

***SEARCH ENGINE* PADA DOKUMEN *ONLINE SHOP*
MENGUNAKAN METODE BERBASIS KONSEP**

RIZKI ADI UTOMO



**DEPARTEMEN ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
INSTITUT PERTANIAN BOGOR
BOGOR
2016**

PERNYATAAN MENGENAI SKRIPSI DAN SUMBER INFORMASI SERTA PELIMPAHAN HAK CIPTA

Dengan ini saya menyatakan bahwa skripsi berjudul *Search Engine* pada Dokumen *Online Shop* Menggunakan Metode Berbasis Konsep adalah benar karya saya dengan arahan dari komisi pembimbing dan belum diajukan dalam bentuk apa pun kepada perguruan tinggi mana pun. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan maupun tidak diterbitkan dari penulis lain telah disebutkan dalam teks dan dicantumkan dalam Daftar Pustaka di bagian akhir disertasi ini.

Dengan ini saya melimpahkan hak cipta dari karya tulis saya kepada Institut Pertanian Bogor.

Bogor, Juni 2016

Rizki Adi Utomo
NIM G64120094

ABSTRAK

RIZKI ADI UTOMO. *Search engine* pada Dokumen *Online shop* Menggunakan Metode Berbasis Konsep. Dibimbing oleh JULIO ADISANTOSO dan YENI HERDIYENI.

Ecommerce atau *online shop* telah banyak menawarkan kemudahan, salah satunya berupa fitur mesin pencari. Mesin pencari yang digunakan *online shop* pada saat ini masih menggunakan sistem pencarian berbasis *keyword* dan menimbulkan beberapa masalah seperti efek kata polisemi dan homonim. Hal ini menyebabkan kinerja mesin pencari menjadi tidak optimal. Oleh karena itu, diperlukan model mesin pencari berbasis konsep untuk mengatasi masalah tersebut. Teknik temu kembali berbasis konsep mengurutkan dokumen berdasarkan kombinasi kemiripan kata dan konsep, sehingga mesin pencari dapat melakukan interpretasi lebih jauh terhadap kueri pengguna. Struktur konsep yang baik diperlukan untuk meningkatkan kinerja pencarian berbasis konsep, terutama situs *online shop* yang memiliki banyak data produk. Cara terbaik untuk mendapatkan struktur konsep terbaik adalah dengan melakukan percobaan.

Kata Kunci: *online shop*, mesin pencari, berbasis konsep, struktur konsep.

ABSTRACT

RIZKI ADI UTOMO. Search engine with Online shop Document using Concept-Based Method. Supervised by JULIO ADISANTOSO and YENI HERDIYENI.

E-commerce has provide so many advantage to user today, one of them is search engine. Today's online shop search engines are still following the paradigm of keyword-based search and contains some problem like polisemy and homonym. The problems makes search engine's performance give not an optimal result. Therefore, we need search engine model concept-based to solve the problems. Information retrieval concept based rank the document based on similarity between term and concept, so the search engine have ability to accomplish more idea of query. The best concept structure needed to improve search engine performance especially online shop, which is contains so many product. The best way to come up with an optimal value is through experimenting.

Keywords: online shop, serach engine, concept-based, concept structure.

**SEARCH ENGINE PADA DOKUMEN ONLINE SHOP
MENGUNAKAN METODE BERBASIS KONSEP**

RIZKI ADI UTOMO

Skripsi
sebagai salah satu syarat untuk memperoleh gelar
Sarjana Komputer
pada
Departemen Ilmu Komputer

**DEPARTEMEN ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
INSTITUT PERTANIAN BOGOR
BOGOR
2016**

Penguji:

1 Mayanda Mega Santoni S.Komp, M.Kom

Judul Skripsi: *Search Engine* pada Dokumen *Online Shop* Menggunakan Metode
Berbasis Konsep
Nama : Rizki Adi Utomo
NIM : G64120094

Disetujui oleh

Ir Julio Adisantoso, Mkom
Pembimbing I

Dr Yeni Herdiyeni, SSi MKom
Pembimbing II

Diketahui oleh

Dr Ir Agus Buono, MSi, MKom
Ketua Departemen

Tanggal Lulus: _____

PRAKATA

Puji dan syukur penulis panjatkan kepada Allah *subhanahu wa ta'ala* atas segala karunia-Nya sehingga karya ilmiah ini berhasil diselesaikan. Tema yang dipilih dalam penelitian yang dilaksanakan sejak bulan November 2015 ini ialah temu kembali informasi, dengan judul *Search Engine* pada Dokumen Online Shop menggunakan Metode Berbasis Konsep.

Selain itu, penulis menyadari bahwa dalam proses penyusunan skripsi ini tidak lepas dari kontribusi dan dukungan semua pihak. Oleh karena itu, penulis ingin menyampaikan terima kasih kepada:

1. Bapak Ir Julio Adisantoso, MKom dan Ibu Dr Yeni Herdiyeni, SSi MKom selaku pembimbing yang telah mendukung, dan memberikan inspirasi yang luar biasa dalam penyusunan skripsi.
2. Ayahanda Susilo Utomo dan Ibunda Titin Astuti yang telah memberikan dukungan moral dan doa yang tak terbatas kepada penulis sehingga mampu menjalani banyak hal sampai tahap ini.
3. PT. Global Digital Niaga yang telah memberikan kesempatan untuk melakukan kerjasama terkait penelitian yang telah dilakukan.
4. Bapak Ifnu Bima, Bapak Ronald Prasetya, dan seluruh rekan-rekan karyawan Blibli.com yang telah membantu terkait penelitian.
5. Rijen Sianturi, Muhammad Syarif, dan Muhammad Ammar Imron sebagai teman satu bimbingan yang telah memberikan motivasi serta pengalaman-pengalaman yang tak terlupakan selama melakukan penelitian.
6. Teman-teman satu angkatan Ilmu Komputer IPB 49 yang telah memberikan semangat dalam melakukan penelitian, dan Meong yang telah membantu dan memberikan semangat dalam penyusunan skripsi.
7. Semua pihak yang telah memberikan kontribusi, dukungan, dan doa kepada penulis selama ini.

Penulis berharap kajian *Search Engine* pada Dokumen *Online Shop* menggunakan Metode Berbasis Konsep ini mampu memberikan manfaat dan sumbangsih terhadap khazanah ilmu pengetahuan.

Bogor, Agustus 2016

Rizki Adi Utomo

DAFTAR ISI

DAFTAR TABEL	vi
DAFTAR GAMBAR	vi
DAFTAR LAMPIRAN	vi
PENDAHULUAN	1
Latar Belakang	1
Perumusan Masalah	2
Tujuan Penelitian	2
METODE	3
Pengumpulan Dokumen	3
Pembuatan Indeks	5
Temu Kembali Berbasis Konsep	7
Apache Solr	9
Pemrosesan Kueri	10
Evaluasi	10
HASIL DAN PEMBAHASAN	12
Pemodelan Struktur Konsep	12
Model <i>User</i>	14
Implementasi dan Desain	15
Evaluasi	15
SIMPULAN DAN SARAN	18
Simpulan	18
Saran	19
DAFTAR PUSTAKA	19
RIWAYAT HIDUP	27

DAFTAR TABEL

1	Analisis teks berdasarkan <i>analysis chain</i>	6
2	<i>Confusion Matrix</i>	9
3	Struktur konsep pencarian online shop Blibli.com	12
4	Deskripsi use case sistem	14
5	Perbandingan struktur konsep berdasarkan nilai MAP	15
6	Perbandingan penggunaan metode struktur konsep pada Apache Solr	17

DAFTAR GAMBAR

1	Metodologi penelitian	3
2	Format dokumen produk online shop Blibli.com	4
3	Contoh dokumen produk Blibli.com	4
4	Tipe analyzer pada Apache Solr	7
5	Struktur konsep <i>online shop</i> Blibli.com	8
6	Struktur <i>XML</i> yang dihasilkan	12
7	Skema konsep Apache Solr Blibli.com	12
8	Implementasi algoritma <i>brute force</i>	13
9	Use case diagram tool evaluasi	14
10	Grafik perbandingan nilai <i>average interpolation precision</i>	17

DAFTAR LAMPIRAN

1.	Daftar <i>stopwords</i>	21
2.	Koleksi kueri	23
3.	Desain antarmuka	24
4.	<i>Precision</i> tiap kueri	26
5.	Grafik bobot konsep berdasarkan MAP	27

PENDAHULUAN

Latar Belakang

E-commerce atau *online shop* merupakan bidang usaha yang lebih menekankan pada penggunaan teknologi informasi dan komunikasi (TIK) dalam transaksi antarbisnis, antarorganisasi dan transaksi antara bisnis dengan konsumen. *E-commerce* menawarkan kemudahan dalam melakukan transaksi jual-beli, salah satunya berupa *search engine* atau mesin pencari. Dengan adanya mesin pencari, pengunjung situs *online shop* dapat melakukan pencarian terhadap produk yang diinginkan dan dapat mengakses informasi yang berkaitan dengan barang-barang dan layanan mereka, sehingga pada akhirnya melakukan pembelian terhadap barang tersebut (Munandar 2011). Platform pencarian yang banyak digunakan *online shop* adalah Apache Solr. Apache Solr adalah mesin pencari yang dibuat untuk skala enterprise, memiliki kemampuan pencarian secara cepat, dan memiliki skalabilitas yang tinggi yang dibangun menggunakan Apache Lucene (Shahi 2015).

Menemukan informasi yang tepat dalam koleksi dokumen yang besar merupakan hal yang sulit. Kesulitan tersebut disebabkan banyaknya mesin pencari yang menggunakan metode pencarian berbasis *keyword* (Haav dan Lubi 2005). Pada metode pencarian berbasis *keyword*, daftar *keyword* digunakan untuk menjelaskan konten dari suatu dokumen dan tidak memiliki hubungan semantik antar-*keywords*. Pada metode ini, hasil pencarian akan memberikan dokumen sebagai jawaban jika terdapat *keyword* yang sama dengan *keyword* yang menjelaskan suatu dokumen dan *keyword* yang terdapat pada kueri. Metode pencarian berbasis *keyword* memiliki kelemahan, yaitu efek pada kata polisemi dan efek pada kata homonim (Krovertz 1997). Efek pada kata polisemi menyebabkan dokumen yang tidak relevan terambil karena kata yang terdapat pada kueri memiliki banyak makna. Sementara itu, efek pada kata homonim menyebabkan dokumen relevan pada suatu koleksi dokumen yang tidak terambil karena menggunakan kata yang sama pada kata yang terdapat pada kueri namun memiliki makna yang berbeda.

Banyak metode yang digunakan untuk memecahkan masalah penggunaan kata pada mesin pencari. Salah satu pendekatan yang telah digunakan adalah mempertimbangkan aspek semantik pada dokumen. Pendekatan ini melihat dokumen sebagai koleksi konsep dan melakukan uji kemiripan antara kueri dan dokumen pada konsep tersebut, seperti *Implemented fuzzy logic to automatically construct concept from concept hierarchies* (Goyal 2009). Selain itu, mengingat bahwa dokumen bukan saja merupakan koleksi *keyword* tetapi juga merupakan koleksi konsep, diperlukan suatu teknik temu kembali berbasis konsep. Teknik temu kembali berbasis konsep berbeda dengan temu kembali berbasis *keyword* pada tahap temu kembali. Teknik temu kembali berbasis konsep mengurutkan dokumen berdasarkan kombinasi kemiripan kata dan konsep.

