PENCARIAN BERITA BAHASA INDONESIA MENGGUNAKAN METODE *GENERALIZED VECTOR SPACE MODEL* (GVSM)

Syaiful Huda[1)], Joan Santoso[2)]
[1)]Universitas Nurul Jadid
Karanganyar Paiton Probolinggo
[2)]Institut Sains dan Teknologi Terpadu Surabaya
Baratajaya Gubeng Surabaya
e-mail: huda@unuja.ac.id[1)], joan@stts.edu[2)]

ABSTRAK

Berita bisa dating atau diperoleh dari mana saja, semisal dari teman, guru, atau media elektronik seperti mesin-pencari. Salah satu metode yang dapat digunakan untuk membangun mesin pencari adalah *Vector Space Model* (VSM). Masalah yang muncul adalah terdapat dokumen yang tidak ditemukan padahal mengandung istilah yang berkaitan dengan *query*. Berdasarkan permasalahan ini diperlukannya sebuah metode yang lebih menyeluruh dalam melakukan pencarian yang tidak hanya terpaku pada ada tidaknya suatu istilah di dalam dokumen. Untuk itu dipilihlah metode GVSM yang diharapkan mampu mengatasi masalah tersebut. Metode *Generalized Vector Space Model* (GVSM) adalah pengembangan dari VSM yang menambahkan hubungan antar istilah (*Semantic Relatedness*) dalam melakukan penghitungan kesamaan antara vektor *query* dengan vektor dokumen. Dengan-memperhitungkan relasi antar istilah maka pencarian sebuah dokumen akan lebih luas. Berdasarkan hasil uji coba yang telah dilakukan maka dapat disimpulkan bahwa penerapan metode GVSM belum mampu meningkatkan hasil pencarian berita Bahasa Indonesia dibandingkan dengan metode VSM. Dikarenakan penerapan metode GVSM pada sistem hanya mampu meningkatkan *recall* dan *accuracy* saja dengan persentase peningkatan masing-masing sebesar 30% dan 0.16%. Sedangkan *precision* memiliki nilai yang lebih rendah 11,17% dari pada metode VSM.

Kata Kunci: Berita, *Generalized Vector Space Model*, Pencarian, *Semantic Relatedness*.

ABSTRACT

News can come or be obtained from anywhere, for example from friends, teachers, or electronic media such as search engines. One of the methods that can be used to build a search engine is the Vector Space Model (VSM). The problem that arises is that there are documents that cannot be found even though they contain terms related to the query. Based on this problem, a more comprehensive method of searching is needed that is not only fixed on the presence or absence of a term in the document. For this reason, the GVSM method was chosen which is expected to be able to overcome this problem. The Generalized Vector Space Model (GVSM) method is a development of VSM which adds a relationship between terms (Semantic Relatedness) in calculating the similarity between query vectors and document vectors. By taking into account the relationship between terms, the search for a document will be broader. Based on the results of the trials that have been carried out, it can be concluded that the application of the GVSM method has not been able to improve Indonesian news search results compared to the VSM method. This is because the application of the GVSM method to the system can only increase the recall and accuracy levels with an increase of 30% and 0.16%, respectively. While the precision has a lower value of 11.17% than the VSM method.

***Keywords**: Generalized Vector Space Model, News, Searching, Semantic Relatedness.*

I. PENDAHULUAN

INFORMASI adalah kabar atau berita tentang sesuatu yang siapa pun sewaktu-waktu membutuhkannya[19]. Berita bisa datang atau diperoleh dari mana saja, semisal dari teman, guru, atau media elektronik. Di masa seperti saat ini, internet adalah salah satu media yang dapat digunakan untuk mencari berita.

Berita di internet tersebar di banyak situs, sehingga siapa pun yang ingin mencari berita tentang sesuatu, setidaknya tahu situs mana yang harus dikunjungi. Jika pun seseorang tahu akan suatu situs, bukan kah ia juga perlu mencari di mana atau pada halaman mana berita yang diinginkan bisa ia dapatkan. Sepertinya bukan hal yang mudah untuk mencari berita yang tersebar di internet.

Mesin pencari hadir untuk menjawab permasalahan tersebut. Cukup dengan menuliskan satu atau beberapa kata *(query)*, mesin akan mencari untuk kemudian menyajikan hasil pencariannya kepada kita.

Penggunaan *query* serta penulisan yang benar akan mempengaruhi terhadap hasil pencarian.

Salah satu metode yang dapat digunakan dalam membangun mesin pencari adalah *Vector Space Model* (VSM) yang merepresentasikan dokumen ke dalam ruang vektor untuk menghitung kesesuaian *query* dengan koleksi dokumen atau sumber informasi (dapat juga berupa situs di internet). Metode ini termasuk dari metode yang sangat terkenal di kalangan para peneliti. Kesesuaian antara *query* dan dokumen dihitung berdasarkan jarak vektor antara *term query* dan kumpulan *term* yang terdapat pada dokumen tersebut. Selanjutnya akan diurut berdasarkan jarak yang paling dekat hingga yang paling jauh.

Sebuah dokumen dianggap relevan ketika dokumen tersebut memiliki kata yang sama dengan kata yang ada di dalam *query*. Jika terdapat dokumen yang tidak memiliki satu kata pun yang sama dengan kata yang terdapat pada *query* maka dokumen tersebut dianggap tidak relevan. Hal itu juga berlaku pada proses pengurutan hasil pencarian dari yang paling dianggap relevan.

Sebagai contoh pencarian menggunakan *query* "siswa", tentunya mesin akan mencari dokumen yang mengandung kata "siswa". Jika terdapat dokumen yang tidak mengandung kata "siswa" maka tidak akan ditampilkan pada hasil pencarian.

Dari contoh tersebut, bagaimana jika terdapat sebuah dokumen yang memang tidak mengandung kata "siswa", namun di dalamnya terdapat kata "murid", "pelajar", atau "peserta didik". Bukankah dokumen tersebut juga relevan jika dilihat dari segi maknanya. Pada Kamus Besar Bahasa Indonesia, arti kata siswa adalah murid (terutama pada tingkat sekolah dasar dan menengah); pelajar[20].

