Information Retrieval dengan Menggunakan Metode Latent Semantik Indexing (LSI) pada Proses Searching dan Klasifikasi Buku: Studi Kasus Perpustakaan STMIK Bumigora

KARTARINA[1], HARTARTO JUNAEDI[2]
[1]STMIK Bumigora Mataram
[2]Institut Sains Teknologi Terapan Surabaya
[1]augustin.kartarina@gmail.com, [2]hartarto.j@gmail.com

Abstract

A library in STMIK Bumigora as support unit in educational institution. It is provided to support teaching learning process. Enhencing library function optimally will improve the quality of education. The problems of STMIK Institution library are that in book classification by librarian in STMIK Bumigora is done manually, it means the books added and labelled to the shelves based on the information title as administrator knowledge, that sometimes books with the same tittle numbers are placed in different places. As a result, the visitor have problems when they search the book, sometimes book and the place is not as expected, for example the title and contents does not match. Therefore the research obataining IR of books or documents indexing in library can be derived by using *Laten Semantic Indexing (LSI)*method in searching process and clasifiying process. LSI represents terms of the document in the form of vectors which arrange in *Term Document Matrix.* The exellence of LSI is to deal with the polysemy and synonym words which usually found in documents. It works by counting the rapport among terms in one document with terms in other documents. It can be done by decomposing *term document matrix* as *inverted file* of LSI in order to get the rapport of those terms. So the relevant document though it does not contain any term of a query, stillit can be generated. Decomposing matrix can use *Singular Value Decomposition* (SVD) method. This research discusses IR of library system by appliying LSI method and using SVD as the decomposition method classification. Experiment has been conducted toward the document of book summary contained on back cover. The preliminary data is in the form of scanned textbooks in Bahasa, then it is converted into text document files (.txt file). This proposed method is expected to assist the process of classifying and searching books in the library. LSI can provide appropriate information to its visitors to find books that is relevant to the query inputs.

Keywords*: Information Retrieval, Singular Value Decompostion, Latent Semantic Indexing, term document matrix.*

I. PENDAHULUAN

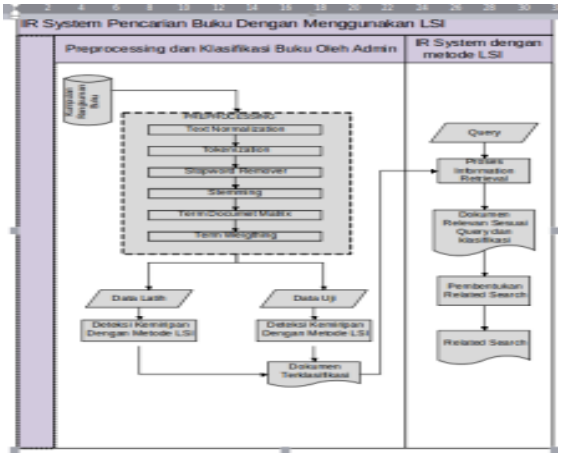
Perpustakaan dalam sebuah lembaga pendidikan merupakan salah satu fasilitas yang disediakan sebagai pendukung dan penunjang proses kegiatan belajar mengajar. Keberadaan Perpustakaan sangat membantu untuk menambah atau meningkatkan pengetahuan dan wawasan. Meningkatkan fungsi perpustakaan secara maksimal diharapakan juga akan memberikan kualitas pendidikan yang optimal,tidak hanya meningkatkan kualitas pendidikan tetapi juga akan menambah kualitas perpustakaan itu sendiri. *Information Retrieval* merupakan Studi tentang system pengindeksan, pencarian, dan mengingat data, khususnya teks atau bentuk tidak terstruktur lainnya dimana pengindeksan dan pengambilan informasi berasal dari sumber informasi heterogen [1].

Adanya permasalahan dalam klasifikasi/ memasukkan buku kedalam kelompoknya dan masalah pencarian buku oleh pengunjung yang memperoleh buku kadang buku yang diperoleh dari tempatnya tidak sesuai dengan yang diharapkan misalnya judul dan isi tidak sesuai. Dari latar belakang diatas mendorong penulis untuk melakukan penelitian mengenai *Information Retrieval*. Untuk mendapatkan *Information Retrieval* dalam indeks buku / dokumen di perpustakaan mencoba menggunakan metode*Laten Semantic Indexing* (LSI) yaitu dengan menerapkan model *Singular Value Decomposition* (SVD) [2].