Pendekatan berbasis konsep pada temu kembali informasi telah menjadi teknik yang mengedepankan kreativitas peneliti pada penelitian temu kembali informasi dan juga digunakan pada ilmu disiplin lain (Cleary *et al.* 2008). Meskipun terdapat perbedaan pada cara bagaimana membangun struktur konsep pada tahap implementasi, pendekatan berbasis konsep merupakan pendekatan yang lebih baik daripada pendekatan berbasis kata. Beberapa sistem menggunakan taksonomi

konseptual sebagai struktur konsep, seperti *Sun Microsystems Conceptual Indexing* (Woods 1997) atau menggunakan analisis semantik eksplisit yang menggunakan teknik temu kembali berbasis konsep (Egozi *et al.* 2011). Pemilihan *domain* ontologi sebagai struktur konseptual merupakan model yang paling banyak dipilih karena mempunyai beberapa kelebihan pada saat implementasi. Beberapa sistem yang menggunakan *domain* ontologi adalah *Ontoseek* (Guarino *et al.* 1999), *Ontobroker*, (Haav dan Lubi 2005) dan *Concept-based Information Retrieval with Ontology in Cross-Language Searching* (Rad *et al.* 2010). Namun ontologi juga mempunyai kekurangan seperti pengetahuan yang sudah dibuat seringkali tidak sesuai dengan kebutuhan *user* terkait mendapatkan konsep yang diinginkan. Selain itu, penggunaan ontologi juga memerlukan usaha yang lebih untuk membangun sebuah pengetahuan secara manual dan memerlukan sumber daya waktu yang tidak sedikit (Goyal *et al.* 2009).

Saat ini sistem temu kembali yang diimplementasikan pada *online shop* Blibli.com sudah berbasis konsep. Hal ini karena Apache Solr merupakan salah satu platform pencarian yang menerapkan sistem temu kembali berbasis konsep. Namun kombinasi bobot pada struktur konsep yang digunakan masih belum dipastikan apakah merupakan kombinasi yang paling baik di antara kombinasi struktur konsep lainnya. Penelitian ini melakukan percobaan terhadap kombinasi bobot konsep dengan menggunakan dokumen *online shop* Blibli.com.

Perumusan Masalah

Penelitian ini dilakukan untuk menjawab masalah-masalah sebagai berikut:

- 1 Bagaimana kinerja pencarian temu kembali *online shop* Blibli.com menggunakan metode berbasis konsep?
- 2 Konsep apa yang paling berpengaruh terhadap hasil temu kembali dengan dokumen produk Blibli.com?
- 3 Bagaimana kombinasi struktur konsep pada dokumen produk *online shop* yang menghasilkan nilai *mean average precision* tertinggi?

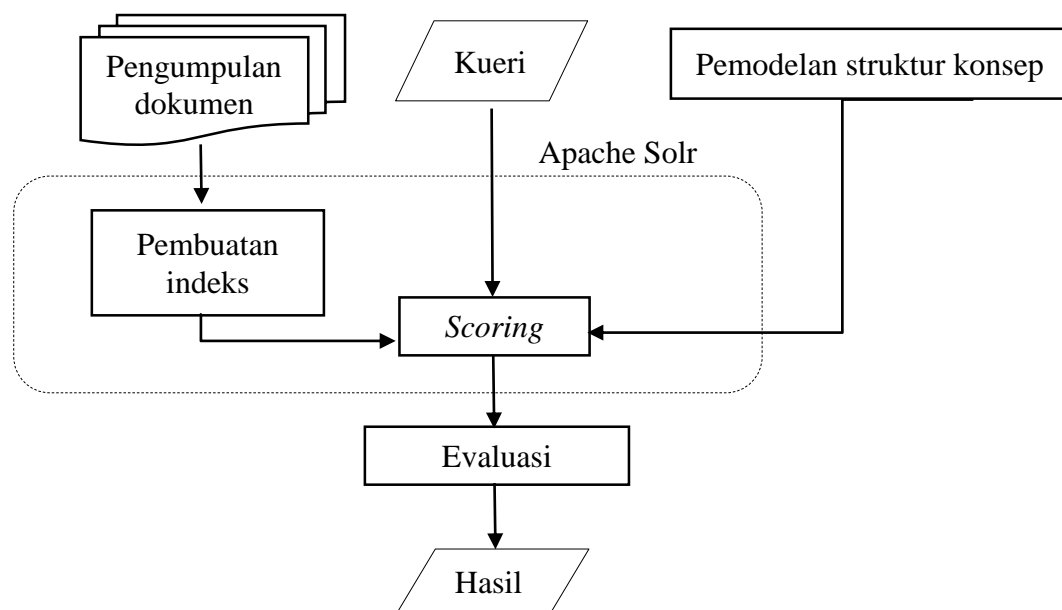
Tujuan Penelitian

Tujuan dari penelitian ini adalah:

- 1 Melakukan analisis kinerja sistem temu kembali menggunakan metode berbasis konsep pada dokumen produk *online shop* Blibli.com
- 2 Melakukan analisis terhadap konsep dari struktur dokumen yang memberikan pengaruh paling besar pada hasil temu kembali.
- 3 Melakukan analisis terhadap kombinasi struktur konsep dengan nilai *mean average precision* tertinggi pada dokumen produk *online shop* Blibli.com.

METODE

Penerapan aplikasi dari sistem temu-kembali informasi adalah *search engine* atau mesin pencari yang terdapat pada jaringan internet. Pengguna dapat mencari suatu informasi dengan melakukan penelusuran pada halaman-halaman *web* dengan menggunakan mesin pencari (Zuliarso 2010). Mesin pencari (*search engine*) adalah salah satu sistem temu-kembali informasi yang mengolah informasi dan mengambil daftar, peringkat maupun urutan dari dokumen berdasarkan relevansi antara *query* dengan dokumen yang dibutuhkan dalam rangka memenuhi pencarian yang dilakukan oleh *user*. Mesin pencari pada dasarnya terbagi menjadi dua komponen utama, yaitu pengindeksan (*indexing*) dan gabungan antara *user interface* dan *look-up-table* (Sudirman dan Kodar 2012). Pada penelitian ini, kedua komponen tersebut dilakukan oleh *platform* pencarian yang digunakan, yakni Apache Solr. Apache Solr merupakan platform pencarian yang melakukan pencarian berdasarkan konsep. Apache solr memiliki konfigurasi struktur konsep yang dapat diubah nilainya sesuai kebutuhan (Shahi 2015). Tahap-tahap yang dilakukan pada penelitian ini dapat dilihat pada Gambar 1.



Gambar 1 Metode penelitian

Pengumpulan Dokumen

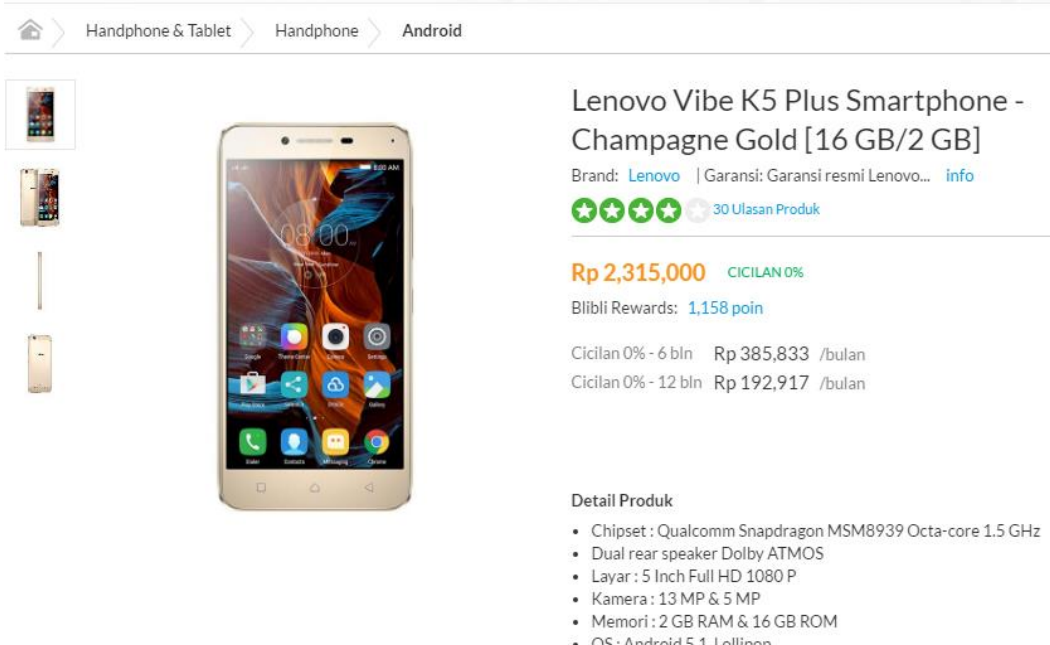
Data yang digunakan pada penelitian ini berasal dari *database* produk yang dimiliki oleh *online shop* Blibli.com dengan jumlah 76.518 dokumen. Dokumen diperoleh dari data produk yang terindeks pada *server user acceptance test* Blibli.com. Dokumen ini memiliki bentuk *XML* yang seragam untuk setiap dokumen. Format struktur *XML* untuk dokumen produk Blibli.com dapat dilihat pada Gambar 2 dan Gambar 3 untuk contoh dokumen produk Blibli.com.

```

<doc>
<str name="brandSearch">Sanrio</str>
<arr name="categories">
<str>Kuliner</str>
<str>Penyimpanan Dapur</str>
<str>Botol Minum</str>
</arr>
<str name="descriptionSearch">
Sanrio Daniel Pc Love Shoppe Bottle 560 ml Pink, botol minuman
berbahan plastik yang didesain untuk si kecil dengan Hello Kitty
themes yang cute.
</str>
<str name="nameSearch">
Sanrio Daniel Pc Love Shoppe Bottle 560 ml Pink
</str>
</doc>

```

Gambar 2 Format dokumen produk online shop Blibli.com



Handphone & Tablet > Handphone > Android

Lenovo Vibe K5 Plus Smartphone - Champagne Gold [16 GB/2 GB]

Brand: [Lenovo](#) | Garansi: Garansi resmi Lenovo... [info](#)

★★★★☆ 30 Ulasan Produk

Rp 2,315,000 CICILAN 0%

Blibli Rewards: 1,158 poin

Cicilan 0% - 6 bln Rp 385,833 /bulan
Cicilan 0% - 12 bln Rp 192,917 /bulan

Detail Produk

- Chipset : Qualcomm Snapdragon MSM8939 Octa-core 1.5 GHz
- Dual rear speaker Dolby ATMOS
- Layar : 5 Inch Full HD 1080 P
- Kamera : 13 MP & 5 MP
- Memori : 2 GB RAM & 16 GB ROM
- OS : Android 5.1, Lollipop

Gambar 3 Contoh dokumen produk Blibli.com

Dokumen produk *online shop* dikelompokkan ke dalam beberapa *tag*. *Tag* digunakan untuk mengidentifikasi struktur elemen dari suatu dokumen. Dokumen produk Blibli.com dapat dikelompokkan berdasarkan *tag* menjadi tiga bagian, yaitu *tag* `<doc>` yang mewakili keseluruhan dokumen, *tag* `<str>` yang mewakili data dokumen dengan tipe data *string*, dan *tag* `<arr>` yang mewakili data dokumen dengan tipe data *array*. Beberapa *tag* memiliki atribut nama yang berbeda-beda dan akan digunakan oleh Apache Solr untuk melakukan pencarian berdasarkan kueri tertentu. Pada Gambar 5 terdapat beberapa *tag* dengan empat atribut yang berbeda-beda. *Tag* dengan nilai atribut *nameSearch* dapat dilihat pada tulisan 'Lenovo Vibe

K5 Plus Smartphone-Champagne Gold [16 GB/2 GB]' yang mewakili nama dari suatu produk. *Tag* dengan nilai atribut *categories* dapat dilihat pada tulisan 'Handphone & Tablet' yang mewakili kategori dari suatu produk. *Tag* dengan nilai atribut *brandSearch* dapat dilihat pada teks yang terdapat pada penjelasan 'Brand', yaitu Lenovo, yang mewakili merek suatu produk. Adapun *tag* dengan nilai atribut *descriptionSearch* dapat dilihat pada penjelasan 'Detail Produk', yang mewakili deskripsi suatu produk.