Berdasarkan permasalahan tersebut diperlukannya sebuah metode yang lebih menyeluruh dalam melakukan pencarian yang tidak hanya berpaku pada ada tidaknya suatu *term* tertentu. Metode *Generalized Vector Space Model* (GVSM) adalah pengembangan dari VSM yang menambahkan hubungan antar *term* (*Semantic Relatedness*) dalam melakukan perhitungan jarak vektor sehingga berpengaruh pada hasil pencarian.

II. TEORI PENUNJANG

A. Using a Weighted Semantic Network for Lexical Semantic Relatedness[25]

Dalam penelitian ini dijelaskan bahwa *lexical semantic relatedness* adalah pengukuran bagaimana dua kata saling terkait dalam makna. Banyak aplikasi pengolahan bahasa alami seperti *textual entailment*, *question answering*, atau *information retrieval* memerlukan pengukuran yang kuat tentang keterkaitan semantik.

Berdasarkan masalah tersebut, peneliti membuat sebuah pendekatan yang dalam hal ini dapat dikategorikan kedalam tiga kategori utama, yaitu: teori yang mengandalkan *lexicon* dan strukturnya, pendekatan yang menggunakan hipotesis distributif pada *corpus* besar, dan pendekatan *hybrid*. Peneliti fokus pada pendekatan berbasis *lexicon* untuk mengukur keterkaitan semantik yang didasarkan pada jaringan semantik terberat yang mencakup 26 hubungan semantik yang ditemukan di *WordNet* selain informasi yang ditemukan di dalam *gloss*.

Dengan mengklasifikasikan hubungan *WordNet* kedalam beberapa kelas, peneliti dapat memberi bobot kontribusi sebuah relasi berdasarkan kelas yang dimilikinya, berlawanan dengan memberikan bobot kontribusi pada setiap hubungan. Tabel 1 menunjukkan tujuh kategori semantik yang telah didefinisikan. Bobot 1 ditugaskan sebagai kelipatan dari nilai kecil α, yang merupakan bobot terendah, dan penambahan 2 untuk setiap pengganda dalam daftar mewakili nilai yang lebih tinggi dari kategori yang kurang terkait.

Tabel 1
Tabel Kategori Hubungan dan Bobot yang Sesuai[25]

Category	***Weight***	***Semantic Relations in WordNet***
Similar	A	*antonym, cause, entailment, participle of verb, pertainym, similar to, verb group*
Hypernym	α × 2	*derivationally related, instance hypernym, hypernym*
Sense	4 ×α + β	*lemma-synset*
Gloss	6 ×α	*lemma-gloss content words*
Part	8 ×α	*holonym (part, member, substance), inverse gloss, meronym (part, member, substance)*
Instance	10 ×α	*instance hyponym, hyponym*
Other	12 ×α	*also see, attribute, domain of synset (topic, region, usage), member of this domain (topic, region, usage)*

Untuk menghitung *semantic relatedness* antar kata digunakan rumus berikut[25]:

$$S(w_1, w_2) = max\left(\frac{M - (P_{min}(w_1,w_2) - K}{M}, \frac{M - (P_{min}(w_2,w_1) - K}{M}\right) \ldots\ldots(1)$$

Nilai terendah dari w_1 sampai w_2, mungkin berbeda dari w_2 ke w_1, sehingga perlu ditentukan arah mana yang akan digunakan dengan cara mengambil nilai terendah antara keduanya. M adalah konstanta yang mewakili bobot setelah dua kata dianggap tidak terkait, dan K adalah konstan yang mewakili bobot sinonim sejati. Dalam implementasinya, peneliti menetapkan:

$$M = 2 \times (12 \times \alpha) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2)$$

$$K = 2 \times (4 \times \alpha) \quad \text{.....(3)}$$

B. Generalized Vector Space Model for Text Retrieval Based on Semantic Relatednes[26]

Dalam penelitian ini disebutkan bahwa penggunaan informasi semantik atau pengelompokan teks adalah sesuatu yang kontroversial. Penelitian yang dilakukan oleh Fellbaum (1998) menunjukkan penggunaan GVSM yang dipadukan dengan *WordNet* dapat meningkatkan pengelompokan teks terutama untuk rangkaian pelatihan kecil. Sebaliknya Sanderson (1994) melaporkan bahwa keakuratan *Word Sense Disambiguation* (WSD) yang mencapai 90% tidak dapat menjamin peningkatan *retrieval*. Demikian pula, Voorhees (1993) melaporkan bahwa pencarian yang didasarkan pada informasi WSD menurunkan kinerja *retrieval*. Sebaliknya, pembangunan model pencarian menggunakan *sense-based retrieval* oleh Stokoedkk. (2003) menyatakan dapat meningkatkan kinerja pencarian, sementara beberapa tahun sebelumnya, Krovetz dan Croft (1992) telah menunjukkan bahwa melibatkan arti kata dapat memperbaiki proses pencarian.

Adapun ukuran kesamaan antara dokumen dan *query* yang digunakan oleh peneliti ini ditunjukkan pada rumus 2.4. Dimana $\vec{t}_i$ dan $\vec{t}_j$ adalah istilah vektor dalam ruang vektor dimensi kedua, $\vec{d}_k$, $\vec{q}$ adalah vektor dokumen dan *query*, seperti sebelumnya, $\acute{a}_{ki}$, $\acute{q}_j$ adalah bobot baru, dan $\acute{n}$ dimensi ruang baru[26].

$$cos(\vec{d}_k, \vec{q}) = \frac{\sum_{j=1}^{\acute{n}} \sum_{i=1}^{\acute{n}} \acute{a}_{ki} \acute{q}_j \vec{t}_i \vec{t}_j}{\sqrt{\sum_{i=1}^{\acute{n}} \acute{a_{ki}}^2} \sum_{j=1}^{\acute{n}} \acute{q}_j^2} \text{.....(4)}$$

Dari persamaan di atas, istilah vector $\vec{t}_i$ dan $\vec{t}_j$ tidak perlu diketahui, asalkan korelasi antara istilah t_i dan t_j diketahui. Korelasi yang dimaksudkan didapat dari perhitungan *semantic relatedness* menggunakan database leksikal "*WordNet*".