II. Metode Penelitian

IR Sistem Dengan LSI

2.1 Arsitektur Sistem



Gambar 1
Arsitektur Sistem

2.2 Preprocessing [5]

Proses diawali dengan menentukan / membagi data menjadi data latih dan data uji. adapun langkah-langkah dalam preprocessing adalah sebagai berikut:

- *Text Normalization*
  Merupakan proses mengubah teks menjadi huruf kecil (*lowercase*) dan menghapus simbol dan karakter *alphabet*.
- *Tokenizing*
  Merupakan proses untuk pemisahan kalimat menjadi kata per kata atau potongan tunggal.
- *Stopword*
  Merupakan proses untuk menghilangkan kata-kata yang sering muncul, namun tidak mengandung arti,*Stopword* dikatakan tidak memiliki arti karena tidak memiliki keterkaitan dengan topik tertentu. Untuk mengetahui suatu kata merupakan suatu *stopword* atau bukan adalah menggunakan kamus *stopword* yang sudah ditentukan ditentukan.
- *Stemming*
  merupakan proses untuk mengambil kata dasar dengan cara memotong imbuhan pada kata tersebut, baik itu berupa awalan, akhiran, maupun sisipan. Pada penelitian ini menggunakan model stemming Sastrawi. Sastrawi merupakan library PHP sederhana yang dapat digunakan dalam proses stemming, yaitu mengubah kata berimbuhan menjadi kata dasar dalam bahasa Indonesia. Contohnya:
  - menahan => tahan
  - berbalas-balasan => balas

  Proses stemming dengan sastrawi dapat membantu menemukan dokumen yang sedang dicari yaitu dengan menanggalkan imbuhan-imbuhan hingga hanya menyisakan kata dasar. Dalam penelitian ini menggunakan library PHP untuk proses stemming dikarenakan mudah diintegrasikan dengan framework / package lainnya selain sederhana dan mudah digunakan. Proses stemming oleh Sastrawi sangat bergantung pada kamus kata dasar.
- *Term Document Matrix* (TDM)
  Term Document Matrix (TDM), merupakan proses untuk membentuk matrik yang memuat nilai kemunculan (*frekuensi*) masing-masing kata di setiap dokumen. TDM merupakan bentuk *inverted file* atau file index dari *Vector Space Model* (VSM) yang juga digunakan dalam LSI. TDM merupakan sebuah matrik yang terdiri dari seluruh term index sebagai baris matrik dan seluruh dokumen sebagai kolom matrik. Elemen dari *term* dokumen matrik merupakan frekuensi dari kemunculan

term term yang bersangkutan pada dokumen yang ditunjuk. Proses perhitungan VSM melalui tahapan perhitungan *term frequency (tf)* menggunakan persamaan berikut ini:

$tf = tf_{ij}$ ..... **[2]**

Dengan *tf* adalah *term frequency*, dan tf_{ij} adalah banyaknya kemunculan term *ti* dalam dokumen *dj*, *term frequncy (tf)* dihitung dengan menghitung banyaknya kemunculan term *ti* dalam dokumen *dj*.

- *Term Weighting*
  merupakan proses pembobotan otomatis berdasarkan frekuensi kemunculan suatu kata dalam sebuah dokumen (*term frequency*) dan frekuensi kemunculan dalam kumpulan dokumen (*inverse document frequency*), yang biasa disebut pembobotan TF-IDF. Dengan menggunakan persamaan:
  tf-idf(t) = tf(*t*,*d*) x idf(*t*) ... [2]
- *Similarity*
  Merupakanpengukuran kemiripan teks yang paling poopuler adalah menggunakan *Cosine Similarity*. Pada Metode ini mengukur nilai *cosinus* sudut antara dua vektor. *Cosinus* dari dua vektor dapat diturunkan dengan menggunakan rumus *Euclidean dot product.* **[3]**
  a.b =||a||.||b|| cos θ sehingga

$$\cos\theta = \frac{\vec{a}\vec{b}}{||\mathbf{a}||.||\mathbf{b}||}$$

Dimana $\vec{a}$ dapat mewakili dari query $\vec{q}$ dan $\vec{b}$ mewakili dari dokumen $\vec{d}$. Sehingga persamaan ini dapat dituliskan sebagai :

$$\mathbf{Sim(q,d_j)} = \frac{\vec{q}\,\vec{dj}}{||\mathbf{q}||.||\mathbf{dj}||} = \frac{\sum_{i=1}^{t} W_{iq}.W_{ij}}{\sqrt{\sum_{i=1}^{t}(W_{iq})^2 * \sum_{i=1}^{t}(W_{ij})^2}}$$