Pembuatan Indeks

Pembuatan indeks diperlukan untuk melakukan representasi terhadap suatu dokumen. Indeks adalah gugus kata atau konsep terpilih sebagai penunjuk ke informasi atau dokumen terkait. Indeks dalam berbagai bentuk merupakan inti setiap sistem temu kembali informasi modern karena menyediakan akses yang lebih cepat dalam memperoleh informasi dan mempercepat pemrosesan kueri (Baeza-Yates dan Ribeiro-Neto 1999). Proses pembuatan indeks pada penelitian ini dilakukan oleh Apache Solr dan menggunakan teknik *inverted index*.

Proses pembuatan indeks dapat dilakukan secara mudah jika data tersebut memiliki struktur dan format yang baik, namun pembuatan indeks akan sulit dilakukan jika data tidak memiliki struktur yang baik dan tersedia pada format dan sumber data yang berbeda (Shahi 2015). Apache Solr mendukung berbagai format dokumen termasuk *XML* dan *JSON* sebagai dokumen masukan. Struktur file yang diindeks akan berisi kumpulan dokumen, dan setiap dokumen akan memiliki *field* dan konten yang harus diindeks berdasarkan *field* tersebut. Nama *field* yang terdapat pada dokumen masukan harus dihubungkan dengan nama *field* yang didefinisikan pada *file schema.xml*.

Inverted index merupakan bentuk struktur data pada *search engine* yang berisi kata-kata yang unik dan merujuk pada semua dokumen dimana kata tersebut berada. Tiap kata pada dokumen adalah *key*, dan *value* nya adalah kumpulan dokumen dimana kueri tersebut berada. Struktur data Lucene merupakan *inverted data index*, karena traversalnya merupakan *backward / inverted* dari kata ke dokumen. *Inverted index* mempunyai kemiripan pada halaman index pada akhir buku, yang berisi kumpulan kata dan merujuk pada halaman dimana kata tersebut berada (Shahi 2015).

Tokenisasi

Tokenisasi adalah proses memotong teks input menjadi unit-unit terkecil yang disebut token dan pada saat yang sama dimungkinkan untuk membuang karakter tertentu, seperti tanda baca (Manning *et al.* 2008). Token tersebut dapat berupa suatu kata, angka, atau suatu tanda baca. Proses ini bertujuan untuk mempermudah dalam mengetahui frekuensi kemunculan tiap token pada suatu dokumen. Pada penelitian ini proses tokenisasi dilakukan oleh Apache Solr. Proses tokenisasi dilakukan pada saat Apache Solr melakukan proses *indexing* terhadap suatu dokumen dan pada saat melakukan proses pencarian terhadap suatu kueri (Shahi 2015).

Filtering

Filtering adalah proses menghilangkan kata yang sudah didefinisikan pada sebuah daftar *stopword*. Eliminasi *stopwords* memiliki banyak keuntungan, yaitu mengurangi *space* pada tabel *term* indeks hingga 40 persen atau lebih (Baeza-Yates dan Ribeiro-Neto 1999). Pembuangan *stopword* adalah proses pembuangan *term* yang tidak memiliki arti atau tidak relevan. *Term* yang diperoleh dari tahap tokenisasi dicek dalam suatu daftar *stopword*, apabila sebuah kata masuk di dalam daftar *stopword* maka kata tersebut masuk ke proses berikutnya. Pada penelitian ini daftar *stopwords* yang digunakan memiliki jumlah 327 kata (Lampiran 1).

Stemming

Proses *stemming* digunakan untuk mengubah *term* yang masih melekat dalam *term* tersebut berupa awalan, sisipan, dan akhiran. Proses *stemming* dilakukan dengan cara menghilangkan semua imbuhan (*affixes*) baik yang terdiri dari awalan (*prefixes*), sisipan (*infixes*), akhiran (*suffixes*) dan *confixes* (kombinasi dari awalan dan akhiran) pada kata turunan. *Stemming* digunakan untuk mengganti bentuk dari suatu kata menjadi kata dasar dari kata tersebut yang sesuai dengan struktur morfologi Bahasa Indonesia yang benar (Tala 2003).

Chain analyzer

Praproses pada saat pencarian dan pengindeksan dokumen dilakukan oleh platform pencarian Apache Solr menggunakan *chain analyzer*. *Chain analyzer* adalah kumpulan dari beberapa proses yang bertujuan untuk melakukan analisis teks pada kueri. Proses yang pertama dilakukan adalah memecah kueri berdasarkan spasi menggunakan tokenizer *WhitespaceTokenizerFactory*. Proses kedua adalah melakukan pengecekan pada kata yang termasuk sebagai non-ASCII karakter, jika ditemukan non-ASCII karakter, maka dilakukan konversi menjadi ASCII terdekat. Proses ketiga adalah menghapus kata-kata yang berada pada daftar *stopword*. Proses keempat adalah melakukan konversi pada tiap huruf menjadi huruf kecil. Proses kelima adalah mengubah tiap kata menjadi kata dasar, dan proses terakhir adalah melakukan *trim* atau pemangkasan pada tiap kata (Shahi 2015). Contoh analisis teks menggunakan *analysis chain* dapat dilihat pada Tabel 1.

Tabel 1 Analisis teks berdasarkan *analysis chain*

<i>Analysis Chain</i>		Kueri			
WhitespaceTokenizeFactory	to	cuts	the	Grass	
AsciiFoldingFilterFactory	to	cuts	the	Grass	
StopFilterFactory		cuts		Grass	
LowerCaseFilterFactory		cuts		grass	
PorterStemFilterFactory		cut		grass	
TrimFilterFactory		cut		grass	

Proses tokenisasi pada penelitian ini dilakukan oleh Apache Solr menggunakan proses *analysis chain*. Pada saat dilakukan *indexing* terdapat tiga proses yang dilakukan berdasarkan tipe *analyzer indexing*, yaitu

standardTokenizerFactory, *StopFilterFactory*, dan *LowerCaseFilterFactory*. Pada saat dilakukan proses pencarian menggunakan kueri tertentu, terdapat empat proses yang dilakukan, yaitu *StandardTokenizerFactory*, *StopFilterFactory*, *SynonymFilterFactory* dan *LowerCaseFilterFactory*. *StandardTokenizerFactory* merupakan proses yang melakukan pemisahan kata berdasarkan spasi dan tanda baca. *StopFilterFactory* melakukan proses pembuangan kata pada tiap kata yang sudah didefinisikan pada *file stopwords*. *LowerCaseFilterFactory* merupakan proses yang melakukan konversi semua kata menjadi *lower case*. Adapun *synonymFilterFactory* merupakan proses *matching* kata berdasarkan *file* yang berisi kata dan kata yang mirip. Proses yang dilakukan Apache Solr baik pada saat *indexing* dan pencarian dapat dilihat pada Gambar 4.

```
<analyzer type="index">
  <tokenizer class="solr.StandardTokenizerFactory"/>
  <filter class="solr.StopFilterFactory"
ignoreCase="true" words="stopwords.txt" />
  <filter class="solr.LowerCaseFilterFactory"/>
</analyzer>

<analyzer type="query">
  <tokenizer class="solr.StandardTokenizerFactory"/>
  <filter class="solr.StopFilterFactory"
ignoreCase="true" words="stopwords.txt" />
  <filter class="solr.SynonymFilterFactory"
synonyms="synonyms.txt" ignoreCase="true" expand="true"/>
  <filter class="solr.LowerCaseFilterFactory"/>
</analyzer>
```

Gambar 4 Tipe analyzer pada Apache Solr

Temu Kembali Berbasis Konsep

Model temu kembali berbasis konsep merupakan model temu kembali yang menggunakan hubungan konseptual antara teks dengan suatu kumpulan dokumen. Model temu kembali berbasis konsep memiliki ide bahwa arti dari sebuah teks atau kata lebih bergantung pada hubungan konseptual daripada hubungan linguistik pada sebuah kumpulan dokumen. Oleh karena itu model temu kembali berbasis konsep akan menghubungkan kumpulan kata pada konsep, seperti isi dari kumpulan dokumen yang dideskripsikan dengan sekumpulan konsep. Hal yang penting pada model ini adalah keberadaan dari struktur konsep untuk menghubungkan objek deskripsi dari informasi dengan konsep yang digunakan pada saat memberikan kueri (Haav dan Lubi 2015). Jika kata kunci atau kata benda digunakan, maka kata-kata tersebut harus dihubungkan dengan konsep yang terdapat pada struktur konsep. Struktur konsep dapat berupa *domain* umum atau *domain* khusus, dan dapat dibuat secara manual atau otomatis. Perbedaan model temu kembali berbasis konsep dengan *XML retrieval* adalah ada pada kueri yang digunakan pengguna. *XML retrieval* memungkinkan pengguna untuk melakukan temukembali suatu bagian dokumen tertentu dan bukan bagian dokumen secara keseluruhan (Manning *et al.* 2008). Sehingga pengguna dapat berfokus pada strategi pengembalian komponen dokumen yang merupakan jawaban dari sebuah kueri.

Srtuktur Konsep

Struktur konsep dapat berupa *domain* general atau *domain* spesifik, bisa dibentuk secara otomatis atau manual, dan memiliki struktur yang berbeda-beda tergantung pada cara membangun suatu hubungan antarkonsep (Haav dan Lubi 2005). Untuk membuat definisi dari konsep dibutuhkan identifikasi terhadap teks dan kemudian melakukan klasifikasi terhadap konsep yang ditemukan berdasarkan struktur konseptual yang ada. Teks secara eksplisit lebih banyak mengandung kata daripada konsep. Karena konsep didefinisikan dengan bahasa alami, maka hal ini memungkinkan untuk melakukan identifikasi lebih jauh mengenai konsep pada teks dengan melakukan analisis terhadap frase. Pada banyak model temu kembali berbasis konsep, pemrosesan bahasa alami digunakan untuk melakukan analisis *syntax* dan semantik pada teks untuk melakukan kategorisasi (Adi *et al.* 1999).

Pada penelitian ini struktur konsep direpresentasikan sebagai *query boosting* pada platofrm pencarian Apache Solr yang digunakan Blibli.com. *Query boosting* merupakan salah satu faktor yang menentukan hasil dari *solr scoring*. Semakin tinggi nilai *boost* maka nilai dari *solr scoring* akan semakin tinggi. Penggunaan *query boosting* menyebabkan perhitungan pada *solr scoring* yang tidak sama rata terhadap semua konsep, melainkan bergantung pada bobot suatu konsep. Dalam menentukan bobot suatu konsep, seorang peneliti harus memperhatikan kebutuhan dari tingkat kepentingan suatu konsep pada kueri dan pengaruhnya terhadap hasil pencarian. Cara terbaik untuk mendapatkan hasil yang optimal dalam menentukan bobot suatu konsep diperlukan suatu percobaan (Shahi 2015). Blibli.com menggunakan Solr 4.3 yang memiliki kombinasi *default*, seperti yang terlihat pada Gambar 5.

```
<str name="qf">
brandSearch^8 nameSearch^9 categories^10 descriptionSearch^10
</str>
```

Gambar 5 Struktur konsep online shop Blibli.com

Pada Gambar 5, bobot konsep dari masing-masing konsep adalah 8 untuk konsep *brandSearch*, 9 untuk konsep *nameSearch*, 10 untuk konsep *categories*, dan 10 untuk *descriptionSearch*. Bobot konsep tersebut ditentukan dengan cara melakukan perkiraan dan tidak ada perhitungan empiris untuk membuktikanya. Adapun untuk konsep ditentukan dengan cara melihat konsep-konsep apa saja yang paling mempresentasikan dokumen produk *online shop* Blibli.com.