C. A Study on the Semantic Relatedness of Query and Document Terms in Information Retrieval[23]

Peneliti mengungkapkan bahwa saat ini kita menghadapi pertumbuhan jumlah dokumen elektronik yang semakin pesat di semua area kehidupan. Hal ini menuntut cara yang lebih efektif dan efisien dalam melakukan pencarian pada beberapa dokumen untuk mendapatkan informasi. Terutama konten yang dihosting di web adalah sumber data berjumlah besar yang menimbulkan kesulitan khusus bagi IR.

Kata-kata yang tepat sering kali sulit untuk diprediksi dan sistem pencarian informasi (IR) saat ini didasarkan pada asumsi bahwa makna dokumen dapat disimpulkan dari kejadian atau tidak adanya istilah di dalamnya. Untuk menghasilkan kinerja pengambilan yang baik, yaitu, mengambil semua dokumen yang relevan tanpa mengambil dokumen yang tidak relevan, *query* harus dirumuskan oleh pengguna dengan cara yang tepat.

III. ARSITEKTUR SISTEM

A. Lucene Core

Lucene Core merupakan salah satu pustaka perangkat lunak sumber terbuka yang dikembangkan oleh Proyek *Apache Lucene*™ selain *Solr*. *Apache Lucene*™ adalah salah satu merek dagang yang dimiliki oleh *Apache Software Foundation*.

Pustaka ini dikembangkan sebagai perangkat lunak pencarian. *Lucene Core* adalah pustaka Java yang menyediakan fitur pengindeksan dan pencarian, pemeriksaan ejaan, penyorotan hasil temuan, serta kemampuan analisis atau tokenisasi tingkat lanjut. Selain tersedia dalama bahasa pemrograman Java, *Lucene Core* juga tersedia dalam bahasa pemrograman Python, yaitu sub proyek yang dinamakan *PyLucene* (*binding* Python untuk *Lucene Core*)[3].

B. Jsoup

Jsoup adalah pustaka Java yang dapat digunakan pada aplikasi yang menangani input berupa HTML. Merupakan proyek sumber terbuka yang didistribusikan di bawah lisensi MIT serta *source code* yang tersedia di *GitHub*.

Pustaka ini menyediakan API yang sangat mudah untuk mengambil URL dan mengekstrak serta memanipulasi data, menggunakan metode DOM HTML5 dan dilengkapi dengan penanganan *CSS selectors*[9]. *Jsoup* bekerja selayaknya browser modern yaitu mengurai HTML ke DOM serta mengimplementasikan spesifikasi WHATWG HTML5.

C. Kateglo

Kateglo adalah situs gratis dan terbuka pertama di Indonesia yang menyatukan kamus, tesaurus, dan peristilahan (*glosarium*) yang dibuat oleh Ivan (penggiat Wikipedia Indonesia) dan Romi (programmer yang menempuh studinya di Bremen, Jerman). Situs ini dikembangkan menggunakan PHP dan MySQL. Beberapa pihak yang terlibat dalam lahirnya situs diantaranya seperti Bahtera, mailing list untuk diskusi mengenai Bahasa Indonesia dan penerjemahan Bahasa Indonesia; Femmy Syahrani, blogger dan penerjemah lepas yang bermukim di Bandung; dan Steven Haryanto, salah satu pendiri perusahaan jasa web hosting PT Master Web Network[12].

Namanya diambil dari akronim unsur layanannya:

ka(mus), te(saurus), dan glo(sarium)[10]. *Kateglo* berisi 72.253 entri kamus, 191200 entri glosarium, 2012 entri peribahasa, serta 3423 entri singkatan dan akronim. *Kateglo* dapat digunakan secara manual, user dapat memasukan kata kunci yang dibutuhkan. *Kateglo* juga dapat digunakan untuk aplikasi dengan mengeluarkan keluaran dalam fomat JSON atau XML[11].

Secara umum alur proses yang digunakan pada sistem adalah sebagai berikut.

Gambar 1. Alur Proses Sistem

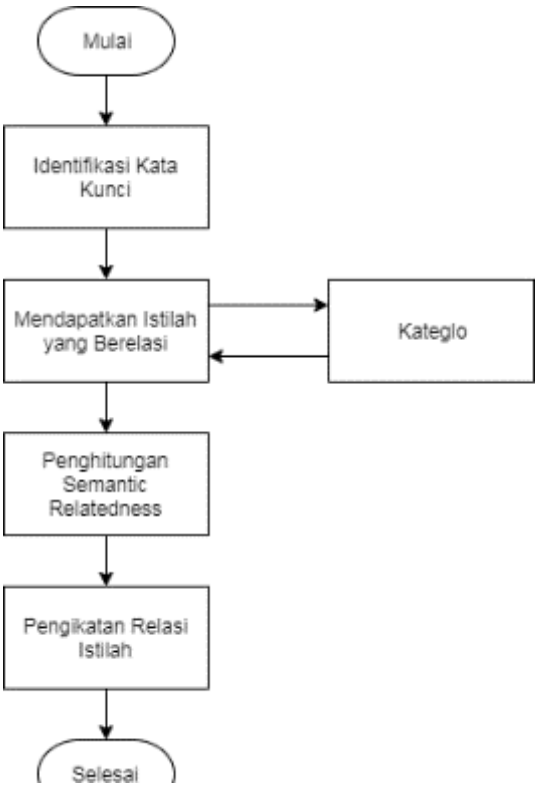
Kata kunci (*query*) adalah data yang dijadikan sebagai rujukan utama oleh sistem untuk menampilkan berita kepada pengguna. Kata kunci yang digunakan dalam penelitian ini dapat terdiri dari satu kata atau lebih. Kata kunci akan dipilih secara acak mengingat sumber berita yang digunakan pada penelitian ini adalah news.detik.com dengan 5.000 berita, tentunya akan banyak kata yang terkandung di dalamnya.

