Sistem Penilaian Otomatis Jawaban Esai Dengan Menggunakan Metode Vector Space Model Pada Beberapa Perkuliahan Di Stmik Indonesia Banjarmasin

Ferdy Febriyanto
Jurusan Sistem Informasi, STMIK Indonesia, Banjarmasin
Jl. Pangeran Hidayatullah Benua Anyar Banjarmasin
Ferdy@gmail.com

INTISARI

Perkembangan sistem e-learning setiap tahunnya terus meningkat, hal ini dikarenakan sistem e-learning memberikan banyak kemudahan dalam pembelajaran. Beberapa institusi pendidikan khususnya perguruan tinggi negeri maupun swasta mulai mengembangkan sistem e-learning pada proses pengajarannya. Dalam konsep e-learning, pelaksanaan ujian dapat dilakukan, mulai dari menjawab soal ujian hingga proses penilaian selama ini kebanyakan proses ujian esai dan penilaiannya dilaksanakan secara manual yaitu dengan membaca esai satu per satu. Para dosen perlu menghabiskan banyak waktu untuk menilai jawaban ujian mahasiswa. Semakin banyak jumlah ujian yang dikoreksi, kualitas penilaian yang diberikan semakin menurun.

Untuk memecahkan masalah tersebut dapat dilakukan dengan membuat suatu aplikasi yang dapat memproses kemiripan teks. Oleh karena itu dalam penelitian tesis ini, penulis menggunakan algoritma TF/IDF (Term Frequency – Inversed Document Frequency) dan VSM (Vector Space Model) yang secara prosesnya dapat mencari nilai kemiripan dari suatu teks jawaban dengan teks kunci jawaban. Nilai kemiripan teks tersebut dapat dijadikan acuan sebagai nilai koreksi jawaban ujian mahasiswa.

Hasil penelitian menggunakan data dari Ujian Akhir Semester di STMIK Indonesia Banjarmasin dengan 10 mata kuliah, yaitu : Desain Grafis, Jaringan Komputer, Pengantar Teknologi Informasi, Kecakapan Antar Personal, Sistem Operasi, Pengantar Manajemen, Etika Profesi, Sistem Basis Data, Microprosessor, dan Pemrograman Web. Masing -masing mata kuliah diinputkan 30 soal dengan setiap soalnya memiliki 3 jawaban benar yang berbeda sebagai pembanding tingkat kemiripannya. Dalam prosesnya, sistem akan menghapus kata - kata yang dianggap tidak penting atau kata - kata yang terlalu umum digunakan termasuk karakter atau bentuk simbol, karena sistem hanya akan memproses soal yang memerlukan jawaban teoritis dan argumentasi bukan matematis. Untuk kasus pada penelitian tesis ini kata - kata dalam bahasa lokal Banjar juga akan dihilangkan oleh sistem untuk penyetaraan penggunaan bahasa Indonesia. Dengan kumpulan kata yang tersisa setelah proses penghilangan kata, perhitungan nilai bobot kata akan dilakukan algoritma TF/IDF dan dengan VSM akan dihitung nilai cosinus, sehingga didapatlah nilai tingkat kemiripan antara jawaban oleh mahasiswa dan jawaban oleh dosen. Tingkat kolerasi yang dihasilkan cukup baik dengan tingkat akurasi rata – rata 80% - 90% bila dibandingkan dengan penilaian yang dilakukan manusia secara manual.

Kata Kunci : *Penilaian Ujian Otomatis, TF/IDF, VSM, Similiaritas.*

ABSTRACT

The development of e-learning system every year keep on increased, this is because the e-learning system provides much convenience in learning. Some educational institutions, especially universities started to develop a system of e-learning in the teaching process. In the concept of e-learning, test execution can be carried out, started from answering the exam until this assessment process during most of the process of essay exams and assessments carried out manually, by reading essays one by one. The lecturers need to spend a lot of time to assess the student exam answers. The more of the number exam that corrected, quality assessment given decreased.

To solve these problems can be done by creating an application that can process text similarity. Therefore, in this thesis, the author uses an algorithm TF / IDF (Term Frequency - Inversed Document Frequency) and VSM (Vector Space Model) in the process can seek similarity value of a answer text with the text of the answer key. The value of text similarity can be used reference as a correction value of the answers student exam.

The results using data from the Final Examination in STMIK Indonesia Banjarmasin with 10 subjects, that is: Graphic Design, Computer Networking, Introduction to Information Technology, Skills Inter-Personal, Operating Systems, Introduction to Management, Profession Ethics, Database Systems, Microprosessor and web Programming. Each subjects entered 30 questions with each question have 3 completely different answers as the comparison level of similarity. In the process, the system will remove the

words are considered unimportant or words are commonly used include characters or symbols, because the system only process the questions that need theoretical and arguments answers, not mathematical. For the case in this thesis, words in the local Banjar language also eliminated by the system to equalize use of Indonesian language. With a set remains of words after the removal of the word, the word weighted value calculation algorithms will do TF / IDF and VSM will be calculated the cosine valule, so obtained value of the degree of similarity between answers by students and answers by lecturers. The correlation level result is good enough with the average accuracy rates 80% - 90% if compared with human assessment manually.

Keywords : *Automatic Exam Assessment, TF / IDF, VSM, Similiarity.*

I. Pendahuluan

Seiring perkembangan teknologi informasi dan komputer, dunia pendidikan mengalami perubahan sistem pengajaran yang cukup signifikan. Persaingan antar institusi pendidikan yang semakin ketat dalam meningkatkan kualitas mahasiswanya membuat perkembangan teknologi di bidang pendidikan semakin cepat. Beberapa institusi pendidikan khususnya perguruan tinggi negeri maupun swasta mulai mengembangkan sistem e-learning pada proses pengajarannya.

Dalam konsep e-learning, pelaksanaan ujian dapat dilakukan, mulai dari menjawab soal ujian hingga proses penilaian selama ini kebanyakan proses ujian esai dan penilaiannya dilaksanakan secara manual yaitu dengan membaca esai satu per satu. Para dosen perlu menghabiskan banyak waktu untuk menilai jawaban ujian mahasiswa. Semakin banyak jumlah ujian yang dikoreksi, kualitas penilaian yang diberikan semakin menurun.

Penilaian dalam ujian adalah suatu proses untuk mengambil keputusan dengan menggunakan informasi yang diperoleh melalui pengukuran hasil belajar baik yang menggunakan instrumen ujian maupun yang tidak. Penilaian dengan esai menjadi pilihan pengajar dalam mengevaluasi tingkat kemampuan dari mahasiswanya. Bentuk esai ini oleh banyak peneliti dianggap alat yang sangat ampuh untuk mengukur hasil pembelajaran, begitu juga untuk mengamati kemahiran berpikir tingkat tinggi.

Dalam penerapannya pemberian nilai dari hasil ujian esai dapat dilakukan dengan mudah oleh manusia, tetapi pemilahan jawaban yang dilakukan secara otomatis dengan komputer akan membawa permasalahan tersendiri. Begitu pula dengan mengukur tingkat kemiripan suatu dokumen kata dengan dokumen kata lainnya, manusia dapat dengan mudah mengukur apakah suatu dokumen kata memiliki tingkat kemiripan/similaritas dengan dokumen kata lainnya, tetapi tidak dengan sistem otomatisasi karena melalui beberapa tahapan terlebih dahulu, khususnya dalam penggalian teks untuk mencari kemiripan tersebut.

Text mining adalah salah satu cara dalam mengatasi permasalahan diatas. Text mining merupakan proses pengambilan data berupa teks dari sebuah sumber dalam hal ini sumbernya adalah dokumen. Dengan text mining dapat dicari kata-kata kunci yang dapat mewakili isi dari suatu dokumen lalu dianalisa dan dilakukan pencocokan antara dokumen dengan database kata kunci yang telah dibuat untuk menentukan atau memilah kategori suatu dokumen.

Sedangkan proses pengukuran tingkat similaritas antar dokumen dilakukan dengan membandingkan suatu kata kunci dengan dokumen. Kata kunci yang digunakan didapat dari proses ekstraksi dokumen pada proses pemilahan kategori dokumen. Agar hasil pengukuran tingkat similaritas dokumen dengan kata kunci mendapatkan hasil yang optimal maka digunakan algoritma text mining dimana dalam prosesnya digunakan algoritma TF-IDF (Term Frequency – Inversed Document Frequency) dan VSM (Vector Space Model) dari IR (Information Retrieval) model untuk mencari nilai Cosine (menghitung nilai cosinus sudut antara dua vector) sebagai pengukur tingkat similaritas antara dokumen dengan keyword yang didapat dari ekstraksi teks pada dokumen.