Pada saat melakukan improvisasi pada struktur konsep Apache Solr, diperlukan sebuah tools yang dikembangkan dengan bahasa pemrograman *Java*, menggunakan *Spring Framework*, *library Document Object Model*, dan platform pencarian Apache Solr. Adapun konsep yang digunakan yaitu, *brandSearch*, *nameSearch*, *categories*, dan *descriptionSearch*. *Tools* diperlukan dengan tujuan agar proses improvisasi dan percobaan terhadap variasi struktur konsep dapat dilakukan melalui sistem secara otomatis dan efisien.

Apache Solr

Apache Solr merupakan pengembangan dari Apache Lucene. Apache Lucene adalah teks *search engine library* dengan performa yang tinggi. Apache Lucene pertama kali dikembangkan dan mulai dipublikasi secara bebas tahun 2000 dan dikembangkan oleh Doug Cutting. Saat ini mesin pencari terbanyak menggunakan teknologi Lucene (Smiley dan Pugh 2011). Apache Solr merupakan platform pencarian yang paling banyak digunakan sebagai jawaban dari permasalahan pencarian pada teks. Hampir semua *website* menggunakan Apache Solr sebagai pendukung fitur pencarian pada suatu *website*. Beberapa kelebihan yang dimiliki solr yaitu memiliki skalabilitas yang tinggi, mendukung pencarian *full*-teks, fleksibel, dan memiliki konfigurasi yang mudah digunakan (Shahi 2015).

Solr scoring

Tingkat relevansi merupakan inti dari mesin pencari. Pada proses mengurutkan dokumen terhadap sebuah kueri, Apache Solr menghitung nilai dari setiap dokumen yang terambil berdasarkan tingkat relevansinya dari tertinggi ke rendah. Solr menggunakan *library* Apache Lucene sebagai standar pembobotan. Pembobotan lucene merupakan kombinasi dari *boolean model* dan *vector space model* (VSM). Pada kueri tertentu, tidak semua dokumen yang berada pada database dilakukan pengurutan, tetapi hanya dokumen yang memenuhi *boolean condition*. Proses ini juga akan mengurangi waktu proses dengan melakukan penyaringan pada dokumen yang tidak relevan sehingga dokumen yang telah diurutkan melalui *boolean model* selanjutnya akan diurutkan dengan VSM. Pada VSM, kueri dan dokumen direpresentasikan sebagai bobot vektor dalam ruang multidimensi, di mana tiap kata pada kueri adalah sebuah dimensi dan nilai TF-IDF mempresentasikan bobot. VSM menggunakan uji kemiripan *cosine* untuk menghitung nilai sebuah dokumen. Formula untuk menghitung nilai menggunakan modifikasi TF-IDF pada Lucene dapat dilihat pada formula berikut:

$$score = coord(q, d).queryNorm(q). \sum_{t \in q} (tf(t \text{ in } d).idf(t).t.getBoost().norm(t, d))$$

dengan fungsi $coord(q, d)$ adalah fungsi yang melakukan perhitungan berdasarkan jumlah kata pada kueri yang tersebar pada dokumen. Fungsi $queryNorm(q)$ adalah fungsi yang melakukan normalisasi nilai antarkueri. Fungsi $tf(t \text{ in } d)$ adalah fungsi yang menghitung frekuensi kemunculan suatu *term* t pada dokumen d dan fungsi $idf(t)$ adalah fungsi yang menghitung jumlah dokumen pada suatu korpus yang mengandung *term* t (Manning *et al.* 2008). Fungsi $t.getBoost()$ adalah fungsi yang menghitung nilai bobot kata pada saat melakukan pencarian. Fungsi $norm(t, d)$ merupakan penggabungan antara fungsi *lengthNorm* dan total waktu yang dibutuhkan untuk melakukan *indexing* pada dokumen. Adapun fungsi *LengthNorm* merupakan panjang dari *field* yang memberikan hasil yang signifikan pada dokumen.

Pemrosesan Kueri

Pada saat menjalankan proses evaluasi, kueri dimasukkan pada sistem pencari dan dilakukan pencarian dengan variasi 625 kombinasi bobot struktur konsep. Jumlah kueri yang digunakan pada penelitian ini ialah 34 kueri untuk dokumen *online shop* (Lampiran 2). Masing-masing kueri memiliki jumlah dokumen relevan yang diharapkan muncul yang direpresentasikan dengan *id* produk (*golden-list*). Proses menentukan dokumen relevan yang ditemukembalikan menggunakan kueri tertentu diidentifikasi berdasarkan pengalaman pengguna dalam melakukan pencarian. Jumlah dokumen relevan yang diharapkan akan ditemukembalikan memiliki jumlah yang berbeda untuk setiap kueri.

Evaluasi

Recall dan Precision

Terdapat banyak jenis ukuran yang dapat digunakan untuk mengevaluasi kinerja suatu sistem temu kembali informasi, antara lain *recall*, *precision*, *F measure*, *interpolated precision*, dan *mean average precision*. Pengukuran yang paling umum yang digunakan adalah *recall* dan *precision* (Manning *et al.* 2008). Perhitungan terhadap parameter *recall* dan *precision* dapat menggunakan komponen dari *confusion matrix* yang ditunjukkan pada Tabel 2.

Tabel 2 *Confusion Matrix*

	Relevan	Tidak Relevan
Ditampilkan	<i>tp</i>	<i>fp</i>
Tidak Ditampilkan	<i>fn</i>	<i>tn</i>

Recall adalah rasio jumlah dokumen relevan yang ditemukan kembali dengan total jumlah dokumen dalam kumpulan dokumen yang dianggap relevan. *Recall* dapat dirumuskan sebagai :

$$Recall = \frac{tp}{(tp+fn)}$$

Precision adalah rasio jumlah dokumen relevan yang ditemukan dengan total jumlah dokumen yang ditemukembalikan. *Precision* dapat dirumuskan sebagai berikut:

$$Precision = \frac{tp}{(tp+fp)}$$

Evaluasi menggunakan *precision* dan *recall* membutuhkan suatu perbandingan untuk menentukan relevansi suatu dokumen. Oleh karena itu, diperlukan suatu data yang berisi kueri dan hasil dokumen yang diharapkan muncul sehingga proses untuk menentukan relevansi suatu dokumen bisa dilakukan secara efektif dan efisien.

11-Point interpolated precision

Melakukan evaluasi dengan melihat nilai seluruh nilai *precision* dan *recall* merupakan langkah yang sangat informatif. Namun terkadang kita ingin melakukan evaluasi dengan melihat beberapa nilai saja atau bahkan dengan satu nilai agar evaluasi dapat dilakukan dengan cepat dan tepat (Manning *et al.* 2008). Cara tradisional untuk melakukan hal tersebut adalah dengan melihat *11-point interpolated precision*. Nilai *11-point interpolated precision* dapat dihitung menggunakan rumus berikut:

$$P_{interp}^{(r)} = \max_{r' \geq r} p(r')$$

dengan $P_{interp}^{(r)}$ menyatakan nilai *precision* pada tingkat *recall* r dan $p(r')$ menyatakan nilai pada *recall* r . *11-point interpolated precision* akan menghasilkan nilai *precision* pada tiap 11 titik *recall* yaitu 0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, dan 1.0 untuk setiap informasi yang dibutuhkan.

Mean average precision

Mean Average Precision (MAP) merupakan suatu perhitungan yang menghasilkan angka yang dapat mempresentasikan nilai *precision* dari tiap kumpulan kueri. MAP merupakan rata-rata nilai *precision* yang diperoleh dari kumpulan dokumen tertentu yang ditemukan setelah setiap dokumen yang relevan ditemukembalikan. MAP diperlukan untuk memudahkan dalam melakukan perbandingan pada tahap evaluasi. MAP dihitung dengan merata-ratakan nilai dari 11 titik rata-rata *interpolated precision* sehingga didapatkan nilai tunggal. *Mean average precision* dapat dirumuskan sebagai berikut:

$$MAP(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} Precision(R_{jk})$$

dengan $|Q|$ merupakan sejumlah kueri tertentu, m_j merupakan jumlah dokumen yang relevan, dan R_{jk} merupakan kumpulan dari hasil temukembali yang sudah diurutkan dari dokumen teratas sampai dokumen tertentu (Manning *et al.* 2008).

HASIL DAN PEMBAHASAN

Pemodelan Struktur Konsep

Pemodelan struktur konsep dilakukan dengan menentukan struktur *XML DOM tree* terlebih dahulu pada dokumen *XML*. Hal ini dilakukan agar halaman *XML* dapat diakses dengan mudah dan terstruktur. Tahap selanjutnya adalah melakukan temukembali pada parameter bobot struktur konsep pada skema konsep Apache Solr Blibli.com. Objek yang diambil sebagai *root element* untuk melakukan pemodelan adalah objek *requestHandler* dengan *attribute name browse*. Hal ini karena *requestHandler* dengan *attribute name browse* merupakan *requestHandler* yang digunakan pada fitur pencarian *online shop* Blibli.com. Objek lain yang diambil adalah *lst* sebagai elemen, dan elemen *str* dengan atribut *name* dengan nilai *qf*. Setelah berhasil mengakses elemen yang paling spesifik, didapatkan struktur konsep Apache Solr Blibli.com dengan tipe data *String*. Struktur *XML* yang dihasilkan dapat dilihat pada Gambar 6. Gambar 7 untuk skema konsep Apache Solr dan Tabel 3 untuk keterangan tiap *field*.

```
<requestHandler class="solr.SearchHandler" name="/browse">
  <lst name="defaults">
    <!-- Query settings -->
    <str name="defType">edismax</str>
    <str name="qf">brandSearch^8 nameSearch^9
categories^10 descriptionSearch^10 </str>
    <str name="df">searchFields</str>
    <str name="mm">100%</str>
    <str name="q.alt">*:*</str>
    <str name="rows">10</str>
    <str name="fl">*,score</str>
  </lst>
</requestHandler>
```

Gambar 6 Struktur XML yang dihasilkan

```
<field name="brandSearch" type="teks_general"
indexed="true" stored="true"/>
<field name="nameSearch" type="teks_general"
indexed="true" stored="true" required="true"/>
<field name="descriptionSearch" type="teks_general"
indexed="true" stored="true"/>
<field name="categories" type="teks_general"
indexed="true" stored="true" multiValued="true"/>
```

Gambar 7 Skema konsep Apache Solr Blibli.com

Objek spesifik yang sudah diambil selanjutnya dilakukan pemisahan antara konsep dan bobot pada struktur konsep. Pada Gambar 6 terlihat bahwa karakter pemisah antara bobot dan konsep adalah karakter caret ('^'). Dengan demikian

pemisahan bisa dilakukan menggunakan metode *split* pada bahasa pemrograman Java.