Pemilihan kata tentunya tidak akan terlepas dari ketentuan yang dapat digunakan sebagai kata kunci pada penelitian ini. Adapun kriteria-kriteria kata kunci yang dimaksud adalah sebagai berikut:

1. Bahasa yang digunakan adalah Bahasa Indonesia.
2. Penulisan kata kunci harus sesuai dengan kaidah penulisan Bahasa Indonesia.

IV. PENGAMBILAN ISTILAH YANG BERELASI

Secara umum proses ini akan melibatkan empat tahapan, pertama ialah mengidentifikasi kata kunci, kemudian mengambil istilah yang berelasi sesuai keluaran dari proses sebelumnya. Lalu menghitung semantik antar kata dan selanjutnya adalah mengikat istilah yang berelasi tersebut dengan kata kunci.



Gambar 2.
Alur Proses Pengambilan Istilah yang Berelasi

A. Identifikasi Kata Kunci

Permintaan istilah yang berelasi dengan kata kunci oleh sistem ke *Kateglo* sebanding dengan banyaknya jumlah kata yang terkandung di dalam kata kunci. Semisal jumlah kata adalah dua buah maka sistem akan melakukan permintaan terhadap *Kateglo* sebanyak dua kali. Semakin banyak jumlah kata yang terdapat di dalam kata kunci, maka akan semakin lama proses yang berjalan.

Agar sistem dapat mengetahui jumlah kata yang terkandung di dalam kata kunci maka masukan yang berupa *string* akan dipisah berdasarkan karakter spasi. Untuk meningkatan keakuratan pengidentifikasian jumlah kata maka spasi ganda akan diubah menjadi spasi tunggal serta spasi di awal dan akhir string akan dihapus. Termasuk kata yang berada di dalam daftar *stop words* (seperti: yang, di, dan lain sebagainya) serta karakter yang tidak termasuk dari sebuah kata juga akan dihapus (seperti: *, %, 1, dan lain sebainya).

B. Mendapatkan Istilah yang Berelasi

Setiap kali sistem menerima respon keluaran dari *Kateglo*, maka sistem akan memeriksa apakah format yang diterima benar-benar berupa XML. *Library Jsoup* yang digunakan oleh sistem dalam hal ini akan memberikan hasil berupa *string* sehingga pengecekan apakah format data berupa XML dilakukan dengan

cara memeriksa tag XML yang biasanya diawali dengan "<?xml". Dari dokumen XML inilah sistem akan mengurai untuk mendapatkan istilah yang berelasi yang berada pada *tag relation* di dokumen tersebut.

C. Penghitungan *Semantic Relatedness*

Dari beberapa tipe relasi yang diberikan oleh *Kateglo*, sistem hanya akan mengambil istilah yang memiliki relasi sinonim. Hal ini dilakukan dengan harapan agar sistem selain menyajikan hasil yang lebih banyak dari pada model *VSM* juga menjaga agar berita yang ditemukan tetap relevan dengan kata kunci yang diberikan oleh pengguna.

Untuk mendapatkan nilai hubungan *semantic relatedness* antara kata kunci dan istilah yang berelasi, sistem akan berbasis pada penghitungan yang dilakukan oleh Reda Siblini dan Leila Kosseim dengan makalah yang berjudul "*Using a Weighted Semantic Network for Lexical Semantic Relatedness*". Dari tujuh kategori hubungan dan bobot yang sesuai pada penelitian tersebut, hanya akan dipilih satu kategori oleh sistem yaitu *Sense* $(4 \times \alpha + \beta)$.

Gambar 3.
Nilai *Semantic Relatedness* Istilah Pameran dan Pertunjukan

Berdasarkan penelitian sebelumnya nilai α adalah 0.25 dan β adalah 0.01 maka bobot antara istilah pameran dengan set sinonim adalah yaitu $(4 \times \alpha + \beta) = 1.01$. Begitu juga bobot antara istilah pertunjukan dengan set sinonim memiliki nilai yang sama. Sehingga jumlah bobot dari istilah pameran ke istilah pertunjukan bernilai 1.01 + 1.01 = 2.02.

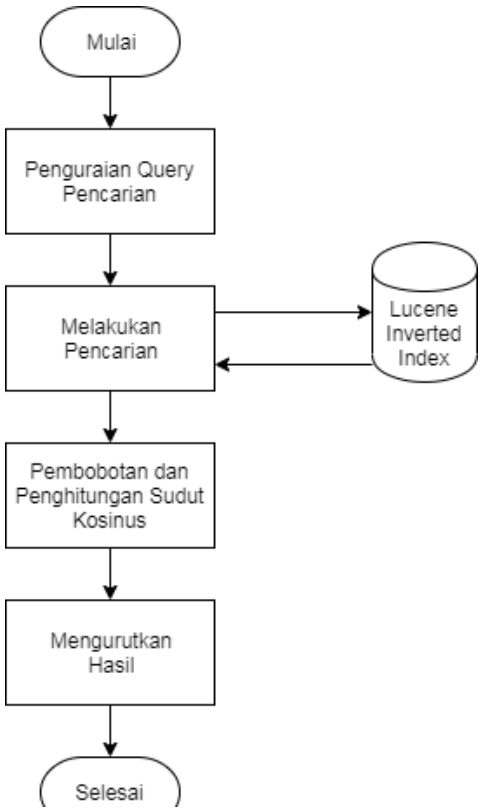
Dikarenakan bobot antara istilah pameran ke pertunjukan dan pertunjukan ke pameran memiliki nilai yang sama, maka dipilih salah satunya dalam hal ini jarak antara istilah pameran ke pertunjukan yang dipilih. Sehingga nilai *semantic relatedness* dari kata pertunjukan dengan $M = 2 \times (12 \times \alpha) = 6$ dan $K = 2 \times (4 \times \alpha) = 2$ berdsarkan rumus 2.1 adalah $(6 - (2{,}02 - 2)) / 6 = 0{,}997$.