Berdasarkan latar belakang di atas dan kajian pustaka yang sudah dilakukan, maka

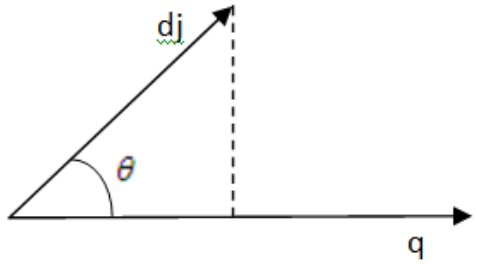
penulis mengusulkan sistem penilaian otomatis jawaban esai menggunakan metode Vector Space Model. Sistem ini digunakan untuk menilai hasil jawaban ujian dalam bentuk esai, dalam kasus penelitian ini dilakukan pada Ujian Akhir Semester (UAS) beberapa mata kuliah di STMIK Indonesia Banjarmasin.

II. Metodologi Penelitian

Vector space model adalah model aljabar untuk mewakili dokumen teks sebagai vektor pengenal, seperti indeks. Hal ini digunakan dalam memilah informasi, pencarian informasi, pengindeksan dan peringkat relevansi. Untuk mengurangi kompleksitas dokumen dan untuk membuat lebih mudah ditangani, dokumen harus diubah dari versi teks ke Dokumen vektor yang menggambarkan isi dari dokumen. Dokumen yang memiliki nilai TF/IDF tinggi memiliki

hubungan kuat dengan *query* dan akan lebih relevan bagi pengguna.

VSM memberikan sebuah kerangka pencocokan parsial adalah mungkin. Hal ini dicapai dengan menetapkan bobot *non-biner* untuk istilah indeks dalam *query* dan dokumen. Bobot istilah yang akhirnya digunakan untuk menghitung tingkat kesamaan antara setiap dokumen yang tersimpan dalam sistem dan permintaan user. Dokumen yang terambil disortir dalam urutan yang memiliki kemiripan, model vektor memperhitungkan pertimbangan dokumen yang relevan dengan permintaan user. Hasilnya adalah himpunan dokumen yang terambil jauh lebih akurat (dalam arti sesuai dengan informasi yang dibutuhkan oleh *user*). Sebuah dokumen dj dan sebuah *query* q direpresentasikan sebagai vektor t-dimensi.



Gambar 1. Representasi Dokumen Dalam Vektor

Dalam VSM koleksi dokumen direpresentasikan sebagai sebuah *matrik term document* (atau *matrik term frequency*). Setiap sel dalam matrik bersesuaian dengan bobot yang diberikan dari suatu term dalam dokumen yang ditentukan. Nilai nol berarti bahwa term tersebut tidak ada dalam dokumen. Gambar 2 menunjukkan *matrik term document* dengan n dokumen dan t term.

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

Gambar 2. *Matrik Term Document*

Keberhasilan dari model VSM ini ditentukan oleh skema pembobotan terhadap suatu term baik untuk cakupan lokal maupun global, dan faktor normalisasi. Pembobotan lokal hanya berpedoman pada frekuensi munculnya term dalam suatu dokumen dan tidak melihat frekuensi kemunculan term tersebut di dalam dokumen lainnya. Pendekatan dalam pembobotan lokal yang paling banyak diterapkan adalah *term frequency* (tf) meskipun terdapat skema lain seperti pembobotan biner, *augmented normalized* tf, *logaritmik* tf dan *logaritmik alternatif*.

Pembobotan global digunakan untuk memberikan tekanan terhadap *term* yang mengakibatkan perbedaan dan berdasarkan pada penyebaran dari *term* tertentu di seluruh dokumen. Banyak skema didasarkan pada pertimbangan bahwa semakin jarang suatu term muncul di dalam total koleksi maka *term* tersebut menjadi semakin berbeda. Pemanfaatan pembobotan ini dapat menghilangkan kebutuhan *stopword removal* karena *stopword* mempunyai bobot global yang sangat kecil. Namun pada prakteknya lebih baik menghilangkan *stopword* di dalam fase *pre-processing* sehingga semakin sedikit *term* yang harus ditangani. Pendekatan terhadap pembobotan global mencakup *inverse document frequency* (idf), *squared* idf, *probabilistic* idf, GF-idf, *entropy*.

Pendekatan idf merupakan pembobotan yang paling banyak digunakan saat ini. Beberapa aplikasi tidak melibatkan bobot global, hanya memperhatikan tf, yaitu ketika tf sangat kecil atau saat diperlukan penekanan terhadap frekuensi *term* di dalam suatu dokumen.. Faktor normalisasi digunakan

untuk menormalkan vektor dokumen sehingga proses retrieval tidak terpengaruh oleh panjang dari dokumen.

Normalisasi ini diperlukan karena dokumen panjang biasanya mengandung perulangan *term* yang sama sehingga menaikkan *term frequency* (tf). Dokumen panjang juga mengandung banyak *term* yang berbeda sehingga menaikkan ukuran kemiripan antara *query* dengan dokumen tersebut, meningkatkan peluang di-retrievenya dokumen yang lebih panjang. Beberapa pendekatan normalisasi adalah normalisasi cosinus, penjumlahan bobot, normalisasi ke- 4, normalisasi bobot maksimal dan normalisasi *pivoted unique*.

Inverse Document Frequency (IDF) didasarkan pada fakta bahwa istilah yang terjadi pada banyak dokumen adalah diskriminator yang tidak baik dan harus diberikan bobot kurang dari satu yang terjadi pada beberapa dokumen.

$$\text{Idf (ti)} = \log \frac{N}{ni}$$

Term Frequency / Inverse Document Frequency Model menggabungkan informasi lokal dan global.

$$W_i = tf_i * \log \left(\frac{D}{dfi}\right)$$

tf_i = frekuensi kata (jumlah kata) atau berapa kali i terjadi pada dokumen.
df_i = frekuensi dokumen atau jumlah dokumen yang mengandung i
D = jumlah dokumen dalam *database*

Ketika ingin mengalikan dua vektor bersama-sama, salah satu cara melakukannya adalah dengan operator yang disebut "*dot product*". simbolnya adalah titik besar "." hasil dari operasi *dot product* adalah skalar, bukan vektor.

$$Q \cdot D_i = \sum_i W_{Q,j} W_{i,j}$$

Menghitung nilai-nilai kesamaan

Cosine D_i = Sim (Q.D_i)

$$\text{Sim (Q.D}_i) = \frac{\sum_i W_{Q,j} W_{i,j}}{\sqrt{\sum_j W_{Q,j}^2} \sqrt{\sum_i W_{i,j}^2}}$$

Information Retrieval

Information Retrieval merupakan bagian dari *computer science* yang berhubungan dengan pengambilan informasi dari dokumen-dokumen yang didasarkan pada isi dan konteks dari dokumen-dokumen itu sendiri. Berdasarkan referensi dijelaskan bahwa *Information Retrieval* merupakan suatu pencarian informasi yang didasarkan pada suatu *query* yang diharapkan dapat memenuhi keinginan user dari kumpulan dokumen yang ada.

Tujuan dari strategi *retrieval* otomatis adalah untuk memperoleh kembali semua dokumen yang relevan dan pada saat yang sama bisa terambil beberapa dokumen yang tidak relevan. Ketika karakterisasi dari sebuah dokumen terpecahkan, harusnya dokumen tersebut direpresentasikan secara relevan dengan sebuah *query*, hal tersebut memungkinkan dokumen diperoleh kembali sebagai respon dari query tersebut.

Dalam cara ini pegindeksan manual mempunyai karakteristik tradisional, ketika memberikan indeks ke dokumen, pengindeks akan berusaha mencarikan dokumen yang diminta oleh *user* sesuai dengan indeksnya. Secara implisit pengindeks membangun *query* untuk dokumen yang relevan. Ketika pengideksan dilakukan secara otomatis, hal tersebut diasumsikan bahwa dengan mendorong teks dari sebuah dokumen (*query*) pada analisis otomatis yang sama, hasilnya akan berupa penyajian dari isi dokumen, dan jika dokumen berkaitan dengan *query*.

Suatu bahasa indeks adalah bahasa yang digunakan untuk menguraikan dokumen dan permintaan. Unsur-Unsur dari bahasa indeks adalah terminologi indeks, yang mungkin diperoleh dari teks dokumen untuk diuraikan, atau mungkin dengan bebas. Bahasa indeks dapat diuraikan menjadi *pre-coordinate* atau *post-coordinate*, yang pertama menunjukkan bahwa terminologi dikoordinir ketika mengindeks dan ketika dalam pencarian. Secara lebih rinci, dalam indeks *pre-coordinate* suatu kombinasi logis tentang segala terminologi indeks mungkin digunakan sebagai suatu label untuk mengidentifikasi suatu kelas dokumen, sedangkan di dalam indeks *post-coordinate* kelas yang sama akan dikenali pada waktu pencarian dengan mengombinasikan kelas dokumen berlabel dengan terminologi indeks individu.