Tabel 3 Struktur konsep pencarian *online shop* Blibli.com

<i>Field</i>	Keterangan
brandSearch	<i>Field</i> ini mendefinisikan <i>brand</i> suatu produk yang tersedia
nameSearch	<i>Field</i> ini mendefinisikan nama dari suatu produk
descriptionSearch	<i>Field</i> ini mendefinisikan deskripsi dari suatu produk
Categories	<i>Field</i> ini mendefinisikan kategori dari suatu produk

Proses improvisasi pada struktur konsep pencarian *online shop* Blibli.com menggunakan algoritma *brute force*. Algoritma *brute force* yang digunakan pada penelitian ini melakukan operasi pembentukan struktur konsep dengan 4 konsep dan nilai bobot pada konsep yang memiliki *range* antara 1 sampai 10 dengan interval sebesar 2. Variabel *w*, *x*, *y*, *z* digunakan untuk mendeklarasikan 4 konsep yang tersedia, tipe data *ArrayList* dibutuhkan untuk membuat struktur konsep dengan bobot konsep yang memiliki nilai *range* yang telah ditentukan. Variasi nilai bobot tersebut menghasilkan 625 kombinasi struktur konsep. Implementasi algoritma *brute force* dapat dilihat pada Gambar 8.

```
while (w < 5) {
    List<String> num1 = new ArrayList<>();
    num1.add(num[w]); num1.add(num[x]); num1.add(num[y]);
    num1.add(num[z]);
    z++; if (z == 5) {y++; z = 0;} if (y == 5) {x++; y = 0;}
    if (x == 5) {w++; x = 0;}
}
```

Gambar 8 Implementasi algoritma brute force

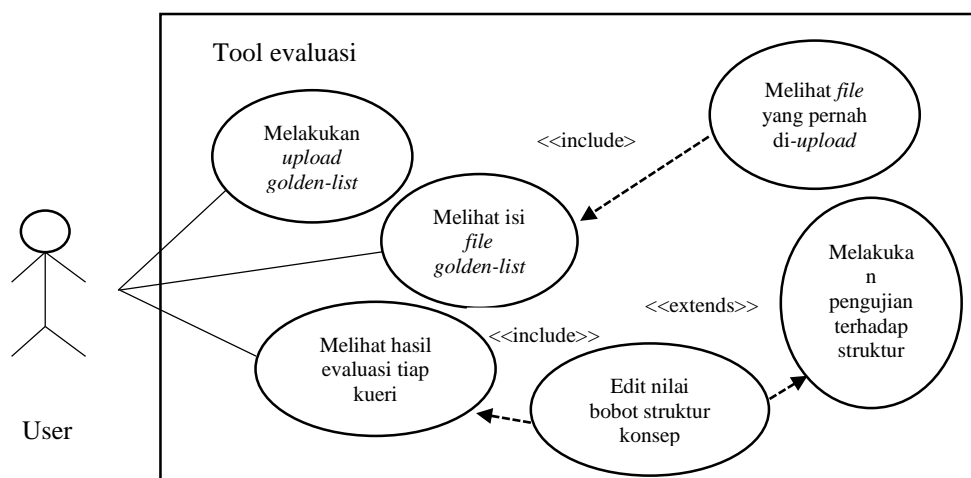
Struktur konsep yang didapatkan dari algoritma *brute force* kemudian diimplementasikan pada Apache Solr dan diuji dengan 34 kueri yang terdapat pada *golden-list* melalui sebuah tools. *Golden-list* merupakan dokumen yang mengandung list berisi kueri-kueri dan dokumen yang seharusnya diterima berdasarkan kueri tersebut. Fungsi lain dari tools yang dikembangkan adalah melakukan improvisasi pada *field-field* pencarian yang dilakukan Apache Solr. Parameter *field* yang ditemukembalikan merupakan parameter *field* yang tersedia pada skema solr yang mempunyai struktur dokumen *XML*.

Implementasi bobot konsep yang telah diubah memerlukan proses *reload collection* yang digunakan pada Apache Solr. Proses *reload collection* diperlukan agar Apache Solr melakukan proses *update* terhadap struktur kombinasi yang telah diubah. Pada aplikasi yang dibuat, proses *reload collection* dilakukan menggunakan API Apache Solr yaitu *collection API*. *Collection API* memiliki *method* untuk melakukan proses *reload* terhadap *collection* yang digunakan tanpa perlu

melakukan *restart* Apache Solr secara manual melalui *command shell*. Dengan demikian proses melakukan uji coba terhadap 625 kombinasi struktur konsep dapat tetap dilakukan secara otomatis melalui *tool*.

Model User

Implementasi model informasi ke dalam sistem memerlukan model *user*. Model *user* berfungsi sebagai *controller* untuk menampilkan informasi yang diperoleh dari model informasi ke tampilan *web* atau *views*. *Controller* yang akan dibuat terdiri atas fungsi-fungsi berdasarkan *use case* atau aktivitas yang dilakukan *user* terhadap sistem. Aktivitas ini digambarkan pada *use case diagram* pada Gambar 9.



Gambar 9 Use case diagram tool evaluasi

Terdapat satu aktor dan enam aktivitas pada *use case* yang dihasilkan. Gambar 9 menampilkan aktivitas aktor. Aktor *user* memiliki aktivitas melakukan *upload file golden-list*, melihat isi dari *file golden-list*, melihat *file* yang pernah di-*upload*, melihat hasil evaluasi tiap kueri, edit nilai bobot struktur konsep, dan melakukan pengujian terhadap struktur konsep.

Tabel 4 menjelaskan deskripsi untuk setiap aktivitas. Kolom pertama pada tabel menunjukkan aktivitas dan kolom kedua merupakan deskripsi dari aktivitas yang bersangkutan.

Tabel 4 Deskripsi *use case* sistem

Aktivitas	Deskripsi
Melihat file yang pernah di- <i>upload</i>	Untuk dapat melihat <i>file-file</i> apa saja yang pernah di <i>upload</i> kedalam sistem. Selain itu, <i>user</i> juga dapat melihat detail dari <i>file</i> yang pernah di <i>upload</i> dengan mengeklik <i>button</i> yang berada pada setiap <i>file</i> . (Lampiran 3a).

Aktivitas	Deskripsi
Melakukan <i>upload golden-list</i>	<i>User</i> mengakses halaman <i>upload</i> dari sistem dan menekan tombol <i>button browse</i> untuk memilih file yang akan di <i>upload</i> (Lampiran 3b).
Edit nilai bobot struktur konsep Apache Solr	<i>User</i> dapat melakukan proses edit pada nilai tiap konsep yang digunakan, dengan memberikan nilai pada <i>field</i> yang tersedia. Data yang diubah harus berupa angka dan tidak boleh kurang dari angka 0 (Lampiran 3c).
Melakukan pengujian terhadap kombinasi struktur konsep Apache Solr	<i>User</i> dapat melakukan pengujian terhadap kombinasi struktur konsep solr, dengan menekan tombol <i>process query</i> . Pengujian dilakukan terhadap seluruh kueri yang sudah di- <i>upload</i> dengan membandingkan id dokumen yang diharapkan muncul dan id dokumen yang terambil pada kueri tertentu (Lampiran 3d).
Melihat isi <i>file golden-list</i>	<i>User</i> melihat kumpulan kueri dan id dokumen produk yang diharapkan muncul pada kueri tertentu, id produk diilustrasikan dengan kotak berwarna <i>orange</i> (Lampiran 3e).
Melihat informasi hasil temu kembali	<i>User</i> dapat melihat hasil temu kembali dari <i>golden-list</i> tertentu. Informasi yang ditampilkan berupa nilai <i>interpolated average precision</i> dari tiap kueri, nilai <i>mean average precision</i> , dan grafik <i>11-interpolated precision</i> (Lampiran 3f).

Implementasi dan Desain

Implementasi Model User

Pembangunan sistem dilakukan menggunakan *web framework* Spring. Spring merupakan *framework* yang menggunakan pendekatan *model view controller* (MVC) dan *repository* Maven untuk *depedencies*. *Models* mengandung fungsi kueri yang dibuat pada tahap implementasi model informasi, sedangkan model *user* berisi *controllers* untuk memanggil *models* dan menampilkannya ke *views* atau halaman HTML.

Desain Antarmuka

Antarmuka dibuat berdasarkan fungsi yang terdapat dalam sistem yaitu fungsi untuk melakukan *upload file*, menampilkan isi dari *file* yang diupload, menampilkan informasi tentang hasil evaluasi yang dilakukan terhadap *file* tertentu, dan melakukan perubahan pada bobot konsep Apache Solr. Desain antarmuka dapat dilihat pada Lampiran 3.

Evaluasi

Evaluasi dilakukan dengan menghitung nilai *mean average precision* dan *interpolated average precision*. Perhitungan nilai *interpolated average precision* pada setiap kueri menggunakan struktur konsep dengan nilai MAP tertinggi dapat

dilihat pada Lampiran 4. Pada 625 struktur konsep terdapat kombinasi struktur konsep *brandSearch* dengan bobot sebesar 2, *nameSearch* dengan bobot sebesar 2, *categories* dengan bobot sebesar 6, dan *descriptionSearch* dengan bobot sebesar 6, menghasilkan nilai *mean average precision* tertinggi, yaitu 0.80. Struktur konsep dengan MAP terendah memiliki kombinasi struktur konsep *brandSearch* dengan bobot sebesar 2, *nameSearch* dengan bobot sebesar 2, *categories* dengan bobot sebesar 2, dan *descriptionSearch* dengan bobot sebesar 10, dengan nilai MAP, yaitu 0.55. Struktur konsep *default* Blibli.com memiliki kombinasi struktur konsep *brandSearch* dengan bobot sebesar 8, *nameSearch* dengan bobot sebesar 9, *categories* dengan bobot sebesar 10, dan *descriptionSearch* dengan bobot sebesar 10 menghasilkan nilai MAP sebesar 0.77. Perbedaan nilai MAP pada tiap kombinasi disebabkan setiap kombinasi bobot mempengaruhi nilai *scoring* pada pembobotan Apache Solr sehingga dokumen yang ditemukembalikan memberikan hasil yang berbeda untuk tiap struktur konsep. Perbandingan struktur konsep berdasarkan nilai MAP dapat dilihat pada Tabel 5.

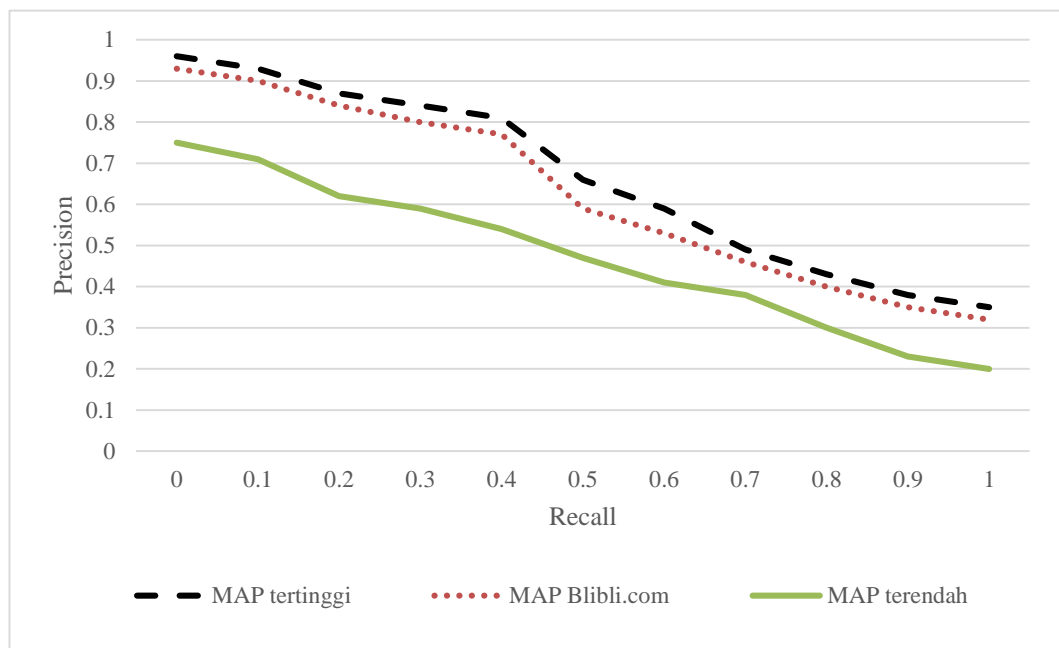
Tabel 5 Perbandingan struktur konsep berdasarkan nilai MAP