Similaritas antara query dan dokumen atau Sim (q,d_j) berbanding lurus terhadap jumlah bobot query (q) dikali bobot dokumen (d_j) dan berbanding terbalik terhadap akar jumlah kuadrat q (|q|) dikali akar jumlah kuadrat dokumen 1 atau menghasilkan bobot dokumen yang lebih besar dibandingkan dengan nilai yang dihasilkan dari perhitungan *inner product*.

Sebagai contoh terdapat dua dokumen $D_1 = 2T_1 + 3T_2 + 5T_3$ dan $D_2 = 3T_1 + 7T_2 + T_3$dan $Q = 0T_1 + 0T_2 + 2T_3$ seperti yang terlihat pada gambar 2 berikut ini.

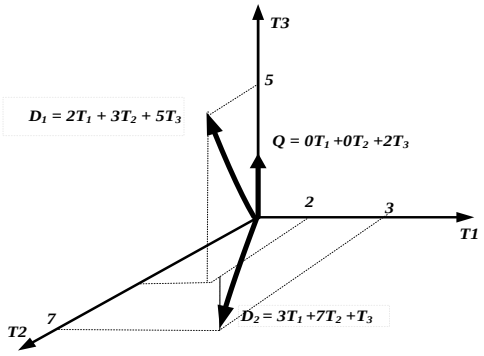
	T_1	T_2	T_3	$T_{...}$	T_t
D_1	W_{11}	W_{21}	W_{31}	...	T_{t1}
D_2	W_{12}	W_{22}	W_{32}	...	T_{t2}
D_3	W_{13}	W_{23}	W_{33}	...	T_{t3}
$D_{..}$	...	...	...	...	...
D_n	W_{1n}	W_{2n}	W_{3n}	...	T_{tn}

Gambar 2. Matrik Term Dokumen

$$\text{Similarity}(D_1, Q) = \frac{((2.0)+(3.0)+(5.2))}{\sqrt{(4+9+25)*(0+0+4)}} = \frac{10}{\sqrt{38*4}} = 0.81$$

$$\text{Similarity}(D_2, Q) = \frac{((3.0)+(7.0)+(1.2))}{\sqrt{(9+49+1)*(0+0+4)}} = \frac{2}{\sqrt{236*4}} = 0.065$$

Contoh ini menunjukkan bahwa dari perhitungan *cosinus*, dokumen D_1 lebih mirip dengan query jika dibandingkan dengan dokumen D_2. faktanya adalah, sudut antara D_1 dan Q_1 lebih kecil, jika dibandingkan dengan sudut antara D_2 dan Q_1
seperti terlihat pada gambar 3 berikut ini



Gambar 3.Kemiripan Diantara D1, D2 dan Q1

2.3 Klasifikasi dan Pencarian Buku

Dokumen berupa ringkasan buku yang telah diinputkan pada sistem di bagi menjadi 2 jenis, yaitu data latih dan data uji. Data latih merupakan kumpulan dokumen yang diinputkan sebagai acuan untuk kemudian digunakan sebagai pembanding oleh data uji (dokumen / buku baru). Data latih dengan nama klasifikasi buku tersebut. Pengisian data buku ini bertujuan untuk membuat klasifikasi-klasifikasi dokumen (pembentukan kelas buku) yang nantinya akan bermanfaat bagi petugas perpustakaan dalam memasukkan data buku baru/ Data uji. Setelah data latih dimasukkan semua kedlm sistem dan dilakukan proses *generat* perhitungan bobot dan perhitungan vektornya hasilnya disimpan dalam *collection data*.