Bahasa indeks yang muncul dari algoritma *conflation* dapat dijelaskan sebagai indeks dengan kosakata yang tak terkendalikan, *post-coordinate* dan merupakan turunan. Kosakata terminologi indeks pada tahap evolusi kumpulan dokumen hanya merupakan satuan

dari semua *conflation* kelas nama.

Ada banyak kontroversi tentang macam bahasa *index* yang mana yang terbaik untuk pencarian kembali dokumen. Perdebatan utama adalah tentang apakah indeks otomatis sebaik atau lebih baik daripada indeks manual. Masing - masing bisa dilakukan pada berbagai tingkatan kompleksitas. Bagaimanapun, sepertinya terbukti dalam keduanya, *indexing* otomatis dan manual, menambahkan kompleksitas dalam wujud kendali yang lebih terperinci. Pesan adalah kosakata tak terkendalikan berdasar pada bahasa alami untuk mencapai efektivitas pencarian kembali yang dapat diperbandingkan dengan kosa kata dengan kendali rumit.

Kategorisasi teks adalah proses untuk menemukan model atau fungsi yang menjelaskan atau membedakan konsep atau kelas data, dengan tujuan untuk dapat memperkirakan kelas dari suatu objek yang labelnya tidak diketahui.

Pada kategorisasi teks, diberikan sekumpulan kategori (label) dan koleksi dokumen yang berfungsi sebagai data latih, yaitu data yang digunakan untuk membangun model, dan kemudian dilakukan proses untuk menemukan kategori yang tepat untuk dokumen test, yaitu dokumen yang digunakan untuk menentukan akurasi dari model. Misalkan ada sebuah dokumen x sebagai inputan, maka output yang dihasilkan oleh model tersebuat adalah kelas atau kategori y dari beberapa kategori tertentu yang telah didefinisikan sebelumnya (y1,,yk).

Adapun contoh dari pemanfaatan kategorisasi teks adalah pengkategorisasian berita ke dalam beberapa kategori seperti bisnis, teknologi, kesehatan dan lain sebagainya. Pengkategorisasian *email* sebagai *spam* atau

bukan. Pengkategorisasian kilasan film sebagai film favorit, netral atau tidak favorit. Pengkategorisasian *paper* yang menarik dan tidak menarik. Dan penggunaan dari kategorisasi teks yang paling umum adalah kategorisasi otomatis dari *web pages* yang dimanfaatkan oleh portal Internet seperti *Yahoo*. Kategorisasi otomatis ini memudahkan proses *browsing* artikel berdasarkan topik tertentu yang dilakukan oleh *user*.

Struktur data yang baik dapat memudahkan proses komputerisasi secara otomatis. Pada *text mining*, informasi yang akan digali berisi informasi-informasi yang strukturnya sembarang. Oleh karena itu, diperlukan proses pengubahan bentuk menjadi data yang terstruktur sesuai kebutuhannya untuk proses dalam *data mining*, yang biasanya akan menjadi nilai-nilai numerik. Proses ini sering disebut *Text Preprocessing*. Setelah data menjadi data terstruktur dan berupa nilai numerik maka data dapat dijadikan sebagai sumber data yang dapat diolah lebih lanjut.

Text Preprocessing adalah mempersiapkan teks menjadi data yang akan mengalami proses pengolahan pada tahapan berikutnya (Mustaqhfiri, 2011). Tujuan dilakukan *pre-processing* adalah memilih setiap kata dari dokumen dan merubahnya menjadi kata dasar yang memiliki arti sempit dan proses *text mining* akan memberikan hasil yang lebih memuaskan (Septiawan, 2010).

Adapun tahapan pada teks *preprocessing* adalah pemecahan kalimat, proses *case folding*, *filtering* kalimat, proses *tokenizing* kata dan proses *stemming* (Mustaqhfiri, 2011). Berikut adalah diagram alir dari *text preprocessing*:

Gambar 3. Diagram Alir Dari *Text Preprocessing*

Pemecahan kalimat teks menjadi kalimat-kalimat. Adapun yang menjadi pemisah kumpulan kalimat adalah tanda tanya "?", tanda titik ".", dan tanda seru "!" (Mustaqhfiri, 2011).

Case Folding adalah pengubahan huruf dalam dokumen teks menjadi huruf kecil ('a' sampai dengan 'z'). Karakter lain selain huruf dianggap sebagai delimiter sehingga karakter tersebut akan dihapus dari dokumen teks. Dalam bahasa pemrograman PHP digunakan fungsi *strtolower* untuk mengubah huruf menjadi huruf kecil semua.

Filtering merupakan proses penghilangan *stopword. Stopword* yaitu katakata yang sering muncul dalam dokumen namun artinya tidak deskriptif dan tidak memiliki keterkaitan dengan tema tertentu. Misal, "di","oleh","karena", dan lain sebagainya. Proses ini dilakukan pada judul dokumen, abstrak dokumen dan masukan *query* secara terpisah. Proses ini lebih mudah dan lebih cepat diproses setelah kata diekstrak dari teks dokumennya.

TABEL I.
CONTOH CASE FOLDING

Kalimat Asal	Setelah Dilakukan Case Folding
Budi berbelanja di warung. Budi membeli roti, selai, dan susu.	budi berbelanja di warung budi membeli roti selai dan susu

Kata yang diperoleh dari tahap *filtering* diperiksa dengan daftar *stopword,* apabila sebuah kata masuk di dalam daftar *stopword* maka kata tersebut tidak akan diproses lebih lanjut (Utomo, 2011).

TABEL III.
CONTOH PROSES TOKENISASI TEKS BAHASA INDONESIA

Teks Bahasa	**Namanya adalah Santiago. Santiago sudah memutuskan untuk mencari sang alkemis.**
Tokens	namanya
	adalah
	santiago
	santiago
	sudah
	memutuskan
	untuk
	mencari
	sang
	alkemis

Biasanya, yang menjadi acuan pemisah antar token adalah spasi dan tanda baca. Tokenisasi seringkali dipakai dalam ilmu linguistik dan hasil tokenisasi berguna untuk analisis teks lebih lanjut.

TABEL II.
CONTOH STOPWORD BAHASA INDONESIA DAN BAHASA BANJAR

yang	pernah	off	akan	pian
mampu	setiap	sering	dengan	ulun
tentang	untuk	pada	belum	kawa
di	dari	hanya	anda	nah
setelah	mendapatkan	atau	sebuah	jua
semua	punya	kita	atas	kada
hampir	telah	sendiri	menurut	nang
juga	memiliki	agak	sesuai	banar
antara	dia	kata	seberang	nangintu
dan	miliknya	begitu	Sebenarnya	ae
ada	bagaimana	beberapa	sekali	lih
seperti	bagaimanapun	mereka	lagi	parak
jadi	jika	kemudian	terhadap	jaka
karena	ke	sana	memungkinkan	haja
sudah	dalam	ini	sendirian	isuk
tetapi	itu	sungguh	bersama	wayah
oleh	sama	keinginan	meskipun	ampih
bisa	paling	adalah	selalu	lawan
tidak	biarkan	kami	apapun	timbul
sayang	mungkin	apa	siapapun	dah
melakukannya	aku	kapan	Saja	amun
lakukan	sebagian	mana	selain	bujur
memang	besar	sementara	muncul	gasan
baik	harus	siapa	Menghargai	hibak
lain	saya	mengapa	tepat	pina

Tokenisasi kata adalah proses untuk membagi teks yang dapat berupa kalimat, paragraf atau dokumen, menjadi token – token / bagian – bagian tertentu.

Stemming merupakan suatu proses yang terdapat dalam sistem IR yang mentransformasikan kata-kata yang terdapat dalam suatu dokumen ke kata-kata akarnya *(root word*) dengan menggunakan aturan-aturan tertentu. Sebagai contoh, kata bersama, kebersamaan, menyamai, akan distem ke *root word*nya yaitu "sama" (Septiawan, 2010).

Teknik *stemming* adalah suatu teknik pencarian bentuk dasar dari suatu *term*. Yang dimaksud dengan *term* itu sendiri adalah tiap kata yang berada pada suatu dokumen teks. *Stemming* dilakukan pada saat pembuatan indeks dari suatu dokumen. Pembuatan indeks dilakukan karena suatu dokumen tidak dapat dikenali langsung oleh suatu sistem temu kembali informasi atau *information retrieval* (IR) *system*. Oleh karena itu, dokumen

tersebut terlebih dahulu perlu dipetakan ke dalam suatu representasi dengan menggunakan teks yang berada di dalamnya. Teknik *stemming* diperlukan selain untuk memperkecil jumlah indeks yang berbeda dari suatu dokumen, juga untuk melakukan pengelompokan kata-kata lain yang memiliki kata dasar dan arti yang serupa namun memiliki bentuk atau *form* yang berbeda karena mendapatkan imbuhan yang berbeda.