Struktur konsep	MAP
brandSearch=2, nameSearch=4, categories=6, descriptionSearch=6	0.80
brandSearch=8, nameSearch=9, categories=10, descriptionSearch=10*	0.77
brandSearch=2, nameSearch=2, categories=2, descriptionSearch=10	0.55

*Struktur konsep default Blibli.com

Dari Tabel 5, diketahui bahwa selisih nilai MAP antara struktur konsep dengan MAP tertinggi dan struktur konsep *default online shop* Blibli.com adalah 0.03, dan selisih nilai MAP antara struktur konsep *default online shop* Blibli.com dan struktur konsep dengan nilai MAP terendah adalah 0.22. Hal ini disebabkan nilai bobot konsep *categories* pada struktur konsep dengan nilai MAP tertinggi dan struktur konsep *default* Blibli.com memiliki nilai yang besar. Adapun pengaruh konsep *brandSearch* dan *descriptionSearch* tidak memberikan pengaruh yang besar terhadap kinerja pencarian, sehingga selisih nilai MAP antara struktur konsep dengan nilai tertinggi dan struktur konsep *default* Blibli.com tidak terlalu tinggi. Sementara itu, selisih nilai MAP pada struktur konsep *default* Blibli.com dan struktur konsep dengan nilai MAP terendah memiliki nilai yang besar. Hal ini karena pada struktur konsep dengan nilai MAP terendah memiliki bobot konsep *categories* paling minimum, yaitu dua. Adapun konsep *categories* merupakan konsep yang seharusnya memiliki bobot tertinggi karena hampir semua kata pada kueri yang terdapat pada *golden-list* mengandung kata yang terdapat pada konsep *categories* pada dokumen. Dengan demikian, bobot terkecil pada konsep *categories* pada struktur konsep dengan nilai MAP terendah mengakibatkan selisih antara nilai MAP pada struktur konsep *default* Blibli.com dan struktur konsep dengan nilai MAP terendah menjadi tinggi. Adanya selisih yang dimiliki ketiga struktur konsep diatas dapat ditentukan bahwa struktur konsep dengan nilai MAP tertinggi dapat meningkatkan kinerja pencarian situs *online shop* Blibli.com sebesar 3.9 persen, sedangkan struktur konsep dengan nilai MAP terendah dapat menurunkan kinerja pencarian situs *online shop* Blibli.com sebesar 28 persen. Untuk perbandingan nilai

11- *interpolated precision* di setiap titik *level recall* pada setiap struktur konsep dapat dilihat pada Gambar 10.



Gambar 10 Grafik perbandingan nilai *average interpolation precision*

Pada Gambar 10 terlihat bahwa pada struktur konsep dengan nilai MAP tertinggi akan menampilkan 35 persen dokumen yang relevan dari semua dokumen yang ditampilkan pada saat sistem mengembalikan semua dokumen yang relevan. Struktur konsep *default* Blibli.com akan menampilkan 31 persen dokumen yang relevan dari semua dokumen yang ditampilkan pada saat sistem mengembalikan semua dokumen yang relevan. Adapun untuk struktur konsep dengan nilai MAP terendah akan menampilkan 20 persen dokumen yang relevan dari semua dokumen yang ditampilkan pada saat sistem mengembalikan semua dokumen yang relevan.

Analisis Pengaruh Penggunaan Bobot Konsep Terhadap Hasil Temu Kembali

Pengujian terhadap penggunaan metode struktur konsep pada pencarian Apache Solr dilakukan dengan membandingkan antara penggunaan struktur konsep dengan menggunakan bobot tertentu dan tidak menggunakan bobot. Apache Solr akan memberikan bobot senilai satu, jika bobot pada konsep tidak ditentukan. Pengaruh penggunaan struktur konsep terhadap hasil temu kembali dapat dilihat pada Tabel 6.

Tabel 6 Perbandingan penggunaan metode struktur konsep pada Apache Solr

Nilai <i>boosting</i>				MAP
<i>nameSearch</i>	<i>brandSearch</i>	Categories	<i>DescriptionSearch</i>	
2	4	6	6	0.80
1	1	1	1	0.76

Pada Tabel 6 terlihat selisih MAP pada kedua pengujian struktur konsep memiliki selisih 0.04. Dengan demikian dapat ditentukan bahwa struktur konsep dengan bobot yang menghasilkan nilai MAP tertinggi dapat meningkatkan kinerja pencarian *online shop* Blibli.com (jika setiap bobot konsep bernilai satu) sebesar 5.2 persen.

Analisis Pengaruh Antar-Konsep Terhadap Hasil Temu Kembali

Analisis pengaruh antar-konsep terhadap hasil temu kembali bertujuan untuk mengetahui konsep yang paling berpengaruh dalam menentukan nilai MAP. Analisis dilakukan dengan melakukan perbandingan pada 625 struktur konsep berdasarkan nilai MAP yang sudah diurutkan dari rendah ke tinggi. Pada konsep *brandSearch* (Lampiran 5a) dan konsep *descriptionSearch* (Lampiran 5b) terlihat nilai bobot yang konstan, sedangkan untuk konsep *categories* (Lampiran 5c) dan *nameSearch* (Lampiran 5d) terjadi perubahan nilai bobot yang memiliki pola. Pada konsep *categories*, semakin nilai MAP meningkat maka bobotnya juga akan semakin meningkat. Hal ini disebabkan kata-kata pada kueri yang terdapat di dalam *golden-list* mengandung banyak kata yang terdapat pada konsep kategori, seperti jas hujan, sepatu, kamera, aksesoris rambut, dan motor. Adapun untuk konsep *nameSearch*, semakin nilai MAP meningkat maka bobotnya akan semakin menurun. Perubahan nilai terjadi akibat beberapa kueri yang berada pada *golden-list* mengandung kata-kata bahasa indonesia, seperti jas hujan yang dalam bahasa inggris diterjemahkan menjadi *raincoat*, kipas angin menjadi *fan*, mesin kopi menjadi *coffee maker*, pengering rambut menjadi *hair dryer*, sepatu *boot black* menjadi *black fat chukka*, lipstik mejadi *lipstick*, raket menjadi *racket*, dan ranjang bayi menjadi *Babybelle*. Selain itu penyebab lain adalah beberapa kueri pada *golden-list* tidak menyebutkan *brand*. Pada kasus kueri ‘botol susu’, hasil yang ditampilkan lebih banyak kata-kata yang mempresentasikan sebuah konsep *brandSearch* daripada konsep *nameSearch*. Hal yang sama juga berlaku pada kueri ‘dot bayi’. Oleh karena itu, nilai bobot pada konsep *nameSearch* akan menurun dari MAP yang sudah terurut dari rendah ke tinggi.

SIMPULAN DAN SARAN

Simpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat ditarik kesimpulan sebagai berikut:

1. Telah ditemukan kombinasi struktur konsep dengan MAP tertinggi pada Apache Solr dengan data produk Blibli.com yaitu, *brandSearch* dengan bobot 2, *field nameSearch* dengan bobot 4, *field categories* dengan 6, dan *field descriptionSearch* dengan bobot 6. Kombinasi struktur tersebut menghasilkan nilai *mean average precision* sebesar 0.80.
2. Pengaruh penggunaan metode struktur konsep dengan MAP tertinggi dapat meningkatkan kinerja pencarian pencarian Apache Solr pada situs *online shop* Blibli.com sebesar 3.9 persen.

- 3 Konsep *categories* memberikan pengaruh paling besar pada hasil temu kembali disebabkan kueri pada *golden-list* yang lebih banyak mengandung kata-kata yang terdapat pada konsep *categories*.

Saran

- 1 Kueri yang terdapat pada *golden-list* masih sedikit sehingga tidak benar benar mempresentasikan kueri yang paling sering digunakan pengguna. Dengan demikian, kueri yang terdapat pada *golden-list* dapat ditambahkan sehingga kueri-kueri yang terdapat pada *golden-list* akan semakin mempresentasikan kueri yang paling sering digunakan pengguna *online shop*.
- 2 Konsep-konsep yang ada masih sedikit, sehingga perlu ditambahkan dengan konsep lain yang tersedia pada struktur dokumen produk *online shop* Blibli.com. Dengan demikian struktur kombinasi yang digunakan akan lebih representatif dan dapat meningkatkan kinerja pencarian Apache Solr pada situs Blibli.com.

DAFTAR PUSTAKA

- Adi T, Ewell OK, Adi P. 1999. High Selectivity and Accuracy with READWARE's Automated System of Knowledge Organization. *Management Information Technologies* [Internet]. [diunduh 2016 Mar 16]. 2(6):98 – 122.
- Baeza-Yates R, Ribeiro-Neto B. 1999. *Modern Information Retrieval*. England (UK): Addison Wesley.
- Clearly B *et al.* 2008. An empirical analysis of information retrieval based on concept location techniques in software comprehension. *Empirical Software Engineering Volume 14*. Number 193-130. doi: 10.1007/s10664-008-9095-3.
- Egozi O, Markovitch S, Gabrilovich E. 2011. Concept-based information retrieval using explicit semantic analysis. *Journal ACM Transactions on Information Systems (TOIS), TOIS Homepage* [Internet]. [diunduh 2016 Apr 2016]. 29(2):156 – 165. Tersedia pada www.cs.technion.ac.il/publications/papers/Egozi2011CBI.pdf.
- Goyal P, Behera L, McGinnity TM. 2009. An information retrieval model based on automatically learnt concept hierarchies. *IEEE International Conference. ICSC '09*. 2:1-4. doi: 10.1109/ITSIM.2008.4631693.
- Guarino N, Masolo C, Vetere G. 1999. OntoSeek: Using large linguistic ontologies for accessing on-line yellow pages and products catalogs. In *Information Extraction A Multidisciplinary Approach to an Emerging Information Technology* (pp. 139-170). Springer Berlin Heidelberg.
- Haav HM, Lubi TL. 2005. A survey of concept-based information retrieval tools on the web. *Institute of Cybernetics at Tallinn Technical University*. Academia Taee 21. 12618 Tallin.
- Krovetz R. 1997. Homonymy and polysemy in information retrieval. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics* (pp. 72-79). Association for Computational Linguistics.
- Manning CD, Raghavan P, Schütze H. 2008. *An Introduction to Information Retrieval*. Cambridge (UK): Cambridge Univ Pr.

- Munandar D. 2011. *E-Business*. Yogyakarta (ID): CV Andi Offset.
- Rad MP, Hassanpour H, dan Pourshaikh R. 2010. Concept-Based Information Retrieval with Ontology Approach for Cross-Language Searching. *World Applied Science Journal* [Internet]. [diunduh 2016 APR 24]. (8): 965-971. Tersedia pada www.cs.technion.ac.il/~ofere/papers/concept-based-ir-esa-tois11.pdf.
- Shahi D. 2015. *Apache Solr: A Practical Approach to Enterprise Search*. California (US): Apress.
- Smiley D, Pugh E. 2011. *Apache Solr 3 Enterprise Search Server*. Birmingham (UK): Packt Publishing Ltd.
- Sudirman S, Kodar A. 2012. Penggunaan model probabilistik untuk sistem temu kembali informasi. Di dalam: *Seminar Nasional Pengaplikasian Telematika (SINAPTIKA 2012)*; 2012 Jul 7; Jakarta, Indonesia. Jakarta (ID): SINAPTIKA. hlm 25-32.
- Tala FZ. 2003. A study of stemming effects on information retrieval in Bahasa Indonesia. *Institute for Logic, Language and Computation*. Netherlands (NL): Universiteit van Amsterdam.
- Woods WA. 1997. *Conceptual indexing: a better way to organize knowledge*. California (USA): Sun Microsystems Laboratories
- Zuliarso E, Februariyanti H, Utomo MS. 2010. Prototipe mesin pencari dokumen teks [skripsi]. Semarang (ID): Universitas Skitubank.