D. Pengikatan Relasi Istilah

Pengikatan yang dimaksud di sini adalah istilah yang memiliki relasi sinonim berdasarkan hasil keluaran yang diberikan oleh *Kateglo*, berikut dengan jenis relasi serta nilai *semantic relatedness* akan diikat dalam satu *array* dengan kata kuncinya. Seperti pada contoh kasus kata kunci pameran, pada akhirnya sistem tidak hanya melakukan pencarian untuk kata kunci tersebut, melainkan juga melakukan pencarian untuk seluruh kata yang berelasi dengan kata kunci tersebut. Untuk mempermudah penjelasan, selanjutnya *array* ini akan disebut dengan *array* kata kunci.

V. PENCARIAN DOKUMEN

Setelah data *array* kata kunci diperoleh maka proses selanjutnya ialah memperoleh data *string* kata kunci serta istilah yang berelasi dengannya. Kumpulan data *string* inilah yang akan menjadi *query* pencarian yang selanjutnya akan diurai sehingga dapat dijadikan bahan pada proses pencarian. Sedangkan nilai *semantic relatedness* serta jenis relasi akan dibutuhkan pada proses selanjutnya.



Gambar 4.
Alur Proses Pencarian Dokumen

A. Penguraian *Query* Pencarian

Query pencarian akan diproses menjadi *object* focus pencarian dan agar memenuhi jenis data pustaka *Lucene*. Adapun maksud dari *object* focus pencarian adalah dari keseluruhan properti yang dimiliki oleh berita, seperti id, judul, deskripsi, isi, waktu rilis, dan waktu pengindeksan berita, maka berdasarkan properti yang mana berita akan dicari. Dalam hal ini sistem telah menentukan data yang akan diindeks adalah judul, deskripsi, serta isi berita. Sehingga *object* focus pencarian yang dimaksud di sini adalah pada judul, deskripsi, dan isi berita.

Selainmenentukan object fokuspencarian, query pencarianakanmelaluitahappenguraianberdasarkanstan darpustaka Lucene-queryparser-8.5.2 sertatokeni-zing, stopword, dan stemming Bahasa Indonesia.

B. Melakukan Pencarian

Lucene secara asal mengurutkan dokumen berdasarkan skor pencarian. Tentunya hasil yang disaji-kan berdasarkan hasil penghitungan pustaka ini. Untuk itu agar sistem dapat mengimplementasikan seluruh penghitungannya maka opsi ini akan dimatikan. Sistem hanya meminta pustaka *Lucene* agar mengurutkan berdasarkan kapan berita itu dirilis sehingga tidak akan berpengaruh pada hasil penghitungan sistem.

C. Pembobotan dan Penghitungan *Cosine Similarity*

Pembobotan pada metode GVSM tidak jauh berbeda dengan pemrosesan pada VSM standar, perbedaannya hanya terletak pada adanya dimensi ruang baru yang jumlahnya sejalan dengan banyaknya kata kunci (yang lolos dari beberapa tahap pemeriksaan seperti telah dijelaskan sebelumnya) serta adanya nilai *semantic relatedness* dalam menentukan bobot baru.

Untuk mempermudah pemahaman bagaimana sistem menerapkan model GVSM serta perbedaannya dengan VSM maka akan disertakan contoh kasus berikut dengan *query* yang digunakan adalah "pameran" dan "mobil". Penentuan idf pada contoh di bawah ini menggunakan logaritma berbasis 10.

Table 2.
Contoh Pembobotan Model VSM

	q	d1	d2	d3	df	idf	wq	wd1	wd2	wd3
pameran (i1)	1	3	0	2	2	0,176	0,176	0,528	0	0,352
pertunjukan (i2)	0	0	3	0	1	0,477	0	0	1,431	0
mobil (i3)	1	2	0	0	1	0,477	0,477	0,954	0	0
otomobil (i4)	0	0	2	0	1	0,477	0	0	0,954	0
kuno (i5)	0	2	0	0	1	0,477	0	0,954	0	0
antik (i6)	0	0	0	2	1	0,477	0	0	0	0,954
barang (i7)	0	0	0	3	1	0,477	0	0	0	1,431

Untuk istilah pameran pada dokumen satu memiliki bobot 3 × 0,176 = 0,528 yang didapatkan dari hasil perkalian jumlah istilah dengan *invers* frekuensi istilah dokumen tersebut. Begitu seterusnya hingga istilah barang pada dokumen tiga.

Jika pada VSM standar bobot istilah dokumen didapatkan dari jumlah istilah dikalikan dengan *invers* frekuensi dokumen istilah tersebut, maka berbeda pada model GVSM yang melibatkan nilai *semantic relatedness* serta *invers* frekuensi dokumen istilah kata kunci per dimensi dan invers frekuensi dokumen istilah itu sendiri.

Tabel 3.
Contoh Penghitungan TF-IDF Model GVSM Dimensi Pertama

	q	Idf	tf-idf (ti,q)	tf-idf (ti,dk)			tf-idf (tj,q)	tf-idf(tj,dk)		
				d1	d2	d3		d1	d2	d3
i1	1	0,176	0,176	0,528	0	0,352	0,176	0,528	0	0,352
i2	1	0,477	0,176	0	0,528	0	0,477	0	1,431	0
i3	0	0,477	0	0,352	0	0	0,000	0,954	0	0
i4	0	0,477	0	0	0,352	0	0,000	0	0,954	0
i5	0	0,477	0	0,352	0	0	0,000	0,954	0	0
i6	0	0,477	0	0	0	0,352	0,000	0	0	0,954
i7	0	0,477	0	0	0	0,528	0,000	0	0	1,431

Dikarenakan contoh pada tabel di atas berfokus pada dimensi baru pertama yang terbentuk oleh istilah pameran maka ada tiga hal yang membedakan antara tabel 2. dan tabel 3., yaitu:

1. Istilah pertunjukan (i2) pada *query* akan bernilai satu dikarenakan adanya keterkaitan istilah dengan pameran (i1).
2. Istilah mobil (i3) bernilai nol karena kata kunci ini akan dihitung pada dimensi kedua.
3. Adanya penghitungan tf-idf ganda yang melibatkan idf istilah pembentuk dimensi baru dan idf dari istilah itu sendiri. Idf yang digunakan pada penghitungan nilai pada kolom $tf - idf(t_i, q)$ dan $tf - idf(t_i, d_k)$ adalah *invers* yang dimiliki oleh istilah pameran, sedangkan pada kolom $tf - idf(t_j, q)$ dan $tf - idf(t_j, d_k)$ adalah sama seperti penghitungan pada model VSM.