Teknik *stemming* terdiri dari berbagai macam metode. Metode pertama yakni *stemming* dengan acuan tabel pemenggalan imbuhan. Proses *stemming* suatu *term* dengan metode ini dilakukan dengan cara menghilangkan imbuhan dari *term* tersebut sesuai dengan tabel acuan pemenggalan imbuhan yang digunakan. Metode kedua merupakan pengembangan dari metode pertama. Metode kedua ini selain menggunakan tabel acuan pemenggalan imbuhan, juga menggunakan suatu kamus kata dasar. Kamus kata dasar ini digunakan sebagai acuan hasil *stemming* saat proses pemenggalan imbuhan selesai dilakukan. Hasil dari proses *stemming* dengan metode ini harus ada pada kamus kata dasar, jika tidak maka *term* yang diinputkan dianggap sebagai bentuk dasar. Metode ketiga dinamakan metode *stemming* berbasis *corpus* (koleksi dokumen) karena hasil *stemming* menggunakan metode ini dipengaruhi oleh koleksi dokumen yang digunakan dalam proses uji coba. Kelas *stem* yang terbentuk dipengaruhi oleh nilai statistik *co-occurence* dari tiap *term* pada kelas *stem* tersebut. Metode ini dikembangkan dari hipotesis awal bahwa dua buah *term* dengan bentuk dasar yang sama akan sering muncul pada koleksi dokumen yang digunakan pada ujicoba. Nilai keseringan muncul secara bersamaan inilah yang dihitung menggunakan statistik *co-occurence*.

Metode ketiga dilatarbelakangi dari masalah *overstemming* dan *understemming*. Inti dari masalah tersebut yakni kemungkinan hasil stemming yang dapat berjumlah lebih dari satu. Kemungkinan hasil

stemming yang lebih dari satu ini diakibatkan oleh algoritma *stemming* yang digunakan. Teknik *hard stemming*, *stemming* dilakukan hingga seluruh imbuhan berhasil dihilangkan, tentunya akan memiliki hasil *stemming* yang berbeda dengan teknik *soft stemming*, proses penghilangan imbuhan langsung dihentikan saat kata dasar dari *term* tersebut ditemukan. Selain itu, ambiguitas pada suatu bahasa juga dapat menyebakan hasil stemming memiliki kemungkinan berjumlah lebih dari satu.

Algoritma Nazief dan Adriani adalah algoritma stemming yang digunakan khusus untuk bahasa indonesia, walaupun ada banyak algoritma stemming lainnya untuk bahasa indonesia, akan tetapi Nazief dan Adriani lebih banyak digunakan oleh para praktisi maupun para pegiat akademik, karena memang sampai saat ini Nazief dan Adriani mempunyai akurat yang baik jika dibandingkan dengan yang lainnya.

Algoritma Nazief & Adriani yang dibuat oleh Bobby Nazief dan Mirna Adriani ini memiliki tahap-tahap sebagai berikut:

1. Pertama cari kata yang akan diistem dalam kamus kata dasar. Jika ditemukan maka diasumsikan kata adalah *root word*. Maka algoritma berhenti.
2. *Inflection Suffixes* (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”) dibuang. Jika berupa *particles* (“-lah”, “-kah”, “-tah” atau “-pun”) maka langkah ini diulangi lagi untuk menghapus *Possesive Pronouns* (“-ku”, “-mu”, atau “-nya”), jika ada.
3. Hapus *Derivation Suffixes* (“-i”, “-an” atau “-kan”). Jika kata ditemukan di kamus, maka algoritma berhenti. Jika tidak maka ke langkah 3a
   a. Jika “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “-k”, maka “-k” juga ikut dihapus. Jika kata tersebut ditemukan dalam kamus maka algoritma berhenti. Jika tidak ditemukan maka lakukan langkah 3b.
   b. Akhiran yang dihapus (“-i”, “-an” atau “-kan”) dikembalikan, lanjut ke langkah 4.
4. Hapus *Derivation Prefix*. Jika pada langkah 3 ada sufiks yang dihapus maka pergi ke langkah 4a, jika tidak pergi ke langkah 4b.
   a. Periksa tabel kombinasi awalan-akhiran yang tidak diijinkan. Jika ditemukan maka algoritma berhenti, jika tidak
   b. Pergi ke langkah 4b.
   c. For i = 1 to 3, tentukan tipe awalan kemudian hapus awalan. Jika *root word* belum juga ditemukan lakukan langkah 5, jika sudah maka algoritma berhenti. Catatan: jika awalan kedua sama dengan awalan pertama algoritma berhenti.
5. Melakukan *Recoding*.

Jika semua langkah telah selesai tetapi tidak juga berhasil maka kata awal diasumsikan sebagai *root word*. Proses selesai.

TABEL IV.
CONTOH HASIL STEMMING

Proses	Kalimat
Isi Dokumen	Pembukaan daftar Wisuda dan pelaksanaan nya lebih baik d umumkan di web ub tidak hanya di Fakultas. sehingga memudahkan Mahasiswa yang ada di luar kota. Pelaksanaan Wisuda sebaiknya terjadwal tidak tergantung pada kuota. sehingga lebih cepat mendapat Ijazah.
Tokenisasi	pembukaan daftar wisuda dan pelaksanaan nya lebih baik d umumkan di web ub tidak hanya di fakultas sehingga memudahkan mahasiswa yang ada di luar kota pelaksanaan wisuda sebaiknya terjadwal tidak tergantung pada kuota sehingga lebih cepat mendapat ijazah
Filtering	pembukaan daftar wisuda pelaksanaan umumkan web ub fakultas memudahkan mahasiswa kota pelaksanaan wisuda sebaiknya terjadwal tergantung kuota cepat ijazah
Stemming	buka daftar wisuda laksana umum web ub fakultas mudah mahasiswa kota laksana wisuda baik jadwal gantung kuota cepat ijazah

Inverted index adalah salah satu mekanisme untuk pengindeksan sebuah koleksi teks yang digunakan untuk mempercepat proses pencarian. Struktur dari *inverted index* terdiri dari dua elemen yaitu kosakata dan posisinya di dalam sebuah dokumen (Baeza-Yates dan Ribeiro-Neto, 1999). Posisi dari sebuah istilah di dalam indeks pada sebuah buku, diterjemahkan dalam bentuk nomor halaman.

Pada *inverted index*, setiap istilah di masukan ke dalam *inverted list* yang menyimpan daftar dari istilah yang menujuk ke sejumlah dokumen yang memiliki istilah tersebut. *Inverted list* juga kadang-kadang di sebut *posting list* (Witten et all, 1999).

Misalkan istilah T_1 terdapat dalam dokumen D_1, D_2, dan D_3 sedangkan istilah T_2 terdapat dalam dokumen D_1 dan D_2 maka *inverted index* yang dihasilkan seperti berikut:

$T_1 \rightarrow D_1, D_2, D_3$
$T_2 \rightarrow D_1, D_2$

Penggunaan *inverted index* di dalam sistem *information retrieval* memiliki kelemahan yaitu lambat di dalam pengindeksan, tetapi cepat di dalam proses pencarian informasi. Menurut Grossman (2002), *Inverted Index* adalah struktur yang dioptimasi untuk proses penemukembalian sedangkan proses *update* hanya menjadi pertimbangan *sekunder*. Struktur tersebut membalik teks sehingga indeks memetakan istilah-istilah ke dokumen - dokumen (sebagaimana indeks sebuah buku yang memetakan istilah-istilah ke nomor halaman).

Pembobotan kemunculan kata dalam suatu dokumen digunakan untuk perhitungan tingkat kemiripan antar dokumen dengan *query* (Abror, 2011). Salah satu metode pembobotan yang sering digunakan adalah TF-IDF.

Terms Frequency – Inverse Document Frequency (TF-IDF)

Metode ini merupakan metode untuk menghitung nilai atau bobot suatu kata (term) pada dokumen. Metode ini akan mengabaikan setiap kata-kata yang tergolong tidak penting. Oleh sebab itu, sebelum melalukan metode ini, proses *stemming* dan *stopword removal* harus dilakukan terlebih dahulu oleh sistem (Pradnyana, 2012).

Faktor lain yang diperhatikan dalam pemberian bobot adalah kalimat yang muncul pada sedikit dokumen harus dipandang sebagai kata yang lebih penting daripada kalimat yang muncul pada banyak dokumen. Pembobotan akan memperhitungkan faktor kebalikan *frekuensi* dokumen yang mengandung suatu kalimat (*inverse document frequency*) (Pradnyana, 2012).