Lampiran 1 Daftar stopwords

<i>stopwords</i>			
acara	biasa	jelas	lewat
ada	biasanya	jenis	lima
adalah	bila	jika	luar
adanya	bisa	juga	maka
agar	bukan	jumat	mampu
akan	bulan	jumlah	mana
akhir	cara	juni	mantan
akhirnya	cukup	justru	masa
akibat	dalam	juta	masalah
aku	dan	kalau	masih
anda	dapat	kali	masing-masing
antara	dari	kami	masuk
apa	datang	kamis	mau
apakah	dekat	karena	maupun
apalagi	demikian	kata	melakukan
asal	dengan	katanya	melalui
atas	depan	ke	melihat
atau	di	kebutuhan	memang
awal	dia	kecil	membantu
badan	diduga	kedua	membawa
bagaimana	digunakan	kegiatan	memberi
bagi	dilakukan	kehidupan	memberikan
bagian	diri	kejadian	membuat
bahkan	dirinya	keluar	memiliki
bahwa	ditemukan	kembali	meminta
baik	dua	kemudian	mempunyai
banyak	dulu	kemungkinan	mencapai
barang	empat	kepada	mencari
barat	gedung	keputusan	mendapat
baru	hal	kerja	mendapatkan
bawah	hampir	kesempatan	menerima
beberapa	hanya	keterangan	mengaku
begitu	hari	ketiga	mengalami
belakang	harus	ketika	mengambil
belum	hasil	khusus	mengatakan
benar	hidup	kini	mengenai
bentuk	hingga	kita	mengetahui
berada	hubungan	kondisi	menggunakan
berarti	ia	kurang	menghadapi
berat	ikut	lagi	meningkatkan
berbagai	ingin	lain	menjadi
berdasarkan	ini	lainnya	menjalani
berjalan	itu	lalu	menjelaskan
berlangsung	jadi	lama	menunjukkan
bersama	jalan	langsung	menurut

Lampiran 1 Lanjutan

<i>stopwords</i>			
besar	persen	sejumlah	sejak
jauh	pertama	sekali	tentang
lebih	pihak	sekarang	tentu
menyebabkan	posisi	sekitar	terakhir
bertemu	program	selain	terhadap
jangan	proses	selalu	terjadi
lanjut	pula	selama	terkait
menyatakan	pun	selasa	terlalu
menyebutkan	punya	selatan	terlihat
merasa	rabu	seluruh	termasuk
mereka	rasa	semakin	ternyata
merupakan	ribu	sementara	tersebut
meski	ruang	sempat	terus
milik	saat	semua	terutama
minggu	sabtu	sendiri	tetapi
misalnya	saja	senin	the
mulai	salah	seorang	tidak
muncul	sama	seperti	tiga
mungkin	sampai	sering	tinggal
nama	sangat	serta	tinggi
namun	satu	sesuai	tingkat
nanti	saya	setelah	ujar
of	sebab	setiap	umum
oleh	sebagai	suatu	untuk
orang	sebagian	sudah	upaya
pada	sebanyak	sumber	usai
padahal	sebelum	tahu	utama
pagi	sebelumnya	tahun	utara
paling	sebenarnya	tak	waktu
panjang	sebesar	tampil	wib
para	sebuah	tanggal	ya
pasti	secara	tanpa	yaitu
pekan	sedang	tapi	yakni
penggunaan	sedangkan	telah	yang
penting	sedikit	teman]
perlu	segera	tempat	[
pernah	sehingga	tengah	


Lampiran 2 Koleksi Kueri (34 kueri)

Query	Jumlah dokumen yang relevan
baterai kamera canon	13
kamera dslr canon	29
sepatu boot black	15
baju batik keraton	25
dress batik katun	14
botol minum stainless 750 ml	31
aksesoris rambut ribbon	5
keripik balado	6
motor jupiter	7
jas hujan	60
gps	34
iphone 5s silver	18
mouse logitech wireless	46
keyboard razer	9
mesin kopi	52
tv led 32 inch	8
raket	59
dot bayi	11
wiper	36
anti virus	21
mp3 player	10
lampu tidur	6
pengering rambut	21
walkie talkie	8
ranjang bayi	48
monitor	45
stylus	29
kipas angin	17
lipstik	16
samsung galaxy note	134
sepatu kulit	33
xperia z	24
botol susu	73
iphone 5s	51

Lampiran 3 Desain antarmuka

A Daftar *file* yang pernah di-*upload*.

Services			
Run All-Evaluation Run All Upload File Field List History			
No	File Name	Time Stamp	Action
1	27-05-2016.csv	2016.07.24.10.40.13	▶
◀ 1 2 3 4 5 ▶			

Copyright © 2011 - 2016 PT. Global Digital Niaga. Powered by 

B *Form* untuk melakukan *upload file*

Services


Compare Results: demo golden answer.txt

[Run All-Evaluation](#)
[Run All](#)
[Upload File](#)
[Field List](#)
[History](#)

File Input

[Choose File](#) No file chosen

[Upload](#)

Copyright © 2011 - 2016 PT. Global Digital Niaga. Powered by 

C *Form* untuk melakukan modifikasi bobot struktur konsep

Services

[Run All-Evaluation](#)
[Run All](#)
[Upload File](#)
[Field List](#)
[History](#)

Precision-Recall graph

List Evaluation

Weight Configuration

brandSearch

nameSearch


categories

descriptionSearch

[Submit](#)

Once you click the button There is no way to go back :(

[Process The Query](#)

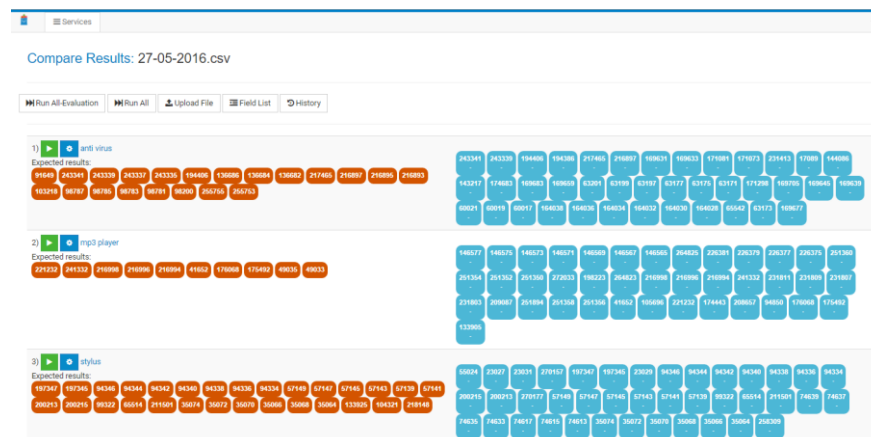
Copyright © 2011 - 2016 PT. Global Digital Niaga. Powered by 

Lampiran 3 lanjutan

D Halaman *evaluasi* 625 srtuktur konsep

List Evaluation													
No	Weight Parameter	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	Mean Average Precision
1	(brandSearch=2, nameSearch=4, categories=6, descriptionSearch=6)	0.96	0.92	0.88	0.85	0.81	0.66	0.56	0.50	0.44	0.36	0.3	0.888440520327427
2	(brandSearch=4, nameSearch=6, categories=10, descriptionSearch=10)	0.96	0.93	0.89	0.85	0.81	0.66	0.58	0.50	0.44	0.36	0.33	0.895299327787874
3	(brandSearch=4, nameSearch=6, categories=6, descriptionSearch=10)	0.94	0.91	0.86	0.83	0.82	0.64	0.56	0.55	0.45	0.35	0.32	0.8942434289542759
4	(brandSearch=4, nameSearch=4, categories=6, descriptionSearch=4)	0.95	0.92	0.87	0.85	0.81	0.66	0.55	0.50	0.44	0.36	0.3	0.799985672059968
5	(brandSearch=2, nameSearch=6, categories=10, descriptionSearch=10)	0.96	0.92	0.88	0.85	0.81	0.66	0.58	0.50	0.44	0.35	0.32	0.786751459054651
6	(brandSearch=4, nameSearch=6, categories=10, descriptionSearch=8)	0.96	0.93	0.87	0.84	0.81	0.66	0.59	0.49	0.43	0.38	0.35	0.781198321315165
7	(brandSearch=2, nameSearch=6, categories=6, descriptionSearch=10)	0.94	0.90	0.86	0.83	0.81	0.63	0.55	0.50	0.44	0.35	0.32	0.772376838080023
8	(brandSearch=6, nameSearch=4, categories=6, descriptionSearch=4)	0.95	0.92	0.87	0.85	0.81	0.66	0.55	0.50	0.44	0.36	0.33	0.7660818320135064
9	(brandSearch=6, nameSearch=6, categories=10, descriptionSearch=10)	0.95	0.92	0.87	0.85	0.81	0.66	0.58	0.50	0.44	0.36	0.33	0.7655466648515154
10	(brandSearch=8, nameSearch=6, categories=10, descriptionSearch=10)	0.95	0.92	0.87	0.85	0.81	0.66	0.58	0.50	0.44	0.36	0.33	0.7655466648515154
11	(brandSearch=4, nameSearch=8, categories=10, descriptionSearch=8)	0.96	0.93	0.87	0.84	0.80	0.62	0.56	0.49	0.43	0.35	0.32	0.762179147467228
12	(brandSearch=2, nameSearch=6, categories=10, descriptionSearch=8)	0.96	0.92	0.87	0.84	0.81	0.66	0.59	0.49	0.43	0.37	0.34	0.759626241389394
13	(brandSearch=6, nameSearch=4, categories=6, descriptionSearch=6)	0.93	0.90	0.85	0.83	0.79	0.63	0.54	0.51	0.45	0.36	0.33	0.746606119275373
14	(brandSearch=6, nameSearch=6, categories=10, descriptionSearch=10)	0.93	0.90	0.85	0.83	0.81	0.63	0.56	0.50	0.44	0.35	0.32	0.7446025194602235
15	(brandSearch=2, nameSearch=6, categories=6, descriptionSearch=6)	0.93	0.89	0.86	0.83	0.82	0.63	0.54	0.51	0.45	0.36	0.33	0.7402524761931674
16	(brandSearch=2, nameSearch=8, categories=10, descriptionSearch=8)	0.96	0.92	0.87	0.84	0.80	0.62	0.55	0.48	0.42	0.35	0.32	0.7393840377190606
17	(brandSearch=6, nameSearch=6, categories=10, descriptionSearch=8)	0.95	0.92	0.87	0.84	0.81	0.66	0.59	0.49	0.43	0.38	0.35	0.736407385738425
18	(brandSearch=10, nameSearch=6, categories=10, descriptionSearch=10)	0.95	0.92	0.85	0.85	0.81	0.66	0.58	0.50	0.44	0.36	0.33	0.733771398429507
19	(brandSearch=8, nameSearch=4, categories=6, descriptionSearch=6)	0.95	0.92	0.87	0.84	0.81	0.65	0.54	0.49	0.43	0.35	0.32	0.7391315648438387
20	(brandSearch=10, nameSearch=4, categories=6, descriptionSearch=6)	0.95	0.92	0.87	0.84	0.81	0.65	0.54	0.49	0.43	0.35	0.32	0.7391315648438387