Langkah selanjutnya adalah menghitung pembobotan istilah pada dimensi pertama yang ditunjukkan pada tabel 4. Keterangan sr pada kolom menerangkan nilai *semantic relatedness* antar istilah, wq merupakan bobot *query*, sedangkan wd adalah bobot istilah pada masing-masing dokumen.

Tabel 4.
Contoh Pembobotan Model GVSM Dimensi Pertama

	sr	wq	wd1	wd2	wd3
i1	0	0,352	1,057	0	0,704
i2	0,997	0,651	0	1,953	0
i3	0	0	1,306	0	0
i4	0	0	0	1,306	0
i5	0	0	1,306	0	0
i6	0	0	0	0	1,306
i7	0	0	0	0	1,960

Mengacu pada pembahasan sebelummnya tentang ruang *vector* baru $d_k(t_i, t_j)$, maka bobot istilah pada *query* di dimensi pertama dapat diketahui dengan cara menjumlahkan kolom $tf - idf(t_i, q)$ dan $tf - idf(t_j, q)$, jika terdapat relasi antara istilah t_i dan t_j maka hasil penjumlahan tersebut akan dikalikan dengan nilai *semantic relatedness* di antara keduanya. Hal ini juga berlaku pada penentuan bobot istilah pada dokumen yaitu menjumlahkan nilai $tf - idf(t_i, d_k)$ dengan $tf - idf(t_j, d_k)$ dan mengalikannya kemudian jika terdapat relasi di antara keduanya.

Contoh istilah pameran (i1) pada *query* akan memiliki

bobot sebesar 0,176 + 0,176 = 0,352 dan istilah pertunjukan (i2) pada dokumen kedua akan memiliki bobot sebesar (0,528 + 1,431) × 0,997 =1,935. Pembobotan pada dimensi kedua memiliki tahapan yang sama dengan penghitungan pembobotan pada dimensi pertama. Yang berbeda adalah dimensi ini dibentuk oleh istilah mobil (i3).

D. Mengurutkan Hasil

Setelah nilai *cosine similiarity* antara dokumen dan *query* diketahui, maka sistem akan menyajikan hasil pencarian berita dengan mengurutkan nilai tersebut dari yang tertinggi hingga yang terendah. Jika terdapat nilai yang sama maka sistem akan mengurutkan secara acak namun tetap memperhatikan urutan nilai di atasnya atau pun di bawahnya untuk nilai yang tidak sama.

VI. UJI COBA DAN EVALUASI

Metode pengujian yang dilakukan pada penelitian ini terdiri dari tiga tahapan. Tahap pertamaya itu menghitung nilai *precision* untuk mengetahui kwalitas hasil pencarian. Selanjutnya melakukan penghitungan nilai *recall* untuk mengetahui kwantitas hasil pencarian. Pada tahap terakhir adalah menghitung nilai *accuracy* untuk mengetahui tingkat keakuratan sistem dalam menyajikan berita.

Nilai *precision* menunjukkan tingkat ketelitian, yaitu perbandingan antara kumpulan informasi yang ditemukan dengan jumlah informasi yang relevan pada kumpulan tersebut.

$$Precision = \frac{jumlah\ dokumen\ relevan\ yang\ ditemukan}{jumlah\ dokumen\ yang\ ditemukan} \quad .....(5)$$

$$Recall = \frac{jumlah\ dokumen\ relevan\ yang\ ditemukan}{jumlah\ dokumen\ relevan\ dalam\ koleksi} \quad .....(6)$$

$$Accuracy = \frac{jumlah\ dokumen\ relevan\ yang\ ditemukan + jumlah\ dokumen\ tidak\ relevan\ dalam\ koleksi}{jumlah\ dokumen\ dalam\ koleksi} \quad .....(7)$$

Sedangkan nilai *recall* menunjukkan tingkat kesempurnaan, yaitu perbandingan antara kumpulan informasi relevan yang ditemukan dengan jumlah keseluruhan informasi yang relevan di dalam sumber data. Nilai ketiganya akan berkisar pada nilai 0-1 yang selanjutnya akan diubah menjadi persentase.

Adapun target keberhasilan penggunaan metode GVSM (*Generalize Vector Space Model*) dari penelitian ini adalah dapat meningkatan hasil pencarian berita dibanding menggunakan metode VSM (*Vector Space Model*). Peningkatan hasil pencarian berita yang dimaksud adalah hasil pengujian metode GVSM memiliki nilai rata-rata *precision, recall*, dan *accuracy* yang lebih tinggi dari pada hasil pengujian metode VSM.

Adapun *query* yang akan digunakan sebanyak lima buah (6 istilah dari total 9197 yang telah diindeks oleh sistem) dengan rincianyang ditunjukkan pada tabel 5. sebagai berikut.

Tabel 5.
Daftar *Query* Pengujian

No	Query	Jml. Dok. Relvan Dalam Koleksi	Jml. Dok Tidak Relevan Dalam Koleksi
1	Imunisasi	3	497
2	Toleransi	2	498
3	Promosi	3	497
4	Pencurian	3	497
5	Pengendara Sepeda	12	488

Pemilihan *query* dilakukan secara acak dari daftar istilah yang memiliki frekuensi yang paling rendah. Opsi ini dipilih dengan tujuan mengetahui seberapa besar pengaruh penambahan nilai *semantic relatedness* pada istilah yang memiliki frekuensi yang rendah.