Untuk data uji (proses input buku baru) maka petugas dapat melakukan testing dengan menerapkan model klasifikasi yang diperoleh dari data latih. Penentuan jenis kelas terhadap data uji secara otomatis akan membandingkan data tersebut dengan data latih yang sudah tersimpan dalam *collection data.*

Pada proses pencarian, Pengguna/pengunjung memasukkan q*uery* (kata). LSI digunakan untuk mendeteksi kemiripan kata/query yang diinput oleh user dengan kata/term yang terdapat pada koleksi dokumen. Menggunakan model SVD pada TDM dokumen yang telah dihitung bobotnya dengan menggunakan model TF-IDF. Matriks kata-dokumen *A* berukuran *m x n*, selanjutnya dikenakan dekomposisi SVD. Hasil SVD menghasilkan tiga buah matrik yaitu *U*, *S*, dan *V*. dimana matriks *A* dapat ditulis dengan:

$$A = U\sum V^T \text{[3]}$$

dimana :

A = Term Document Matrik

U = Matrik dimana kolomnya adalah eigenvektor dari matrik A^TA

S = Atau $\sum$matrik yang elemen diagonalnya adalah nilai singular dari A

V^t = berukuran *k x n* dan Transpose dari *V*

m = Adalah jumlah baris pada *A*

n = Adalah kolom pada *A*

k = Rank pada *A*

Setelah nilai SVD diperoleh maka dilakukan proses *reduce* dimension terhadap matrik *U, S,* dan *V*. Hasil reduksi ini digunakan untuk menghitung nilai vektor *query* dan vektor dokumen dalam dimensi SVD sehingga $Q = q^T Ur \sum r^{-1}$.

Setelah nilai vektor dokumen dan query yang baru dihitung, kemudian menghitung *similarity cosine* dari dokumen dan query tersebut agar didapat ranking *score*-nya.

2.4 Pengukuran Relevansi [4]

Pada sistem, dokumen yang ditampilkan memiliki jumlah yang sama dengan jumlah dokumen relevan yang harus ditampilkan, namun pada kenyataannya sistem juga akan menampilkan dokumen yang tidak relevan dengan permintaan pengguna.Model pengukuran yang digunakan pada penelitian ini adalah *Recall*, *Precison* dan *F-Measure*.Precision dan Recall digunakan bersama-sama yang digabungkan dalam F-Measure.

Recall adalah mengukur perbandingan dokumen relevan yang diterima (*true positive*) dengan jumlah seluruh dokumen yang diharapkan diterima (*true positive and false negatives*). Recall menunjukan jumlah dokumen yang relevan. Perhitungan Recall dinyatakkan pada rumus berikut :

$$Recall = \frac{|R \cap A|}{|R|}$$

Precision adalah mengukur perbandingan dokumen relevan yang ditemukan (*true positive*) dengan jumlah sejumlah dokumen yang relevan (*true positive dan false positif)* Perhitungan precision dinyatakan pada rumus.:

$$Precision = \frac{|R \cap A|}{|A|}$$

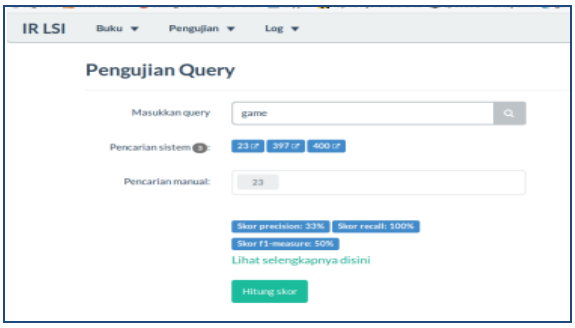
Terkadang sebuah sistem memiliki nilai recall tinggi tetapi memiliki nilai precision rendah dan sebaliknya. Jadi, sebaiknya pengukuran tidak hanya menggunakan salah satu precision atau recall saja. Oleh karena itu, Precision dan Recall digunakan bersama-sama yang digabungkan dalam F-Measure. Perhitungan f-measure dinyatakan pada rumus berikut ini

$$f\text{-}measure = \frac{2 \ Precission \ x \ Recall}{Precission + Recall}$$

III. Pembahasan dan Hasil

Information Retrieval (IR) System pencarian buku dilakukan untuk mengetahui Relevansi *query* dengan dokumen. Pada tahap ini disiapkan data berupa *collection data* ringkasan buku yang diperoleh dari cover belakang buku yang telah di *scan* dan dikonversikan ke dalam file teks, dan di edit file teksnya secara manual sehingga dapat digunakan uji coba penelitian.

Pada pengujian dilakukan dengan cara memasukkan *query* yang akan diuji oleh user yaitu uji coba untuk menampilkan dokumen-dokumen yang memiliki kecocokan dan nlilai terdekat similarity nya. Seperti pada gambar berikut ini menampilkan contoh proses uj query



Gambar 3. Pengujian Query

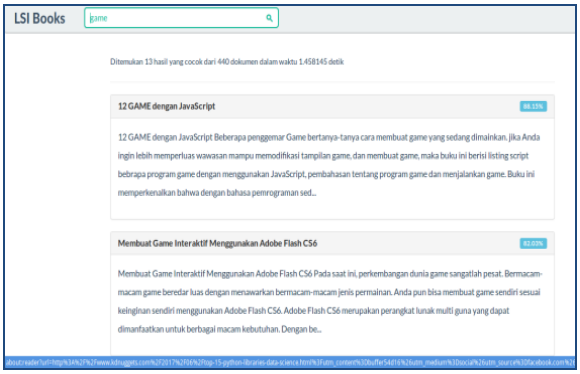
Pada gambar 3 merupakan inputan uji query system terhadap manual (hasil pencarian oleh operator perpustakaan). Setelah mendapatkan kata kunci maka dapat diproses kembali dengan algoritma TF-IDF untuk mendapatkan nilai bobot dokumen, lalu kembali dilakukan perhitungan dengan melakukan perhitungan SVD.Setelah semua proses selesai maka akan muncul nilai bobot dokumen dari nilai terbesar sampai terkecil, dokumen dengan nilai bobot terbesar adalah dokumen yangmemiliki tingkat kemiripan tertinngi. Hasil dari uji coba tersebut dapat dilihat pada tabel 1 dibawah ini:

Tabel 1. Hasil Uji Query

No	Kode Buku	Judul	Nilai Kemiripan	Klasifikasi
1	B00823	12 GAME dengan JavaScript	0.8814962506	Game
2	B00397	Membuat Game Interaktif Menggunakan Adobe Flash CS6	0.8202838898	Game
3	B00400	Membuat Game RPG dengan RPG Maker	0.3970266879	Game
4	B00194	belajar cepat	0.1122898236	Ilmu Praktis
5	B00435	Pemrograman Java	0.1045477912	Belun dites
6	B00821	PEMROGRAMAN JAVA	0.1033073142	Software Enginering
7	B00174	mudah membuat animasi adobe flash CS5	0.0885343924	Animasi
8	B00388	Artificial Intelligence	0.0809969088	Ilmu Komputer
9	B00899	KAMUS ++ JARINGAN KOMPUTER	0.0733341947	Jaringan
10	B00383	Konsep Kecerdasan Buatan	0.0658712760	Ilmu Komputer
11	B00180	panduan praktis menggunakan 3D Studio Max9	0.0595319346	Animasi
12	B00015	Pemrograman Android dengan APP Inventor	0.0572865829	Pemrograman Mobile

Tabel 1 merupakan ringkasan log file hasil uji query pada gambar 3. Untuk mendapatkan koefesien similarity adalah dengan menggunakan cosinus antara query dengan tiap dokumen. Dari hasil perhitungan koefesian similarity dapat diperoleh dokumen-dokumen yang isinya memiliki tingkat kemiripan dengan kata kunci. Seperti uji coba "Game" pada tabel 1yang menghasilkan 13 dokumen yang cocok dengan sistem memiliki nilai similaritas tertinggi yaitu pada dokumen kode buku 23 sebesar 0.8814962506 atau sekitar 88% dan buku masuk kedalam klasifikasi Game.

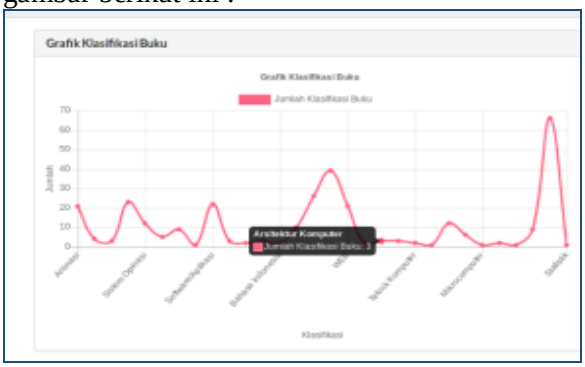
Pengujian query adalah pembuktian kesamaan hasil perolehan pengunjung dalam mencari buku di form pencarian buku seperti pada gambar 4 berikut ini.



Gambar 4. Hasil Pencarian Buku

Proses *searching* dan klasifikasi pada form ini menampilkan hasil IR menggunakan LSI yaitu pada saat memasukkan inputan (query / dokumen yang akan dicari, sistem akan mengambil bobot setiap dokumen dan query inputan kemudian menghitung dekomposisi matriks USV^t pebentukan koordinat vektor dokumen yang baru dan membentuk koordinat vektor query yang baru kemudian hitung nilai similaritas, Dokumen yang berada pada posisi teratas pada saat proses pencarian dokumen adalah dokumen dengan nilai cosine similarity tertinggi hasil dari penghitungan TF-IDF untuk tiap term pada query.hasil akhir adalah kelompokkan dokumen yang ditemukan sesuai klasifikasi (dokumen yang telah ditraining).

Grafik klasifikasi buku dapat digambarkan pada gambar berikut ini :



Gambar 5. Klasifikasi Buku

Klasifikasi dokumen adalah dengan mengelompokkan dokumen yang ditemukan sesuai klasifikasi (dokumen yang telah ditraining) kemudian ambil dokumen ≤ k yang telah ditentukan kemudian jumlahkan jarak dokumen sebanyak k dokumen yang telah diambil dari masing-masing klasifikasi.

dalam kaitannya dengan performa, pengukuran kinerja pada sistem/metode IR ini dengan menggunakan parameter *Recall, Precision* dan *F-measure.*Terkadang sebuah sistem memiliki nilai recall tinggi tetapi memiliki nilai precision rendah dan sebaliknya hal ini dikarenakan pengambilan data relevan oleh sistem tidak seimbang dengan yang manual atau sebaliknya. Seperti pada gambar 3 nilai precision 30%, dan Recall 100% hal ini dikarenakan hasil pencarian pada sistem mendapatkan dokumen relevan lebih banyak dibandingkan dengan yang manua l, maka nilai f-measure adalah 50% didapat dari penggabungan recall100% dan Precision 33%.

IV. Kesimpulan

Dari pembahasan pada penelitian ini kesimpulan yang didapat sebagai berikut:

a. Sistem information retrieval melakukan penentuan kerelevanan dokumen berdasarkan term yang terdapat pada query dan dokumen.
b. Metode LSI dan *cosine similarity* digunakan untuk mendeteksi kemiripan kata-kata dalam dokumen.
c. Memanfaatkan metode TF-IDF sebagai model untuk pembobotan *query relevance.*
d. Percision tinggi apabila tingkat informasi yang diberikan oleh operator lebih banyak dengan informasi yang diberikan oleh sistem, sedangkan recall akan tinggi apabila tingkat keberhasilan sistem dalam memberikan informasi relevan lebih banyak dalam menemukan kembali sebuah informasi dibanding dengan manual.
e. F-Measure merupakan nilai gabungan antara Precision dan Recall.

Daftar Pustaka

[1] Eko Prasetyo.2012.***"Data Mining Konsep dan Aplikasi Menggunakan Matlab"***. Yogyakarta: ANDI.

[2] Jianxiong Yang, Junzo Watada, **"*Decomposition of Term-Document Matrix Representasion for Clustering Analysis"***, (IEEE International Conference on Fuzzy Systems)2011, p978

[3] Michael W. Berry.Susan T. Dumais.Gavin W. O'Brien.1995. Society for Industrial and Applied Mathematics.SIAM review.***"Using Linear Algebra for Intelligent Information Retrieval".***Journal Jstor. PP. 573-995.

[4] Riyanarto Sarno. Yeni Anistyasari. Rahimi Fitri. 2012.***"Semantic Search"***. Yogyakarta: ANDI.

[5] Wei Song.Soon Cheol Park. 2009. Division of Electronics and Information Engineering, Chonbuk National University, Jeonju, 561756, Republic of Korea .***"Genetic algorithm for text clustering based on latent semantic indexing".*** Journal of Machine Learning Research. Elseiver.Vol.57. Issue 11-12Yates, RB. ***"Modern Information Retrieval"***. Boston, USA. Addison Wesley-Pearson International Edition 1999