E Halaman informasi detail *file* yang di-*upload*



F Halaman *evaluasi recall-precision* tiap kueri

Precision-Recall graph

Nilai bobot pada tiap Field

No	Field	Weight
1	brandSearch	4
2	nameSearch	6
3	categories	10
4	descriptionSearch	8

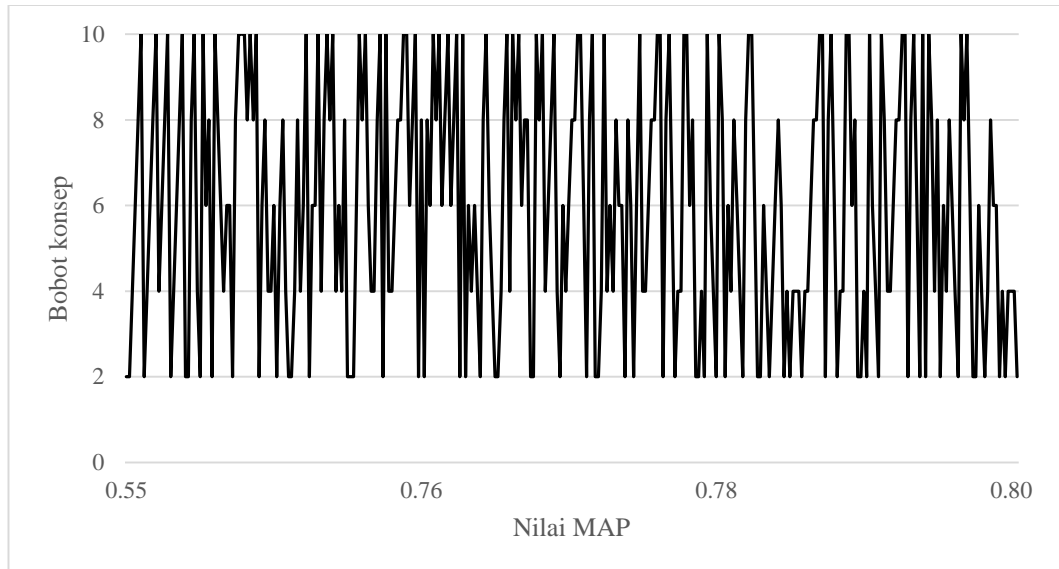
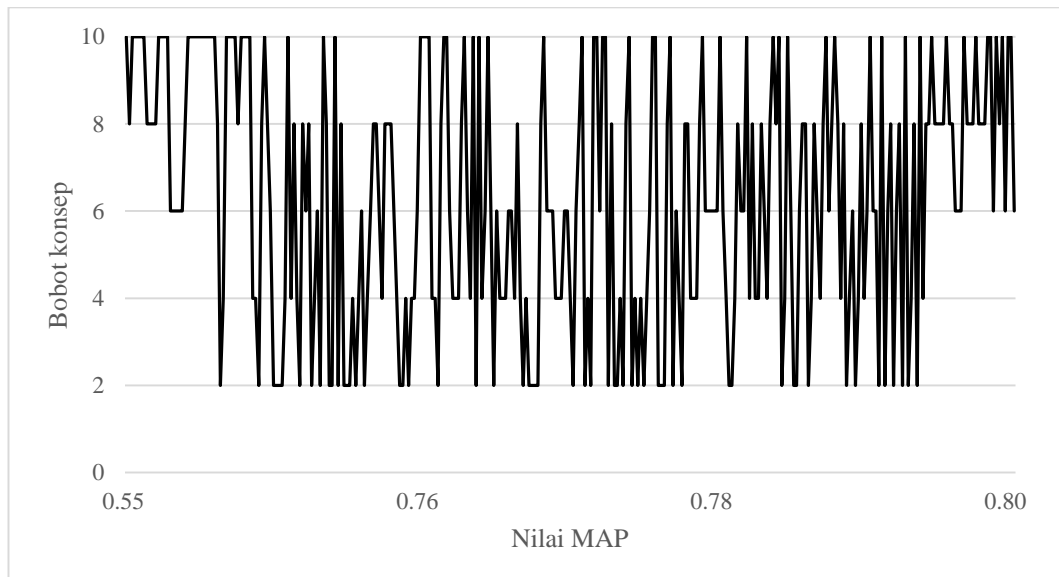
Nilai precision pada Eleven Standard Recall

No	Query	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
1	ars virus	1.0	1.0	0.83	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	mp3 player	0.17	0.17	0.17	0.17	0.17	0.0	0.0	0.0	0.0	0.0	0.0
3	stylus	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.0	0.0	0.0	0.0
4	samsung galaxy note	1.0	0.95	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	botol minum stainless 750 ml	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0
6	motor jupiter	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.87	0.87

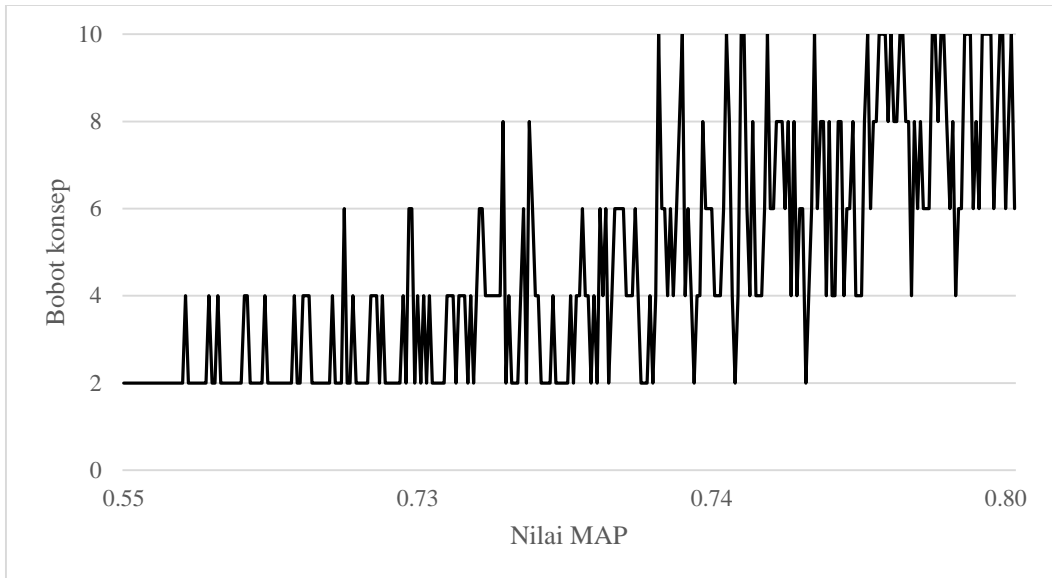
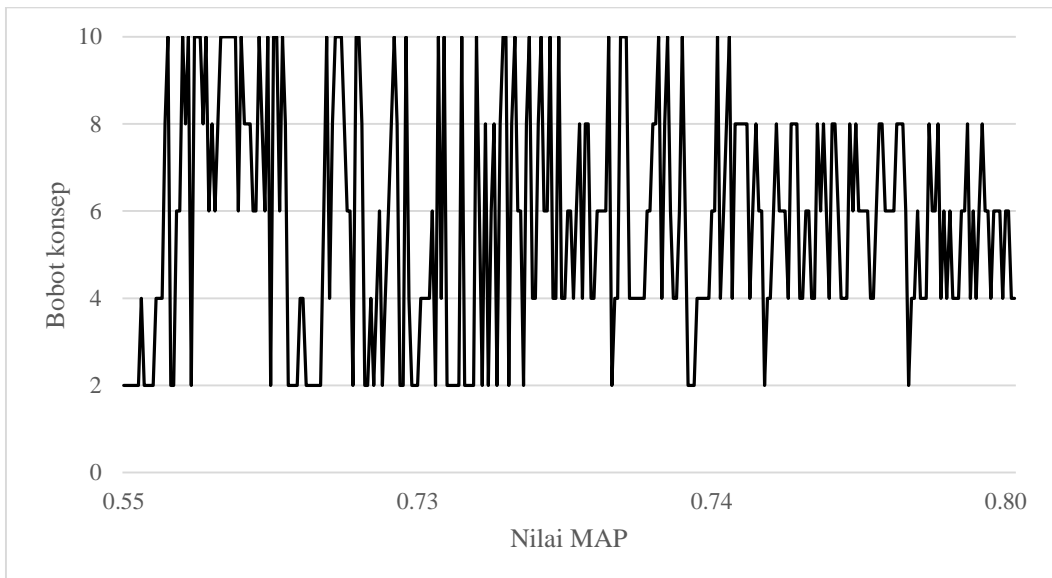
Lampiran 4 *Precision* tiap query

Query	Nilai <i>precision</i> pada <i>eleven standard recall</i>										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
anti virus	1	1	0.83	0	0	0	0	0	0	0	1
mouse logitech wireless	0.17	0.17	0.17	0.17	0.17	0	0	0	0	0	0.17
dot bayi	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0	0	0.75
gps	1	1	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	1
tv led 32 inch	1	1	1	1	1	1	1	1	1	1	1
keyboard razer	1	1	1	1	1	1	1	1	1	0.87	1
botol minum stainless 750 ml	1	1	1	1	0.58	0.58	0	0	0	0	1
motor jupiter	1	1	1	1	1	1	1	1	1	1	1
keripik balado	1	1	1	1	1	1	1	1	1	1	1
wiper	1	1	1	1	1	1	1	1	1	1	1
baju batik keraton	1	1	1	1	1	1	0.93	0.93	0.93	0.93	1
sepatu boot black	1	1	1	1	1	1	1	1	1	1	1
kamera dslr canon	1	1	1	1	1	1	1	1	1	0.33	1
jas hujan	1	1	1	1	1	1	1	1	1	1	1
mesin kopi	1	1	1	1	1	1	1	0.95	0	0	1
dress batik katun	1	1	1	1	1	1	1	1	1	1	1
baterai kamera canon	1	1	0.81	0.81	0.81	0.81	0.8	0.8	0.8	0.8	1
aksesoris rambut ribbon	1	1	0.83	0	0	0	0	0	0	0	1
iphone 5s silver	1	1	0.7	0.7	0.7	0.7	0.7	0.57	0.57	0.57	1
raket	1	1	1	1	1	1	1	1	1	0	1
11-interpolated precision	0.99	0.99	0.97	0.92	0.92	0.92	0.92	0.92	0.92	0.86	0.86
Mean Average Precision	0.80										

Lampiran 5 Grafik bobot konsep berdasarkan MAP

a. Grafik bobot konsep *brandSearch*b. Grafik konsep *descriptionSearch*

Lampiran 5 lanjutan

c. Grafik konsep *categories*d. Grafik konsep *nameSearch*

RIWAYAT HIDUP

Penulis dilahirkan di Purwokerto pada tanggal 03 Februari 1994 sebagai anak pertama dari pasangan Susilo Utomo.SE.MSI dan Titin Astuti SE.MSI. Pendidikan sekolah menengah atas ditempuh di SMAN 2 CIBINONG, lulus pada tahun 2012. Pada tahun 2012, penulis diterima menjadi mahasiswa Departemen Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Institut Pertanian Bogor melalui jalur Seleksi Nasional Masuk Perguruan Tinggi Negeri (SNMPTN) Tulis. Semasa studi di IPB, penulis aktif dalam kepengurusan Himpunan Mahasiswa Ilmu Komputer (Himalkom) tahun 2013-2014.