Seperti kita ketahui relevan atau tidaknya sebuah dokumen hasil pencarian bergantung pada penilaian dari masing-masing pengguna yang membutuhkan informasi. Khusus pada penelitian ini sebuah dokumen dianggap relevan atau tidak dipandang dari apakah sebuah *query* merupakan topik atau objek yang dibicarakan. Meskipun sebuah dokumen berita mengandung istilah yang terdapat di dalam *query* belum tentu dokumen tersebut dianggap relevan. Hasil uji coba disajikan secara berurutan dimulai dari jumlah berita serta id berita yang ditemukan dan dilanjutkan dengan penghitungan.

Tabel 6.
Hasil Pencarian Query Pengujian pada Metode VSM

Query	Waktu Eksekusi	Jml. Dok. yang Ditemukan	Jml. Dok. Relevan yang Ditemukan	ID Dok. Relevan	ID Dok. Tidak Relevan
Imunisasi	0,005 dtk	1	1	d5190490	-
Toleransi	0,008 dtk	2	1	d5190479	d5190034
Promosi	0,064 dtk	8	2	d5188961 d5189262	d5189144 d5188996 d5189018 d5190205 d5188980 d5189069
Pencurian	0,019 dtk	9	3	d5190327 d5190430 d5189835	d5189094 d5190215 d5190024 d5189066 d5188961 d5189280
PengendaraSepeda	0,106 dtk	21	12	d5190040 d5189776 d5190019 d5190393 d5189806 d5190382	d5189315 d5188993 d5189033 d5188997 d5190250 d5188858

				d5189498 d5190021 d5189822 d5190083 d5189380 d5189096	d5190034 d5189936 d5189476

Seperti tampak pada tabel hasil pencarian menggunakan metode VSM di atas bahwa dari lima *query* yang diberikan, masing-masing memiliki dokumen yang ditemukan. Lama tidaknya waktu eksekusi yang dibutuhkan tergantung dari jumlah dokumen dan total jumlah istilah dari seluruh dokumen. Jumlah dokumen saja tidak cukup untuk menentukan lama eksekusi karena pada *query* promosi dan pencurian yang masing-masing mendapatkan 8 dokumen, waktu eksekusinya berbeda. Hal ini disebabkan oleh jumlah istilah yang bervariasi pada setiap berita. Untuk seluruh pencarian yang dilakukan berdasarkan *query* pengujian, tidak ada satu pun yang membutuhkan waktu lebih dari 1 detik. Rata-rata proses pencarian memakan waktu 0,047 detik. Adapun nilai *recall*, *precision*, dan *accuracy* model VSM ditunjukkan pada tabel berikut.

Tabel 7.
Nilai *Recall, Precision*, dan *Accucary* Metode VSM

Query	***Recall***	***Precision***	***Accuracy***
Imunisasi	33,3%	100%	99,6%
Toleransi	50%	50%	99,8%
Promosi	66,67%	25%	99,8%
Pencurian	100%	33,33%	100%
Pengendara Sepeda	100%	57,14%	100%
Rata-rata	**70%**	**53,1%**	**99,84%**

Berdasarkan hasil pengujian pada tabel 7, metode VSM memiliki nilai rata-rata tertinggi pada *accuracy* sebesar 99,84%, diikuti dengan *recall* 70% dan yang terakhir adalah *precision* 53,1%.

Tabel 8.
Hasil Pencarian *Query* Pengujian pada Metode GVSM

Query	**Waktu Eksekusi**	**Jml. Dok. yang Ditemukan**	**Jml. Dok. Relevan yang Ditemukan**	**ID Dok. Relevan**	**ID Dok. Tidak Relevan**
Imunisasi	2 dtk	3	3	d5190490 d5190090 d5190019	-
Toleransi	2 dtk	8	2	d5190479 d5188867	d5190034 d5189220 d5189054 d5188895 d5188924 d5188890
Promosi	1 dtk	11	3	d5188961 d5189262 d5189072	d5189144 d5188996 d5189018 d5190205 d5188980 d5189069 d5190011 d5190509
Pencurian	2 dtk	13	3	d5190327	d5189094 d5190430 d5189835	d5190215 d5190024 d5189066 d5188961 d5189280 d5190390 d5190022 d5189955 d5189803
PengendaraSepeda	4 dtk	35	12	d5190040 d5189776 d5190019 d5190393 d5189806 d5190382 d5189498 d5190021 d5189822 d5190083 d5189380 d5189096	d5189315 d5188993 d5189033 d5188997 d5190250 d5188858 d5190034 d5189936 d5189476 d5189833 d5189218 d5188777 d5189661 d5189658 d5189434 d5188969 d5190024 d5189955 d5190147 d5189803 d5188967 d5188878 d5190516

Seperti tampak pada tabel hasil pencarian menggunakan metode GVSM di atas bahwa dari lima *query* yang diberikan, masing-masing memiliki dokumen yang ditemukan. Waktu eksekusi yang dibutuhkan cenderung lebih lama dari metode VSM. Hal ini disebabkan oleh jumlah dokumen (atau jumlah istilah yang terlibat dalam penghitungan) yang ditemukan lebih banyak dari pada metode VSM. Rata-rata proses pencarian memakan waktu 2,2 detik. Adapun nilai *recall*, *precision*, dan *accuracy* model GVSM ditunjukkan pada tabel berikut.

Tabel 9.
Nilai *Recall*, *Precision*, dan *Accucary* Metode GVSM

Query	***Recall***	***Precision***	***Accuracy***
Imunisasi	100%	100%	100%
Toleransi	100%	25%	100%
Promosi	100%	27,27%	100%
Pencurian	100%	23,04%	100%
Pengendara Sepeda	100%	34,29%	100%
Rata-rata	**100%**	**41,93%**	**100%**

Berdasarkan hasil pengujian pada tabel 9. di atas, metode GVSM memiliki nilai tertinggi pada *accuracy* sebesar 100% serta *recall* 100% dan yang terakhir adalah *precision* 41,93%. Berdasarkan nilai rata-rata dari masing-masing pengujian kedua metode tersebut, maka GVSM memiliki nilai *recall* dan *accuracy* yang lebih tinggi dari pada VSM dengan selesih masing-masing sebesar 30% $(100 - 70)$ dan 0,16% $(100 - 99{,}84)$. Sedangkan VSM memiliki nilai *precision* yang lebih tinggi sebesar 11,17% $(53{,}1 - 41{,}93)$.

VII. KESIMPULAN

Berdasarkan hasil penelitian dan pembahasan yang telah dilakukan, dapat disimpulkan bahwa metode *Generalized Vector Space Model* (GVSM) dapat meningkatan nilai *recall* dan *accuracy*, namun tidak dapat meningkatkan nilai *precision*. Hal ini telah dibuktikan pada pengujian sistem dengan peningkatan nilai *recall* sebesar 30% dan *accuracy* sebesar 0.16% serta nilai *precision* yang lebih rendah 11,17% dari pada metode *Vector Space Model* (VSM). Sehingga kesimpulan akhir dari penelitian ini adalah penerapan metode GVSM belum mampu meningkatkan hasil pencarian berita Bahasa Indonesia dibandingkan dengan metode VSM. Dikarenakan penerapan metode GVSM pada sistem hanya mampu meningkatkan kwantitas dan akurasi saja, sedangkan tingkat ketelitian tidak dapat ditingkatkan.

DAFTAR PUSTAKA

[1] Amin, Fatkhul, (2012). *"*Sistem Temu Kembali Informasi dengan Metode *Vector Space Model"*.Jurnal Sistem Informasi Bisnis 02 (2012).

[2] *Anonymous*, "*A First Take at Building an Inverted Index*", *https://nlp.stanford.edu/IR-book/html/htmledition/a-first-take-at-building-an-inverted-index-1.html*, diakses 13 November 2020.

[3] *Anonymous*, "*Apache Lucene*", *https://lucene.apache.org,* diakses 4 Agustus 2020.

[4] *Anonymous*, "*Apache Lucene* - Wikipedia", *https://en.wikipedia.org/wiki/Apache_Lucene*, diakses 4 Agustus 2020.

[5] *Anonymous*, "*Boolean Retrieval*", *https://nlp.stanford.edu/IR-book/pdf/01bool.pdf*, diakses 13 November 2020.

[6] *Anonymous*, "*Colophon*", *https://jsoup.org/ colophon*, diakses 15 Agustus 2020.

[7] *Anonymous*, "*Dot Product*", *https://nlp. stanford.edu/IR-book/html/htmledition/dot-products-1.html*, diakses 14 November 2020.

[8] *Anonymous*, "*Inverse Document Frequency*", *https://nlp.stanford.edu/IR-book/html/ htmledition/inverse-document-frequency-1.html*, diakses 14 November 2020.

[9] *Anonymous*, "*Jsoup*", *https://jsoup.org*, diakses 15 Agustus 2020.

[10] *Anonymous*, "*Kateglo*", *http://kateglo.com/?mod=doc&doc=README.txt*, diakses November 2017.

[11] *Anonymous*, "*Kateglo* – Onno Center Wiki", *https://lms.onnocenter.or.id/wiki/index.php/Kateglo*, diakses November 2017.

[12] *Anonymous*, "*Kateglo* - Tempo", *https://tekno.tempo.co/read/181924/kateglo-situs-pertama-penyatu-kamus-dan-tesaurus*, diakses November 2017.

[13] *Anonymous*, "*Search Engine Indexing*", *https://en.wikipedia.org/wiki/Search_engine_indexing*, diakses 15 November 2020.

[14] *Anonymous*, "*Text Preprocessing*", *https://www.codecademy.com/learn/natural-language-processing/modules/nlp-text-preprocessing*, diakses 13 November 2020.

[15] *Anonymous*, "*Term Frecuency and Weighting*", *https://nlp.stanford.edu/IR-book/html/htmledition/term-frequency-and-weighting-1.html*, diakses 14 November 2020.

[16] *Anonymous*, "*TF-IDF Weighting*", *https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html*, diakses 14 November 2020.

[17] Birger Hjorland, (2018). *"Indexing: Concepts and Theory"*, Knowl. Org. 45 (2018) No.7.

[18] Hidayat Huang, "Metode Penelitian Kuantitatif", *http://www.globalstatistik.com /metode-penelitian-kuantitatif*, diakses 18 Desember 2017.

[19] Kemdikbud, "KBBI Daring Informasi", *https://kbbi.kemdikbud.go.id/entri/informasi*, diakses November 2017.

[20] Kemdikbud, "KBBI Daring Siswa", *https://kbbi.kemdikbud.go.id/entri/siswa*, diakses November 2017.

[21] Kemdikbud, "KBBI Daring Informasi", *https://kbbi.kemdikbud.go.id/entri/informasi*, diakses November 2017.

[22] Ajit Kumar Mahapatra dan Sitanath Biswas, (2011). *"Inverted indexes: Types and techniques",* IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 4, No 1, July 2011.

[23] Christof Muller, dan Iryna Gurevych, (2009). *"A Study on the Semantic Relatedness of Query and Document Terms in Information Retrieval"*, Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing.

[24] Ahmad M Odat, (2015). *"Similarity Measurements of Vector Space Model on Arabic Text"*. Research Journal of Applied Sciences, Engineering and Technology 11(8): 860 864, 2015.

[25] Reda Siblini, dan Leila Kosseim, (2013). *"Using a Weighted Semantic Network for Lexical Semantic Relatedness"*. Proceedings of Recent Advances in Natural Language Processing.

[26] George Tsatsaronis dan Vicky Panagiotopoulou, (2009). *"Generalized Vector Space Model for Text Retrieval Based on Semantic Relatedness"*. EACL Student Research Workshop.