Setelah bobot (W) masing-masing dokumen diketahui, maka dilakukan proses pengurutan (sorting) dimana semakin besar nilai W, maka semakin besar pula tingkat kesamaan (similarity) dokumen tersebut terhadap kata yang dicari, demikian pula sebaliknya.

Fungsi metode ini adalah untuk mencari representasi nilai dari tiap-tiap dokumen dari suatu kumpulan data *training* (*training set*). Dari sini akan dibentuk suatu vektor antara dokumen dengan kata (*documents with terms*) yang kemudian untuk kesamaan antar dokumen dengan *cluster* akan ditentukan oleh sebuah *prototype* vektor yang disebut juga dengan *cluster centroid* (Arifin).

Metode *Cosine Similarity* merupakan metode yang digunakan untuk menghitung similarity (tingkat kesamaan) antar dua buah objek (Pradnyana, 2012).

Metode *cosine similarity* ini menghitung *similarity* antara dua buah objek (misalkan D1 dan D2) yang dinyatakan dalam dua buah vektor dengan menggunakan *keywords* (kata kunci) dari sebuah dokumen sebagai ukuran. Ukuran ini memungkinkan perangkingan dokumen sesuai dengan kemiripannya (relevansi) terhadap *query*. Setelah semua dokumen dirangking, sejumlah tetap dokumen *top-scoring* dikembalikan kepada pengguna. Ukuran pada *cosine similarity* ini menghitung nilai *cosinus* sudut antara dua vektor.

Pada VSM, setiap dokumen dan query dari pengguna direpresentasikan sebagai ruang vektor berdimensi n. Biasanya digunakan nilai bobot istilah (term weigthing) sebagai nilai dari vektor pada dokumen nilai 1 untuk setiap istilah yang muncul pada vektor query. Pada model ini, bobot dari query dan dokumen dinyatakan dalam bentuk vektor, seperti:

$$Q = (W_{q1}, W_{q2}, W_{q3}, \ldots, W_{qt}) \text{ dan}$$
$$D_i = (W_{i1}, W_{i2}, W_{i3}, \ldots, W_{it})$$

Perhitungan Jarak query menggunakan persamaan dan dokumen, menggunakan persamaan

$$|q| = \sqrt{\sum_j^t (W_{i,q})2}$$

Dengan |q| adalah Jarak *query*, dan Wiq adalah bobot *query* dokumen ke-i, maka Jarak query (|q|) dihitung untuk didapatkan jarak *query* dari bobot *query* dokumen (Wiq) yang terambil oleh sistem. Jarak *query* bisa dihitung dengan persamaan akar jumlah kuadrat dari *query*.

$$|d_j| = \sqrt{\sum_i^t (W_{i\,j})\,2}$$

Dengan $|d_j|$ adalah jarak dokumen, dan Wij adalah bobot dokumen ke-i, maka Jarak dokumen ($|d_j|$) dihitung untuk didapatkan jarak dokumen dari bobot dokumen dokumen (W_{ij}) yang terambil oleh sistem. Jarak dokumen bisa dihitung dengan persamaan akar jumlah kuadrat dari dokumen.

Perhitungan pengukuran *Similaritas query document* (*inner product*).

$$\text{Sim } (q, d_j) = \sum_{i=1}^t W_{i\,q} * W_{i\,j}$$

Dengan W_{ij} adalah bobot term dalam dokumen, W_{iq} adalah bobot *query*, dan Sim (q, d_j) adalah Similaritas antara *query* dan dokumen. Similaritas antara *query* dan dokumen atau *inner product* / Sim (q, d_j) digunakan untuk mendapatkan bobot dengan didasarkan pada bobot term dalam dokumen (W_{ij}) dan bobot *query* (W_{iq}) atau dengan cara menjumlah bobot q dikalikan dengan bobot dokumen.

Pengukuran *Cosine Similarity* (menghitung nilai kosinus sudut antara dua vector).

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

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

Ketika sebuah dokumen dimasukkan dalam suatu ruang vektor, maka dibayangkan sebuah ruang vektor yang memiliki dimensi luar biasa besar dan ditentukan oleh banyaknya term/kata yang terbentuk saat proses pengindeksan dokumen. Pengindeksan terhadap kumpulan dokumen yang dapat dicari merupakan sebuah proses tersendiri. Konsep umum yang diterapkan untuk pembuatan struktur indeks adalah menggunakan model *inverted index*. Dalam *inverted index*, ada sebuah daftar kamus (*dictionary*) yang berisi kata atau term hasil dari pemrosesan setiap dokumen. Dari setiap kata yang ada dalam *dictionary*, lalu terbentuk sebuah *linked list* yang berisi urutan dokumen yang mengandung term tersebut.

Contoh kasus untuk pencarian kemiripan nilai suatu kata atau dokumen dengan *Term Document / Inversed Document Frequency* (TF/IDF) dan *Vector Space Model* (VSM).

Kata kunci (kk) = pengetahuan logistik
Dokumen1 (D_1) = manajemen transaksi logistik
Dokumen2 (D_2) = pengetahuan antar individu
Dokumen3(D_3) = dalam manajemen pengetahuan terdapat transfer

pengetahuan logistik Jadi jumlah dokumen (D) = 3

Setelah dilakukan tahap tokenizing dan proses *filtering*, maka kata antar pada dokumen 2 serta kata dalam dan terdapat pada dokumen 3 dihapus. Berikut pada Tabel 3.9 adalah contoh perhitungan TF/IDF.

Dari contoh studi kasus tersebut, dapat diketahui bahwa nilai bobot (W) dari D_1 dan D_2 adalah sama. Apabila hasil pengurutan bobot dokumen tidak dapat mengurutkan secara tepat, karena nilai W keduanya sama, maka diperlukan proses perhitungan dengan algoritma *vector space model*. Ide dari metode ini adalah dengan menghitung nilai *cosinus* sudut dari dua vektor, yaitu W dari tiap dokumen dan W dari kata kunci.

TABEL V.
CONTOH PERHITUNGAN TF/IDF

bobot (W) untuk $D_1 = 0.176 + 0 = 0.176$
bobot (W) untuk $D_2 = 0 + 0.176 = 0.176$
bobot (W) untuk $D_3 = 0.176 + 0.352 = 0.528$

Apabila studi kasus pada algoritma TF/IDF di atas dicari nilai *cosinus* sudut antara vektor masing-masing dokumen dengan kata kunci, maka hasil yang didapatkan akan lebih presisi. Seperti yang ditunjukan Tabel VI.

TABEL VI.
CONTOH PENCARIAN NILAI *COSINUS* PADA MASING – MASING DOKUMEN

Token	kk	D1	D2	D3	kk*D1	kk*D2	kk*D3
manajemen	0	0,031	0	0,031	0	0	0,176
transaksi	0	0,228	0	0	0	0	0
logistik	0,031	0,031	0	0,031	0,031	0	0,031
transfer	0	0	0	0,228	0	0	0
pengetahuan	0,031	0	0,031	0,124	0	0,031	0,062
individu	0	0	0,228	0	0	0	0
	Sqrt(kk)	Sqrt(Di)			Sqrt(kk.Di)		
	0,249	0,539	0,509	0,643	0,031	0,031	0,093

Selanjutnya menghitung nilai *cosinus* sudut antara *vector* kata kunci dengan tiap dokumen dengan mengunakan rumus:

$$Co\ sin\ e(D_i) = sum\ (kk * D_i) / (sqrt(kk) * sqrt\ (D_i)$$

Untuk Dokumen 1 (D_1)

Cosine (D_1) = sum (kk . D_1) / (sqrt(kk) * sqrt(D_1))
= 0.031 / (0.249 * 0.539)
= 0.231

Untuk Dokumen 2 (D_2)

Cosine (D_2) = sum (kk . D_2) / (sqrt(kk) * sqrt(D_2))
= 0.031 / (0.249 * 0.509)
= 0.245

Untuk Dokumen 2 (D_3)

Cosine (D_3) = sum (kk . D_3) / (sqrt(kk) * sqrt(D_3))
= 0.093 / (0.249 * 0.643)
= 0.581

Sesuai perhitungan diatas maka nilai cosinus setiap dokumen telah didapat, seperti Tabel 7.

TABEL VII.
CONTOH HASIL PERHITUNGAN NILAI *COSINUS*

	D1	D2	D3
Cosine	0,231	0,245	0,581
	Rank 3	**Rank 2**	**Rank 1**

Dari hasil akhir tersebut dapat diketahui bahwa dokumen 3 (D_3) memiliki tingkat similaritas tertinggi terhadap kata kunci, kemudian disusul dengan D_2 dan D_1.

