pada sistem yang memiliki ragam spelling. Misalnya colour vs color, center vs centre, etc.
- User ingin mencari beberapa varian dari term. Misalnya pertanian, bertani, petani, dsb.

Alternatif:

- Menggunakan wildcard queries, misalnya s*dney, universi* stuttg*, colo*r, etc.
- Menggunakan aturan RE. Misalnya [∧a][mad$],

Spelling Correction

Bedanya dengan Wildcard Queries?

- User mengetik query, namun mungkin ada kesalahan. Misalnya `Information Sistem`
- Menggunakan teknik ekspansi query, artinya term pada query disempurnakan oleh sistem.

Metode:

- Memberi alternatif query pada user berupa term yang mirip.
- Ukuran kemiripan ditentukan dengan banyak metode, antara lain jarak edit dan n-gram.

n-gram

Berbasis karakter:

- Merupakan potongan n karakter dari suatu string (kata/term).
- Contoh: kata ”komputer”

Nama	**n-gram karakter**
uni-gram	k,o,m,p,u,t,e,r
bi-gram	_k,ko,om,mp,pu,ut,te,er,r_
tri-gram	__k,_ko,kom,omp,mpu,put,ute,ter,er_,r__
dst.	

n-gram

Berbasis kata:

- Merupakan potongan n kata dari suatu kalimat.
- Contoh: ”musim kemarau, bogor mulai kering”

Nama	**n-gram karakter**
uni-gram	musim, kemarau, bogor, mulai, kering
bi-gram	musim kemarau, kemarau bogor, bogor mulai, mulai kering
dst.	

Dapat dikembangkan juga n-gram berbasis kalimat ... antara lain digunakan dalam QAS (Question Answering System)

Ukuran Kemiripan

- Misalkan S_1 dan S_2 adalah himpunan n-gram dari string 1 dan 2.
- Ukuran kemiripan dapat diukur dengan berbagai cara, antara lain menggunakan koefisien Jaccard, yaitu:

$$J(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$$

- Contoh (bi-gram) kata achmad dan ahmad
  - $S_{achmad} = \{\_a, ac, ch, hm, ma, ad, d\_\}$
  - $S_{ahmad} = \{\_a, ah, hm, ma, ad, d\_\}$
  - Maka:

$$J(S_{achmad}, S_{ahmad}) = \frac{5}{8} = 0.625$$

Jarak Edit

- Pengukuran berdasarkan kesalahan ketik: (1) penyisipan, (2) penghilangan, (3) penukaran karakter
- Jarak edit (edit distance) sering dikenal sebagai Levenshtein distance.
- Misalkan S_1 dan S_2 adalah himpunan karakter dari string 1 dan 2.
- Jarak edit S_1 dan S_2 menggunakan dynamic programming algorithm sbb:
  - Jika $S_1[i] = S_2[j]$, maka:
    $m[i,j] = min(m[i-1,j-1], m[i-1,j]+1, m[i,j+1]+1)$
  - Jika $S_1[i] \neq S_2[j]$, maka:
    $m[i,j] = min(m[i-1,j-1]+1, m[i-1,j]+1, m[i,j+1]+1)$

Algoritme Jarak Edit

1: $m[i,j] \leftarrow 0$
2: **for** $i \leftarrow 1$ *to* $|S_1|$ **do**
3: $\quad m[i,0] \leftarrow i$
4: **end for**
5: **for** $j \leftarrow 1$ *to* $|S_2|$ **do**
6: $\quad m[0,j] \leftarrow j$
7: **end for**
8: **for** $i \leftarrow 1$ *to* $|S_1|$ **do**
9: $\quad$ **for** $j \leftarrow 1$ *to* $|S_2|$ **do**
10: $\quad\quad m[i,j] = min(m[i-1,j-1]+X, m[i-1,j]+1, m[i,j+1]+1)$
11: $\quad$ **end for**
12: **end for**
13: **return** $m[|S_1|,|S_2|]$

Contoh

$S_1 = \{f, a, s, t\}$ dan $S_1 = \{c, a, t, s\}$

		f		a		s		t	
	0	1	1	2	2	3	3	4	4
c	1 1	*1* *2*	*2* *1*	2 2	3 2	3 3	4 3	4 4	5 4
a	2 2	2 3	2 2	*1* *3*	*3* *1*	*3* *2*	*4* *2*	4 3	5 3
t	3 3	3 4	3 3	3 4	2 2	*2* *3*	*3* *2*	*2* *3*	*4* *2*
s	4 4	4 5	4 4	4 5	3 3	2 4	3 2	*3* *3*	*3* *3*

Jarak edit S_1 dan S_2 adalah 3.

Phonetic Correction

- Pengukuran berdasarkan kemiripan bunyi pengucapan.
- Menggunakan algoritme Soundex, mengubah string menjadi 4 karakter dengan aturan:
  1. Ambil karakter pertama
  2. Ubah setiap karakter menjadi angka dengan aturan sbb:
     - A, E, I, O, U, H, W, Y $\rightarrow$ 0.
     - B, F, P, V $\rightarrow$ 1.
     - C, G, J, K, Q, S, X, Z $\rightarrow$ 2.
     - D,T $\rightarrow$ 3.
     - L $\rightarrow$ 4.
     - M, N $\rightarrow$ 5.
     - R $\rightarrow$ 6.
  3. Ubah angka sama bersebelahan menjadi satu angka
  4. Hapus semua angka '0'

Implementasi dalam IR