III. ANALISA SISTEM

Retrival informasi merupakan bagian dari ilmu komputer yang berhubungan dengan pengambilan informasi dari dokumen-dokumen yang didasarkan pada isi dan konteks dari dokumen-dokumen itu sendiri. Retrival informasi merupakan suatu pencarian

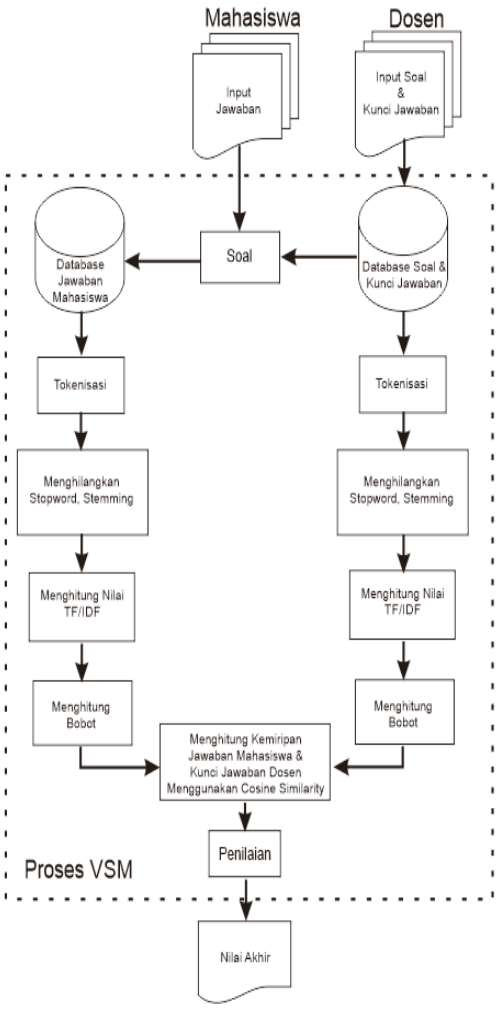
informasi (biasanya berupa dokumen) yang didasarkan pada suatu query (input pengguna) yang diharapkan dapat memenuhi keinginan pengguna dari kumpulan dokumen yang ada.

Sedangkan, definisi query dalam retrival informasi merupakan sebuah formula yang digunakan untuk mencari informasi yang dibutuhkan oleh pengguna, dalam bentuk yang paling sederhana, sebuah query merupakan suatu keywords (kumpulan kata kunci) dan dokumen yang mengandung keywords merupakan dokumen yang dicari dalam sistem retrival informasi. Proses dalam retrival informasi dapat digambarkan sebagai sebuah proses untuk mendapatkan dokumen yang relevan dari kumpulan dokumen yang ada melalui pencarian query yang di-input oleh pengguna.

Retrival dokumen teks adalah sebuah cabang dari retrival informasi dimana informasi yang disimpan adalah berupa teks. Sistem retrival dokumen teks menemukan informasi dari kriteria yang diberikan dengan mencocokkan query yang dimasukkan oleh pengguna akhir dengan dokumen-dokumen teks yang telah tersimpan. Sebuah sistem retrival dokumen teks terdiri dari koleksi dokumen teks, sebuah algoritma klasifikasi untuk membangun indeks, dan sebuah antarmuka pengguna untuk menghubungkan dengan koleksi.

Sebuah sistem retrival dokumen teks memiliki dua tugas utama, yaitu :

1. Menemukan dokumen teks yang relevan sesuai dengan query yang dimasukkan.
2. Mengevaluasi dokumen teks yang cocok dan mengurutkan sesuai dengan relevansinya dengan menggunakan algoritma tertentu.



Gambar 3. Blok Diagram

Seperti yang digambarkan dalam blok diagram pada Gambar 3, sistem ini dimulai dari penginputan data soal beserta kunci jawaban oleh dosen atau admin, yang mana setiap 1 soal memiliki 3 jawaban benar yang berbeda. Database kunci jawaban jawaban yang telah tersimpan akan dilakukan proses kategorisasi teks, begitu pula dengan jawaban yang diinput oleh mahasiswa. Sedangkan soal ujian yang telah tersimpan, akan ditampilkan saat mahasiswa melakukan ujian sesuai mata kuliah yang dipilih. Jawaban dari mahasiswa tersebutlah yang nantinya akan diproses sehingga bisa didapat nilai akhirnnya.

Adapun tahap-tahap kategorisasi teks seperti yang digambarkan dalam blok diagram diatas, akan dijelaskan sebagai berikut :

1. Tokenisasi

Tokenisasi adalah sebuah proses untuk memilah isi dokumen teks sehingga menjadi satuan kata - kata. Pemilahan ini biasanya dilakukan dengan cara memisahkan kalimat menjadi kata - kata dan menghilangkan kata-

kata yang bukan merupakan alfabet dan angka. Semua huruf kapital diubah menjadi huruf kecil agar token dapat diurutkan secara alfabet dan diperlakukan sama dengan token-token lain. Dalam penelitian tesis ini proses token dibuat menjadi 4 Dalam penelitian tesis ini proses token dibuat menjadi 4, yaitu : token untuk Jawaban mahasiswa (Query), token untuk kunci jawaban 1 (Dokumen 1), token untuk kunci jawaban 2 (Dokumen 2), dan token untuk kunci jawaban 3 (Dokumen 3).

Contoh proses tokenisasi dalam salah satu kalimat jawaban dalam penelitian ini :

Kalimat asal : basis data adalah suatu kumpulan data terhubung yang disimpan secara bersama-sama pada suatu media, yang diorganisasikan berdasarkan sebuah skema atau struktur tertentu, dan dengan software untuk melakukan manipulasi untuk kegunaan tertentu.

Kalimat setelah mengalami tokenisasi : |basis| |data| |adalah| |suatu| |kumpulan| |data| |terhubung| |yang| |disimpan| |secara| |bersama| |-| |sama| |pada| |suatu| |media| |,| |yang| |diorganisasikan| |berdasarkan| |sebuah| |skema| |atau| |struktur| |tertentu| |,| |dan| |dengan| |software| |untuk| |melakukan| |manipulasi| |untuk| |kegunaan| |tertentu| |.|

2. Filtering (Menghilangkan Stopword)

Tahap filtering adalah tahap mengambil kata-kata penting dari hasil token. Biasanya dilakukan dengan cara menghilangkan stopwords dan stoplist dari token yang telah diperoleh dari proses tokenisasi. Dalam penelitian tesis ini, beberapa kata-kata umum dalam bahasa lokal banjar juga dimasukkan kedalam kamus stopword. Data kamus stopword bisa dilihat pada lembar lampiran.

Contoh proses filtering diambil dari kalimat yang sudah ditokenisasi pada contoh kasus sebelumnya dan kamus stopword mengacu pada lembar lampiran A1 :

Kalimat sebelum proses filtering : |basis| |data| |adalah| |suatu| |kumpulan| |data| |terhubung| |yang| |disimpan| |secara| |bersama| |-| |sama| |pada| |suatu| |media| |,| |yang| |diorganisasikan| |berdasarkan| |sebuah| |skema| |atau| |struktur| |tertentu| |,| |dan| |dengan| |software| |untuk| |melakukan| |manipulasi| |untuk| |kegunaan| |tertentu| |.|

Kalimat setelah filtering : |basis| |data| |kumpulan| |data| |terhubung| |disimpan| |media| |diorganisasikan| |berdasarkan| |skema| |struktur| |software| |manipulasi| |kegunaan|

3. Stemming

Stemming merupakan suatu proses untuk menemukan kata dasar dari sebuah kata. Dengan menghilangkan semua imbuhan (affixes) baik yang terdiri dari awalan (prefixes), sisipan (infixes), akhiran (suffixes) dan confixes (kombinasi dari awalan dan akhiran) pada kata turunan. Dalam penelitian tesis ini, proses stemming dilakukan dengan memakai algoritma stemming Nazief dan Adriani.

Contoh hasil proses stemming yang diambil dari kalimat yang sudah ditokenisasi dan difiltering pada contoh kasus sebelumnya :

Kalimat sebelum stemming : |basis| |data| |kumpulan| |data| |terhubung| |disimpan| |media| |diorganisasikan| |berdasarkan| |skema| |struktur| |software| |manipulasi| |kegunaan|

Kalimat setelah proses stemming : |basis| |data| |kumpul| |data| |hubung| |simpan| |media| |organisasi| |dasar| |skema| |struktur| |software| |manipulasi| |guna|

4. TF/IDF (Term Frequency – Inversed Document Frequency)

Term Frequency (tf) factor, yaitu faktor yang menentukan bobot term pada suatu dokumen berdasarkan jumlah kemunculannya dalam dokumen tersebut. Nilai jumlah kemunculan suatu kata (term frequency) diperhitungkan dalam pemberian bobot terhadap suatu kata. Semakin besar jumlah kemunculan suatu term (tf tinggi) dalam dokumen, semakin besar pula bobotnya dalam dokumen atau akan memberikan nilai kesesuian yang semakin besar.

Inverse Document Frequency (idf) factor, yaitu pengurangan dominansi term yang sering muncul di berbagai dokumen. Hal ini diperlukan karena term yang banyak muncul di berbagai dokumen, dapat dianggap sebagai term umum (common term) sehingga tidak penting nilainya. Sebaliknya faktor kejarangmunculan kata (term scarcity) dalam koleksi dokumen harus diperhatikan dalam pemberian bobot. Menurut Mandala (dalam Witten, 1999) 'Kata yang muncul pada sedikit dokumen harus dipandang sebagai kata yang lebih penting (uncommon tems) daripada kata yang muncul pada banyak dokumen. Pembobotan akan memperhitungkan faktor kebalikan frekuensi dokumen yang mengandung suatu kata (inverse document frequency). Hal ini merupakan usulan dari George Zipf. Zipf mengamati bahwa frekuensi dari sesuatu cenderung kebalikan secara proposional dengan urutannya.'

Metode TF-IDF merupakan metode pembobotan term yang banyak digunakan sebagai metode pembanding terhadap metode

pembobotan baru. Pada metode ini, perhitungan bobot term t dalam sebuah dokumen dilakukan dengan mengalikan nilai Term Frequency dengan Inverse Document Frequency.

$$tf_{(i)} = \frac{freq_i(d_j)}{\sum_{i=1}^{k} freq_i(d_j)}$$

$$idf_i = \log \frac{|D|}{|\{d : t_i \in d\}|}$$

5. Menghitung Pembobotan

Setelah didapat nilai hasil perhitungan TF/IDF, nilai tersebut digunakan untuk menghitung bobot (W) masing - masing dokumen terhadap kata kunci dengan rumus yaitu :

$$W_{dt} = tf_{df} * IDF_t$$

Setelah bobot (W) masing-masing dokumen diketahui, maka dilakukan proses sorting / pengurutan dimana semakin besar nilai W, semakin besar tingkat similaritas dokumen tersebut terhadap kata kunci, demikian sebaliknya.

6. Menghitung Cosine Similarity Dengan Metode Vector Space Model

Vector space model (VSM) adalah suatu model yang digunakan untuk mengukur kemiripan antara suatu dokumen dengan suatu query. Pada model ini, query dan dokumen dianggap sebagai vektor-vektor pada ruang n-dimensi, dimana n adalah jumlah dari seluruh term yang ada dalam leksikon. Leksikon adalah daftar semua term yang ada dalam indeks. Salah satu cara untuk mengatasi hal tersebut dalam vector space model adalah dengan cara melakukan perluasan vektor. Proses perluasan dapat dilakukan pada vektor query, vektor dokumen, atau pada kedua vektor tersebut. Pada algoritma vector space model gunakan rumus untuk mencari nilai cosinus sudut antara dua vector dari setiap bobot dokumen (WD) dan bobot dari kata kunci (WK). Rumus yang digunakan adalah sebagai berikut :

$$\text{Sim}(Q, D_i) = \frac{\sum_i W_{Q,j} \; W_{i,j}}{\sqrt{\sum_j W_{Q,j}^2}\sqrt{\sum_i W_{i,j}^2}}$$

7. Penilaian

Setelah dilakukan perhitungan cosine similarity per satu soal yang telah dijawab mahasiswa, nilai tertinggi antara D1, D2, dan D3 akan diambil sebagai poin nilai jawaban. Jika jawaban mahasiswa sama dengan yang ada pada kunci jawaban dosen, maka akan mendapatkan nilai 1. Sedangkan jika jawaban mahasiswa tersebut tidak ada satupun kata yang sama terdapat pada kunci jawaban dosen, maka akan mendapatkan nilai 0.

8. Nilai Akhir

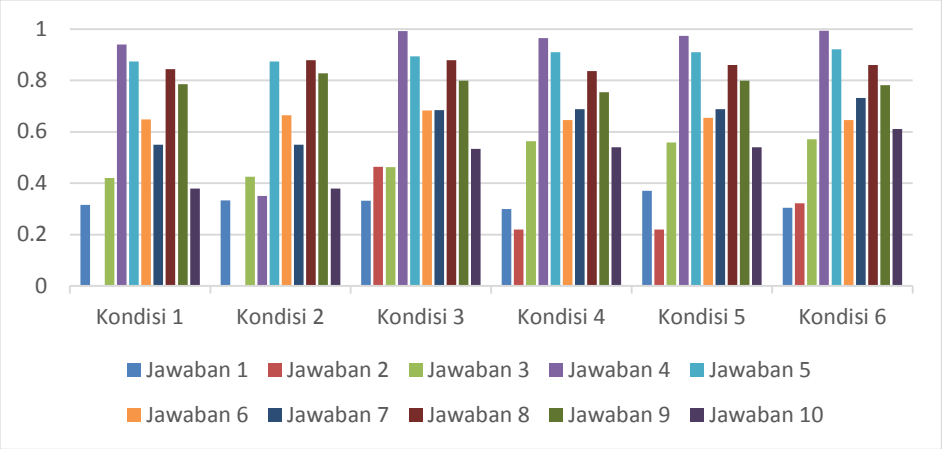
Nilai akhir adalah nilai yang akan didapat oleh mahasiswa setelah menyelesaikan menjawab semua soal yang ditampilkan sesuai mata kuliah yang dipilih. Semua poin nilai per satu soal yang telah dijawab akan dijumlahkan semua (total ada 10 soal).

Uji Coba Sistem

Dalam pengaplikasian sistem penilaian otomatis dengan VSM ini, perhitungan nilai kemiripan kata dilakukan dalam beberapa tahap, yaitu : menghapus karakter atau simbol, menghilangkan imbuhan kata, menghapus kata yang tidak diperlukan atau dianggap tidak penting, menggabungkan kata - kata yang dianggap memiliki satu makna, melakukan pembobotan pada masing – masing kata, menghitung nilai TF/IDF, menghitung nilai cosinus untuk mencari nilai cosinus tertinggi dari tiga dokumen jawaban dosen, menampilkan hasil nilai perhitungan cosinus yang tertinggi sebagai nilai hasil jawaban mahasiswa.

Pada Gambar 4, telah dilakukan perbandingan dari hasil nilai jawaban mahasiswa 1 (mengacu pada lembar lampiran A1, Hal. A-12). Proses dilakukan pada soal dan jawaban yang sama tetapi dengan pengkondisian yang berbeda. Dalam contoh kasus penelitian ini dilakukan dengan 6 pengkondisian proses, yaitu :

- Kondisi 1 : Stopword aktif, penggabungan kata aktif, sinonim kata aktif
- Kondisi 2 : Stopword aktif, penggabungan kata off, sinonim kata aktif
- Kondisi 3 : Stopword aktif, penggabungan kata off, sinonim kata off
- Kondisi 4 : Stopword off, penggabungan kata aktif, sinonim kata aktif
- Kondisi 5 : Stopword off, penggabungan kata off, sinonim kata aktif
- Kondisi 6 : Stopword off, penggabungan kata off, sinonim kata off

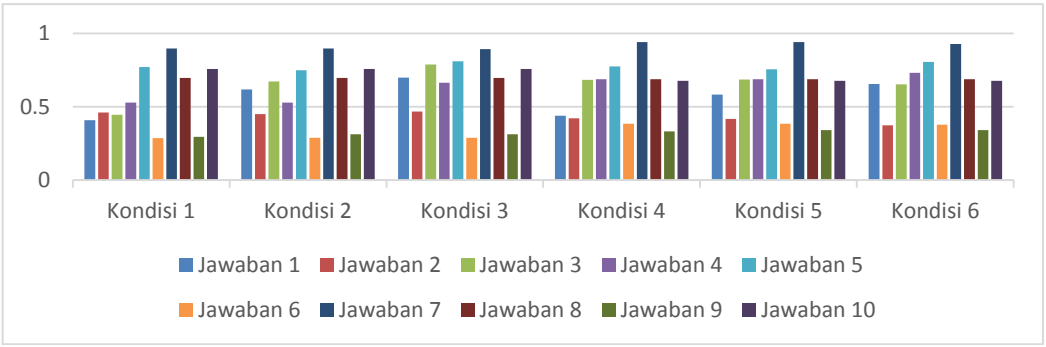


Gambar 4.
Grafik Perbandingan Hasil Jawaban Mahasiswa Pertama

Dari grafik diatas dapat dilihat rata-rata nilai cosine similarity menjadi sedikit lebih tinggi saat kondisi 6 (Stopword tidak aktif, penggabungan kata tidak aktif, sinonim kata tidak aktif). Selain itu, nilai jawaban ke 2 dari kondisi 1-2 bernilai 0. Tetapi saat diuji pada kondisi 3-6, nilai jawaban ke 2 memunculkan angka.

Pada Gambar 5, telah dilakukan perbandingan dari hasil nilai jawaban mahasiswa 2 (mengacu pada lembar lampiran A1, Hal. A-19). Proses dilakukan pada soal dan jawaban yang sama tetapi dengan pengkondisian yang berbeda. Dalam contoh kasus penelitian ini dilakukan dengan 6 pengkondisian proses, yaitu :

- Kondisi 1 : Stopword aktif, penggabungan kata aktif, sinonim kata aktif
- Kondisi 2 : Stopword aktif, penggabungan kata off, sinonim kata aktif
- Kondisi 3 : Stopword aktif, penggabungan kata off, sinonim kata off
- Kondisi 4 : Stopword off, penggabungan kata aktif, sinonim kata aktif
- Kondisi 5 : Stopword off, penggabungan kata off, sinonim kata aktif
- Kondisi 6 : Stopword off, penggabungan kata off, sinonim kata off



Gambar 5.
Grafik Perbandingan Hasil Jawaban Mahasiswa Kedua

Dari grafik diatas dapat dilihat rata-rata nilai cosine similarity menjadi lebih beragam, masing-masing kondisi menghasilkan nilai tertinggi yang berbeda. Tetapi nilai cosine yang lebih dominan tertinggi dihasilkan pada kondisi 3 (Stopword aktif, penggabungan kata tidak aktif, sinonim kata tidak aktif).

IV. KESIMPULAN

Berdasarkan perancangan dan hasil analisis yang dilakukan dalam penelitian, dapat disimpulkan hal-hal berikut ini.

1. Semakin banyak kata atau panjang kalimat dari jawaban dosen yang tersimpan di database akan mempengaruhi lamanya

pemprosesan kata, sehingga saat mahasiswa menjawab soal dan melanjutkan ke soal berikutnya akan terjeda beberapa saat.

2. Saat admin atau dosen membuka data soal untuk melihat hasil proses penilaian dari sistem akan ada jeda beberapa saat yang lamanya tergantung banyak tidaknya kata dalam data tersebut, hal ini dikarenakan sistem akan melakukan proses pengulangan kembali pada data yang dipilih.
3. Penggunaan kata singkatan atau terjadi kesalahan pengetikan kata saat mahasiswa menjawab soal akan mengurangi nilai similaritas, karena tidak terdapat dalam dataset jawaban dosen, sehingga kata tersebut akan dianggap bernilai 0 (kosong) .
4. Rata – rata keberhasilan sistem dalam memproses kemiripan antara jawaban mahasiswa dengan
5. dataset jawaban dosen sangat baik sebesar 80% - 90% karena sesuai dengan pengkategorian secara manual.
6. Kalimat yang terdapat unsur kata bantahan atau negasi dalam beberapa kondisi masih tidak bisa diproses dengan baik.
7. Saat dilakukan ujicoba dengan tidak mengaktifkan penghapusan stopword, penggabungan kata, dan penyamaan kata, waktu proses menjadi lebih singkat dan nilai rata-rata perhitungan menjadi sedikit lebih tinggi.

REFERENSI

[1] Ahmed Alzahrani, Abdulkareem Alzahrani, Fawaz K. Al Arfaj, Khalid Almohammadi, Malek Alrashidi. 2015. School of computer science, University of Essex, Colchester, UK. "AutoScor: An Automated System for Essay Questions Scoring". International Journal of Humanities Social Sciences and Education (IJHSSE), Volume 2, Issue 5, May 2015, PP 182-187 ISSN 2349-0373 (Print) & ISSN 2349-0381 (Online).

[2] Charu C, Aggarwal. Chengxiang Zhai. IBM T. J. Watson Research Center, Yorktown Heights, NY, USA. Niversity of Illinois at Urbana-Champaign, Urbana, IL, USA. “Mining Text Data”. Kluwer Academic Publishers.

[3] D. Manning Christopher, Raghavan Prabhakar, Schütze Hinrich. 2009. Cambridge University Press Cambridge, England. "An Introduction to Information Retrieval". Online edition (c) Cambridge UP.

[4] Feldman Ronen, Sanger James. 2007. Cambridge University Press Cambridge, England. "The Text Mining Handbook Advanced Approaches in Analyzing Unstructured Data". Published in the United States of America by Cambridge University Press, New York.

[5] Hongbo Chen, Ben He, Tiejian Luo, Baobin Li. 2012. School of Computer and Control Engineering Graduate University of the Chinese Academy of Sciences Beijing, China. "A Ranked-based Learning Approach To Automated Essay Scoring". Second International Conference on Cloud and Green Computing.

[6] Ismail, Israml. Yukawa, Takashi. 2004. Nagaoka University of Technology, 1603-1, Kamitomioka-cho, Nagaoka-shi, Niigata 940-2188, Japan. "Question Answering System Using Concept-based Vector Space Model". Working Notes of NTCIR-4, Tokyo, 2-4 June.

[7] Kakkonen Tuomo, Myller Niko, Sutinen Erkki, Timonen Jari. 2008. Department of Computer Science and Statistics, University of Joensuu, Finland. "Comparison of Dimension Reduction Methods for Automated Essay Grading".Educational Technology & Society, 11(3), 275–288.

[8] Karmayasa, Oka. Mahendra, Ida Bagus. Program Studi Teknik Informatika, Jurusan Ilmu Komputer, Fakultas Matematika Dan Ilmu Pengetahuan Alam, Universitas Udayana. "Implementasi Vector Space Model Dan Beberapa Notasi Metode Term Frequency Inverse Document Frequency (TF-IDF) Pada Sistem Temu Kembali Informasi".

[9] Lu, Kun. Wolfram, Dietmar. School of Information Studies U. of Wisconsin-Milwaukee. "Assessing Author Research Focus Using Vector Space Modeling".

[10] Matta, Deepika. Verma, Manoj. 2013. Department of Computer Science, RPIIT, Karnal, India. "Evaluating Relevancy Of Words In Document Queries Using Vector Space Model". Journal of Engineering, Computers & Applied Sciences (JEC&AS). Volume 2, No.6.

[11] Menteri Pendidikan Dan Kebudayaan. “Ketentuan Pokok Penyelenggaraan Perguruan Tinggi Swasta”. Salinan Keputusan Menteri Pendidikan Dan Kebudayaan Republik Indonesia Nomor 0339/U/1994.

[12] Monjurul Islam, Md. Latiful Hoque, A. S. M. 2012. Department of Computer Science and Engineering, Bangladesh University of Engineering & Technology (BUET) Dhaka, Bangladesh. "Automated Essay Scoring Using Generalized Latent Semantic Analysis". Journal Of Computers, Vol. 7, No. 3, March.

[13] Nagata Ryo, Kakegawa Junichi, Yabuta Yukiko. 2009. Konan University, Hyogo University of Teacher Education, and Seisen Jogakuin College. "A Topic-independent Method for Automatically Scoring Essay Content Rivaling Topic-dependent Methods".

Ninth IEEE International Conference on Advanced Learning Technologies.

[14] Rahimi Fitri, Arifin Noor Asyikin. 2015. Jurusan Teknik Elektro Politeknik Negeri Banjarmasin. "Aplikasi Penilaian Ujian Essay Otomatis Menggunakan Metode Cosine Similarity". Jurnal Poros Teknik, Volume 7 No. 2, Desember 2015 : 54-105, ISSN 2085-5761 (Print) & ISSN 2442-7764 (Online).

[15] Sahriar Hamza, M. Sarosa, Purnomo Budi Santoso. 2013. Jurusan Teknik Elektro Universitas Brawijaya, Malang, Indonesia. "Sistem Koreksi Soal Essay Otomatis Dengan Menggunakan Metode Rabin Karp". Jurnal EECCIS Vol. 7, No. 2, Desember 2013.

[16] Wiyogo, Mardiyanto. Firdaus, Yanuar. Widhiana, Rimba. Fakultas Informatika, Institut Teknologi Telkom, Bandung. "Alat Bantu Penilaian Jawaban Esai Bertipe Restricted Response Menggunakan Vector Space Model (VSM) dan Keyphrase Extraction Algorithm (KEA)".

[17] Yali Li, YonghongYan. 2012. Key Laboratory of Speech Acoustics and Content Understanding, Chinese Academy of Sciences Beijing, China. "An effective automated essay scoring system using support vector regression". Fifth International Conference on Intelligent Computation Technology and Automation.

[18] Yousef Moh'd Arikat. 2012. Technology and applied science department Al-Quds Open University. "Subtractive Neuro-Fuzzy Modeling Techniques Applied to Short Essay Auto- Grading Problem". 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT).