Peringkasan Teks Otomatis Berita Berbahasa Indonesia Menggunakan Metode *Maximum Marginal Relevance*

Muchammad Mustaqhfiri[1], Zainal Abidin[2] dan Ririen Kusumawati[3]
Jurusan Teknik Informatika, Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Email: [1]admust86@gmail.com, [2]zainal@ti.uin-malang.ac.id dan
[3]rizn.kusumawati@gmail.com

Abstrak-*Perkembangan teknologi internet berdampak bertambahnya jumlah situs berita berbahasa Indonesia dan menciptakan ledakan informasi. Hal tersebut menuntut semua informasi bisa diakses dengan cepat dan tidak harus membutuhkan banyak waktu dalam membaca sebuah headline berita.Teknologi peringkas teks otomatis menawarkan solusi untuk membantu pencarian isi berita berupa deskripsi singkat (summary). Penelitian diawali dengan lima tahap text preprocessing: pemecahan kalimat,case folding, tokenizing, filtering, dan stemming. Proses selanjutnya menghitung bobot tf-idf, bobot query relevance dan bobot similarity. Ringkasan dihasilkan dari ekstraksi kalimat dengan menggunakan metode maximum marginal relevance. Metode ekstraksi maximum marginal relevance merupakan metode yang digunakan untuk mengurangi redudansi dalam perangkingan kalimat pada multi dokumen. Data uji coba diambil dari surat kabar berbahasa Indonesia online sejumlah 30 berita. Hasil pengujian dibandingkan dengan ringkasan manual yang menghasilkan rata-rata recall 60%, precision 77%, dan f-measure 66%.*

Kata kunci: *peringkasan, text preprocessing, tf-idf, query relevance, similarity, maximum marginal relevance*

1. PENDAHULUAN

Membaca merupakan bagian dari kebutuhan manusia, baik membaca buku, surat kabar, dan majalah. Perkembangan teknologi komunikasi berdampak pada penggunaan internet untuk mempublikasi artikel di situs-situs di internet. Demikian juga dengan artikel-artikel berita, berita banyak yang diunggah di situs-situs surat kabar *on-line*.

Ringkasan dibutuhkan untuk mendapatkan isi artikel secara ringkas. Ringkasan merupakan ekspresi yang ketat dari isi utama suatu artikel, tujuannya untuk memberitahu pada pembaca inti dari suatu pikiran utama (Sartuni, Finoza dan Sundari, 1984:97). Konsep sederhana ringkasan adalah mengambil bagian penting dari keseluruhan isi dari artikel. Menurut Mani dan Maybury, ringkasan adalah mengambil isi yang paling penting dari sumber informasi yang kemudian menyajikannya kembali dalam bentuk yang lebih ringkas bagi penggunanya (Mani dan Maybury, 1999). Aplikasi peringkasan teks otomatis merupakan teknologi yang menawarkan solusi untuk mencari informasi dengan menghasilkan ringkasan (*summary)* berita.

2. TINJAUAN PUSTAKA

2.1 Peringkasan Teks Otomatis

Peringkasan teks otomatis (*automatic text summarization*) adalah pembuatan bentuk yang lebih singkat dari suatu teks dengan memanfaatkan aplikasi yang dijalankan dan dioperasikan pada komputer. Sedangkan menurut Hovy, ringkasan adalah teks yang dihasilkan dari sebuah teks atau banyak teks, yang mengandung isi informasi dari teks asli dan panjangnya tidak lebih dari setengah panjang teks aslinya (Hovy, 2001).

Penelitian peringkasan teks otomatis dipelopori oleh Luhn sejak tahun 1958. Teknik-teknik yang digunakan dalam peringkasan: (1) teknik pendekatan statistika: teknik *word frequency* (Luhn, 1958), *position in text* (Baxendale, 1958), *cue words and heading* (Edmudson, 1969), *sentence position* (Lin dan Hoovy, 1997), (2) teknik pendekatan dengan *natural language analysis*: *inverse term frequency and NLP technique* (Aone, 1990), *lexical chain* (Mc Keown, 1997), *maximal marginal relevance* (Cabonell dan Goldstein, 1998).

2.2 *Maximum Marginal Relevance*

Algoritma *maximum marginal relevance* (*MMR*) merupakan salah satu metode ekstraksi ringkasan (*extractive summary*) yang digunakan untuk meringkas dokumen tunggal atau multi dokumen. *MMR* meringkas dokumen dengan menghitung kesamaan (*simlarity*) antara bagian teks.

Pada peringkasan dokumen dengan metode *MMR* dilakukan proses segmentasi dokumen menjadi kalimat dan dilakukan pengelompokan sesuai dengan *gender* kalimat tersebut. *MMR* digunakan dengan mengkombinasikan matrik *cosine similarity* untuk merangking kalimat-kalimat sebagai tanggapan pada *query* yang diberikan oleh *user*.

Kebanyakan mesin pencarian *information retrieval* (*IR*) modern menghasilkan daftar perangkingan dari dokumen yang diukur dari penurunan relevansi terhadap permintaan user (*user query*). Penaksiran pertama untuk mengukur hasil peringkasan yang relevan adalah dengan mengukur hubungan antar informasi yang ada dalam dokumen dengan *query* yang diberikan oleh user dan menambah kombinasi linier sebagai sebuah matrik. Kombinasi linier ini disebut "*marginal relevance*" (Carbonell dan Goldstein, 1998).

Sebuah dokumen dikatakan mempunyai *marginal relevance* yang tinggi jika dokumen tersebut relevan terhadap isi dari dokumen dan mempunyai kesamaan bobot *term* maksimum dibandingkan dengan *query*. Peringkasan dokumen dengan tipe ekstraktif, nilai akhir diberikan pada kalimat S_i dalam *MMR* dihitung dengan persamaan 1.

$$MMR = \arg\max\left[\lambda * Sim_1(S_i, Q) - (1-\lambda) * \max Sim_2(S_i, S')\right] \quad (1)$$

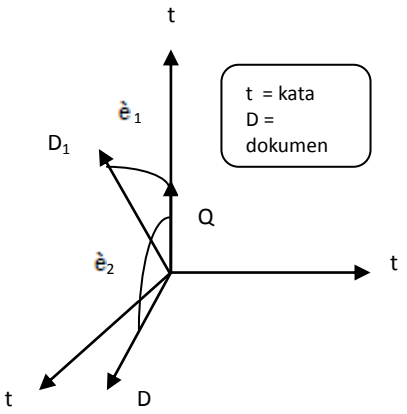
S_i adalah kalimat di dokumen, sedangkan *S'* adalah kalimat yang telah dipilih atau telah diekstrak (Shasha Xie, 2010). Koefisien λ digunakan untuk mengatur kombinasi nilai untuk memberi penekanan bahwa kalimat tersebut relevan dan untuk mengurangi redudansi. Pada penelitian ini, *Sim1* dan *Sim2* merupakan dua fungsi *similarity* yang merepresentasikan kesamaan kalimat pada seluruh dokumen dan memilih masing-masing kalimat untuk dijadikan ringkasan. *Sim1* adalah matrik *similarity* kalimat S_i terhadap *query* yang diberikan oleh user sedangkan *Sim2* adalah matrik *similarity* kalimat S_i terhadap kalimat yang telah diekstrak sebelumnya (Shasa Xie, 2010).

Nilai parameter λ adalah mulai dari 0 sampai dengan 1 (range [0,1]). Pada saat parameter $\lambda = 1$ maka nilai *MMR* yang diperoleh akan cenderung relevan terhadap dokumen asal. Ketika $\lambda = 0$ maka nilai *MMR* yang diperoleh cenderung relevan terhadap kalimat yang diekstrak sebelumnya. Oleh sebab itu sebuah kombinasi linier dari kedua kriteria dioptimalkan ketika nilai λ terdapat pada interval [0,1]. Untuk peringkasan *small* dokumen, seperti pada berita (*news*), menggunakan nilai parameter $\lambda = 0.7$ atau $\lambda = 0.8$, karena akan menghasilkan ringkasan yang baik (Jade Goldstein, 2008).

Untuk mendapatkan hasil ringkasan yang relevan maka harus menetapkan nilai λ ke nilai yang lebih dekat dengan λ. Kalimat dengan nilai *MMR* yang tertinggi akan dipilih berulang kali ke dalam ringkasan sampai tercapai ukuran ringkasan yang diinginkan.

2.3 *Cosine Similarity*

Cosine similarity digunakan untuk menghitung pendekatan relevansi *query* terhadap dokumen. Penentuan relevansi sebuah *query* terhadap suatu dokumen dipandang sebagai pengukuran kesamaan antara vektor *query* dengan vektor dokumen. Semakin besar nilai kesamaan vektor *query* dengan vektor dokumen maka *query* tersebut dipandang semakin relevan dengan dokumen. Saat mesin menerima *query*, mesin akan membangun sebuah vektor Q (w_{q1},w_{q2},…w_{qt}) berdasarkan istilah-istilah pada *query* dan sebuah vektor D (d_{i1},d_{i2},…d_{it}) berukuran t untuk setiap dokumen. Pada umumnya *cosine similarity (CS)* dihitung dengan rumus *cosine measure* (Grossman, 1998).



Gambar 1. Vektor Skalar

$$CS(b_1, b_2) = \frac{\sum_{t=1}^{n} W_{t,b_1} W_{t,b_2}}{\sqrt{\sum_{t=1}^{n} W^2_{t,b_1} \sum_{j=1}^{n} W^2_{t,b_2}}} \quad (2)$$

Keterangan:
T = *term* dalam kalimat
$w_{t,b1}$ = bobot term t dalam blok b_1
$w_{t,b2}$ = bobot term t dalam blok b_2

Model *Dot product:*

$$CS(Q,Di) = (Di)(Q) \quad (3)$$

Model Dice**:**

$$S\langle(Q,Di) = 2(Di)(Q)/(Di2)(Q2) \quad (4)$$

Model *Jaccard:*

$$CS(Q,Di) = (DiQ)/((Di)2 + Q2 - |Di||Q|) \quad (5)$$

Model *Cosine similarity* Larson dan Herast:

$$CosSim(dj,q) = \frac{\vec{dj}.\vec{q}}{|\vec{dj}|.|q|} = \frac{\sum_{i=1}^{t}(W_{ij} \cdot W_{iq})}{\sqrt{\sum_{i=1}^{t} Wij^2 \cdot Wiq^2}} \quad (6)$$

2.4 *Pembobotan* TF-IDF

Pembobotan dapat diperoleh berdasarkan jumlah kemunculan suatu *term* dalam sebuah dokumen *term frequency* (*tf*) dan jumlah kemunculan *term* dalam koleksi dokumen *inverse document frequency* (*idf*). Bobot suatu istilah semakin besar jika istilah tersebut sering muncul dalam suatu dokumen dan semakin kecil jika istilah tersebut muncul dalam banyak dokumen (Grossman, 1998). Nilai *idf* sebuah *term* (kata) dapat dihitung menggunakan persamaan sebagai berikut:

$$IDF = \log\left(\frac{D}{dfi}\right) \quad (7)$$

D adalah jumlah dokumen yang berisi *term* (*t*) dan *dfi* adalah jumlah kemunculan (frekuensi) *term* terhadap D. Adapun algoritma yang digunakan untuk menghitung bobot (W) masing-masing dokumen terhadap kata kunci (*query*), yaitu:

$$W_{d,t} = tf_{d,t} * IDF_t \quad (8)$$

Keterangan :
d = dokumen ke–d
t = *term* ke–t dari kata kunci
tf = *term* frekuensi/frekuensi kata
W = bobot dokumen ke–d terhadap term ke–t

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

2.5 *Text Preprocessing*

Text preprocessing adalah tahapan untuk mempersiapkan teks menjadi data yang akan diolah di tahapan berikutnya. Inputan awal pada proses ini adalah berupa dokumen. *Text preprocessing* pada penelitian ini terdiri dari beberapa tahapan, yaitu: proses pemecahan kalimat, proses *case folding,* proses *tokenizing* kata, proses *filtering,* dan proses *stemming*

2.5.1 Pemecahan Kalimat

Memecah dokumen menjadi kalimat-kalimat merupakan langkah awal tahapan *text preprocessing*. Pemecahan kalimat yaitu proses memecah string teks dokumen yang panjang menjadi kumpulan kalimat-kalimat. Dalam memecah dokumen menjadi kalimat-kalimat menggunakan fungsi **split()**, dengan tanda titik ".", tanda tanya "?" dan tanda tanya "!" sebagai delimiter untuk memotong string dokumen.

2.5.2 *Case Folding*

Case folding adalah tahapan proses mengubah semua huruf dalam teks dokumen menjadi huruf kecil, serta menghilangkan karakter selain a-z.

2.5.3 *Tokenizing*

Tokenizing adalah proses pemotongan string input berdasarkan tiap kata yang menyusunnya. Pemecahan kalimat menjadi kata-kata tunggal dilakukan dengan men-*scan* kalimat dengan pemisah (*delimiter*) *white space* (spasi, tab, dan *newline*).

2.5.4 *Filtering*

Filtering merupakan proses penghilangan *stopword*. *Stopword* adalah kata-kata yang sering kali muncul dalam dokumen namun artinya tidak deskriptif dan tidak memiliki keterkaitan dengan tema tertentu. Didalam bahasa Indonesia *stopword* dapat disebut sebagai kata tidak penting, misalnya "di", "oleh", "pada", "sebuah", "karena" dan lain sebagianya.

Tabel 1. Aturan Pembentukan Partikel Infleksional

Sufiks	Pengganti	Kondisi Tambahan	Contoh
Kah	NULL	NULL	bukukah→buku
lah	NULL	NULL	terimalah→terima
tah	NULL	NULL	apatah→ apa
pun	NULL	NULL	bukupun→buku

Tabel 2. Aturan Pembentukan Kata Ganti Milik Infleksional

Sufiks	Pengganti	Kondisi Tambahan	Contoh
ku	NULL	NULL	bukuku→buku
mu	NULL	NULL	bukumu→buku
nya	NULL	NULL	bukunya→buku

2.5.5 *Stemming*

Stemming merupakan proses mencari akar (*root)* kata dari tiap *token* kata yaitu dengan pengembalian suatu kata berimbuhan ke bentuk dasarnya (*stem*) (Tala, 2003). Pada penelitian ini menggunakan *porter stemming* untuk bahasa indonesia (Tala, 2003). Terdapat lima aturan pada proses *stemming* untuk bahasa Indonesia menggunakan *porter stemmer*, yaitu ada lima aturan tahap dalam proses *stemming* pada bahasa Indonesia, yaitu :

1) Penanganan terhadap partikel inflek sional, yaitu: lah, tah, kah. Contoh: duduklah, makanlah.
2) Penanganan terhadap kata ganti infleksional, yaitu: ku, mu, dan nya. Contoh: sepedaku, bukunya.
3) Penanganan terhadap prefiks derivasional pertama, yaitu : meng dan semua variasinya, peng dan semua variasinya, di, ter, dan ke. contoh : membakar, pengukur, kekasih.
4) Penanganan terhadap prefix derivasional kedua, yaitu : ber dan semua variasinya serta per dan semua variasinya. Contoh: berlari, belajar, perkata.
5) Penanganan terhadap Sufiks derivasional, yaitu kan, am dan i. Contoh: ambilkan, janjian dan dapati.

Karena struktur morfologi bahasa Indonesia yang rumit maka kelima tahap aturan tidak cukup untuk menangani pro-

ses *stemming* bahasa Indonesia. Kesulitan membedakan kata yang mengandung imbuhan baik prefiks maupun sufiks dari kata dasar yang salah satu suku katanya merupakan bagian imbuhan, terutama dengan kata dasar yang mempunyai suku kata lebih besar dari dua.

Tabel 3. Aturan Pembentukan Prefiks Derivasional Pertama

Prefiks	Pengganti	Kondisi Tambahan	Contoh
meng	NULL	NULL	mengukur→ukur
meny	s	V…*	menyapu → sapu
men	NULL	NULL	menduga →duga
men	t	V…	menuduh→tuduh
mem	p	V…	memilah → pilah
mem	NULL	NULL	membaca→ baca
me	NULL	NULL	merusak → rusak
peng	NULL	NULL	pengukur→ ukur
peny	s	V…	penyelam→selam
pen	NULL	NULL	pendaki → daki
pen	t	V…	penari → tari
pem	p	V…	pemilah→pilah
pem	NULL	NULL	pembaca→baca
di	NULL	NULL	diukur→ukur
ter	NULL	NULL	tersipu→sipu
ke	NULL	NULL	kekasih→kasih

Tabel 4. Aturan Pembentukan Prefiks Derivasional Kedua

Prefiks	Pengganti	Kondisi Tambahan	Contoh
Ber	NULL	NULL	berlari→lari
Bel	NULL	Ajar	belajar→ajar
Be	NULL	Kerja	bekerja→kerja
Per	NULL	NULL	perjelas→jelas
Pel	NULL	Ajar	pelajar → ajar
Pe	NULL	NULL	pekerja→kerja

Tabel 5. Aturan Pembentukan Sufiks Derivasional

Sufiks	Pengganti	Kondisi Tambahan	Contoh
kan	NULL	prefiks ¢{ke, peng}	tarikkan→tarik (meng)ambilkan → ambil
An	NULL	prefiks ¢{di, meng, ter}	makanan→makan (per)janjian → janji
I	NULL	prefiks ¢{ber, ke, peng}	tandai → tanda (men)dapati → dapat

Contoh :
- sekolah → sekolah (kata dasar, tidak dilakukan *stemming*)
- duduklah → duduk (dilakukan proses *stemming*)

Berdasarkan urutan tahapan pada penanganan kata berimbuhan, maka terdapat beberapa kemungkinan dalam kesulitan membedakan suatu suku kata merupakan imbuhan atau bagian kata dasar :

1) Kata dasar mempunyai suku kata terakhir (partikel infleksional) serta kata tersebut tidak mendapat imbuhan apapun. Contoh: istilah.
2) Kata dasar mempunyai suku kata terakhir (partikel infleksional) dan mempunyai prefiks. Contoh: bersalah.
3) Kata dasar mempunyai suku kata terakhir (kata ganti infleksional) serta kata dasar tersebut tidak mendapatkan imbuhan apapun. Contoh : bangku.
4) Kata dasar mempunyai suku kata terakhir (kata ganti infleksional) dan mengandung prefiks. Contoh: bertanya.
5) Kata dasar mempunyai suku kata pertama (prefiks derivasional pertama) serta kata dasar tersebut tidak mendapatkan imbuhan apapun. Contoh: diagram, kenang.
6) Kata dasar mempunyai suku kata pertama (prefiks derivasional pertama) dan mempunyai sufiks derivasional. Contoh: disiplinkan, pentungan.
7) Kata dasar mempunyai suku kata pertama (prefiks derivasional kedua) serta kata dasar tersebut tidak mendapatkan imbuhan apapun. Contoh : pelangi, perban.
8) Kata dasar mempunyai suku kata pertama (prefiks derivasional) dan mempunyai sufiks derivasional. Contoh: belakangan, pejamkan.
9) Kata dasar mempunyai suku kata terakhir (sufiks derivasional). Contoh: koleksi, dominan.

Berdasarkan dari permasalahan tersebut, maka dibuat kamus-kamus kecil untuk melengkapi proses *stemming* ini. Terdapat 9 kamus kecil, yaitu :

1) Kamus partikel. Seperti: masalah

2) Kamus partikel berprefiks. Seperti: menikah
3) Kamus milik. Seperti: bangku.
4) Kamus milik berprefiks. Seperti: bersuku.
5) Kamus prefiks 1. Seperti: median.
6) Kamus prefiks 1 bersufiks. Seperti: terapan.
7) Kamus prefiks 2. Seperti : percaya.
8) Kamus prefiks 2 bersufiks. Seperti: perasaan.
9) Kamus sufiks. Seperti: pantai.

Kondisi ukuran adalah jumlah minimum suku kata dalam sebuah kata. Karena dalam bahasa Indonesia, kata dasar setidaknya mempunyai 2 suku kata. Maka kondisi ukuran dalam proses *stemming* bahasa Indonesia adalah dua. Suku kata didefinisikan memiliki satu vokal.

2.6 Tipe *Evaluasi*

Metode untuk mengevaluasi hasil ringkasan merupakan topik yang cukup sulit, baik evaluasi terhadap ringkasan yang dihasilkan dari mesin peringkas otomatis ataupun ringkasan yang manual dibuat oleh *abstractor* yang profesional, dikarenakan tidak terdapat definisi ringkasan ideal. Terdapat dua klasifikasi metode evaluasi (Sparck dan Galliers, 1996), yaitu :

a. Ekstrinsik

Kualitas ringkasan diukur berdasarkan bagaimana ini membantu penyelesaian tugas *user*.

b. Intrinsik

Hanya diukur dari kualitas hasil (*output*) ringkasan yang dihasilkan.

Evaluasi sistem peringkas yang ada saat ini adalah intrinsik. Pengevaluasi menciptakan sekumpulan ringkasan yang manual, masing-masing satu untuk menguji teks. Kemudian membandingkan hasil ringkasan sistem dengan ringkasan ideal. Yang diukur adalah *overlap* dari isi, seringkali disebut dengan *recall* dan *precision* kalimat atau frase, tapi kadang-kadang dengan *overlap* kata tunggal.

$$Precision = \frac{\#\text{kalimat ringkasan sistem} \cap \text{ringkasan manual}}{\sum \text{kalimat ringkasan sistem}} \quad (9)$$

$$Recall = \frac{\#\text{kalimat ringkasan sistem} \cap \text{ringkasan manual}}{\sum \text{kalimat ringkasan manual}} \quad (10)$$

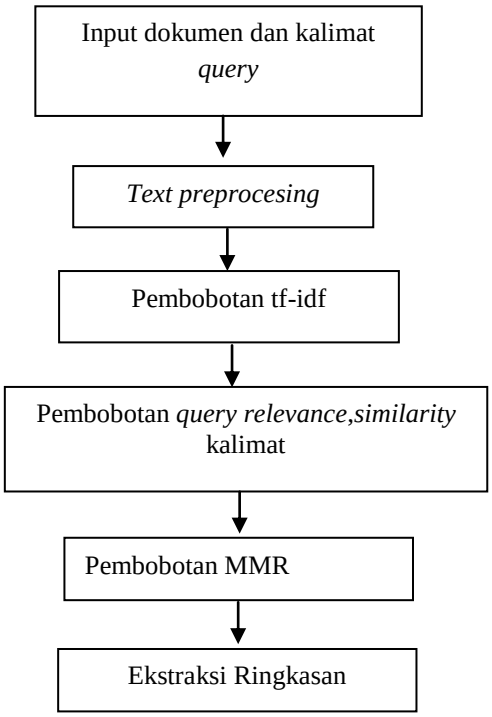
$$f - measure = \frac{2 * precision * recall}{recall + precision} \quad (11)$$

Precision (persamaan 9) dan *recall* (persamaan 10) digunakan untuk mengukur kualitas sebuah ringkasan. Pengukuran *precision* dan *recall* ini sangat dipengaruhi oleh panjang ringkasan manual dan juga panjang ringkasan yang dievaluasi. Akurasi menurun sejalan dengan bertambahnya panjang ringkasan. Sulit untuk mengambil kesimpulan terhadap *performance* sistem dari nilai *precision* dan *recall*. Untuk standarisasi proses evaluasi belum dieksplorasi. Masalah utama dari evaluasi ini adalah sangat nyata, yaitu tidak ada satupun ringkasan yang benar (Edmudson, 1969). Kombinasi antara nilai *recall* dan *precision* menghasilkan *f-measure* (persamaan 11).

3. METODOLOGI

Pada penelitian ini, peringkasan teks otomatis berita yang dibuat merupakan sistem peringkasan dengan inputan berupa *single* dokumen dan secara otomatis menghasilkan ringkasan (*summary*). Proses peringkasan teks otomatis pada penelitian ini terdiri dari: proses *text preprosessing*, pembobotan tf-idf kata, pembobotan *relevance query*, pembobotan *similarity* kalimat, pembobotan MMR dan ekstraksi ringkasan. Gambar 2 menunjukkan bagan proses peringkasan secara umum. Berikut ini alur proses peringkasan sistem:

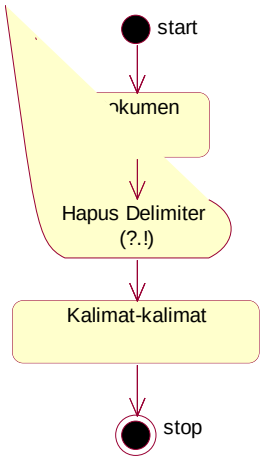
1) User memasukkan teks dokumen yang akan diringkas dan memasukkan kalimat *query*. Proses awal peringkasan *user* memasukkan teks dokumen dan *query* sesuai Tabel 6.
2) Sistem melakukan penyiapan teks (*text preprosessing)* dokumen yang terdiri dari tahap pemecahan kalimat, *case folding*, *tokenizing* kata, *filtering* dan *stemming*.



Gambar 2. Proses Peringkasan

Tabel 6. Teks Asal dan *Query*

Isi: Tempo *interaktif*, london: hubungan mesra antara chelsea dengan didier drogba tampaknya akan segera berakhir. Striker pantai gading itu kemungkinan akan mendapat denda 100,000 poundstering (rp 1,5 miliar) akibat mengkritik klub yang menurutnya tidak mendukung performanya di stamford bridge musim ini. Nilai denda itu setara gaji sepekan yang diterima drogba. Dia juga mengaku tidak berminat untuk kembali bermain setelah dibekap cedera lutut dan mengkritik gaya pemilihan pemain yang ditunjukkan luiz felipe scolari. Komentarnya jelas membuat big phil dan ceo *the blues* peter kenyon marah. Atas aksinya ini klub telah memanggil drogba yang mungkin memilih mengakhiri kariernya bersama chelsea yang dimulainya sejak 2004 setelah hengkang dari marseille. ***Query:*** chelsea denda drogba



Gambar 3. *Activity Diagram* Pemecahan Kalimat

Tabel 7. Pemecahan Kalimat

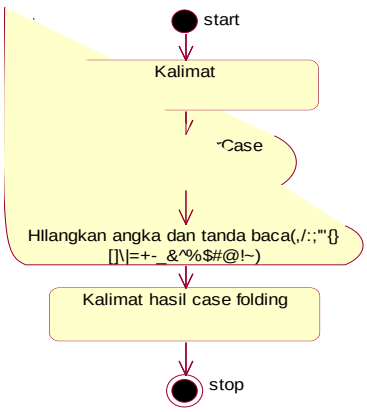
No	Kalimat
1	**Tempo** *interaktif*, London: hubungan mesra antara Chelsea dengan Didier Drogba tampaknya akan segera berakhir
2	Striker pantai gading itu kemungkinan akan mendapat denda 100,000 poundstering (rp 1,5 miliar) akiba mengkritik klub yang menurutnya tidak mendukung performanya di Stamford Bridge musim ini
3	Nilai denda itu setara gaji sepekan yang diterima Drogba
4	Dia juga mengaku tidak berminat untuk kembal bermain setelah dibekap cedera lutut dan mengkritik gaya pemilihan pemain yang ditunjukkan Luiz Felipe Scolari
5	Komentarnya jelas membuat big phil dan ceo *the blues* Peter Kenyon marah
6	Atas aksinya ini klub telah memanggil Drogba yang mungkin memilih mengakhiri kariernya bersama Chelsea yang dimulainya sejak 2004 setelah hengkang dari Marseille

3.1 Pemecahan Kalimat

Tahap pemecahan kalimat adalah memecah string dokumen menjadi kumpulan kalimat dengan menghilangkan tanda tanda akhir kalimat *(delimiter)*. Tanda baca akhir kalimat seperti tanda titik ".", tanda tanya "?", dan tanda seru "!". Gambar 3 adalah *activity diagram* pemecahan kalimat. Tabel 7 merupakan hasil proses pemecahan dokumen menjadi kumpulan kalimat dengan inputan artikel (tabel 6).

3.2 *Case Folding*

Tahap ini, kumpulan kalimat hasil pemecahan diubah menjadi huruf kecil (*lower case*), menghilangkan angka, tanda baca maupun simbo dan hanya menerima karakter UTF8 dengan kode 0061- 007A). Alur *activity diagram case folding* pada Gambar 4.



Gambar 4. *Activity Diagram Case Folding*

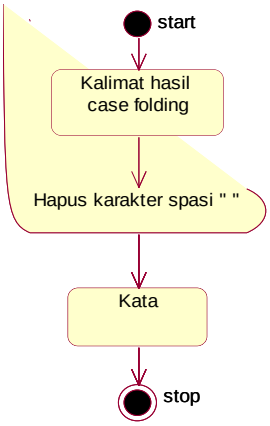
Tabel 8. *Case Folding*

No	Kalimat
1	**tempo** *interaktif* london hubungan mesra antara chelsea dengan didier drogba tampaknya akan segera berakhir
2	striker pantai gading itu kemungkinan akan mendapat denda poundstering rp miliar akibat mengkritik klub yang menurutnya tidak mendukung performanya di stamford bridge musim ini
3	nilai denda itu setara gaji sepekan yang diterima drogba
4	dia juga mengaku tidak berminat untuk kembali bermain setelah dibekap cedera lutut dan mengkritik gaya pemilihan pemain yang ditunjukkan luiz felipe scolari
5	komentarnya jelas membuat big phil dan ceo *the blues* peter kenyon marah
6	atas aksinya ini klub telah memanggil drogba yang mungkin memilih mengakhiri kariernya bersama chelsea yang dimulainya sejak setelah hengkang dari marseille

Tabel 8 merupakan hasil case folding dari data di tabel 7. Huruf awal dari setiap kalimat diubah menjadi huruf kecil. Pada kalimat pertama dan kedua string angka, tanda baca titik dua ":", tanda baca koma ",", tanda kurung buka dan tutup "()" dihilangkan.

3.3 *Tokenizing*

Kumpulan kalimat hasil dari *case folding* kemudian dilakukan proses *tokenizing* kata yaitu menghilangkan karakter pemisah (*delimiter*) yang menyusunnya berupa karakter spasi (UTF8 kode 0020). Alur *activity diagram tokenizing* dapat ditunjukkan pada Gambar 5. Berdasarkan tabel 9, proses *tokenizing* menghasilkan token kata sejumlah 56 kata.



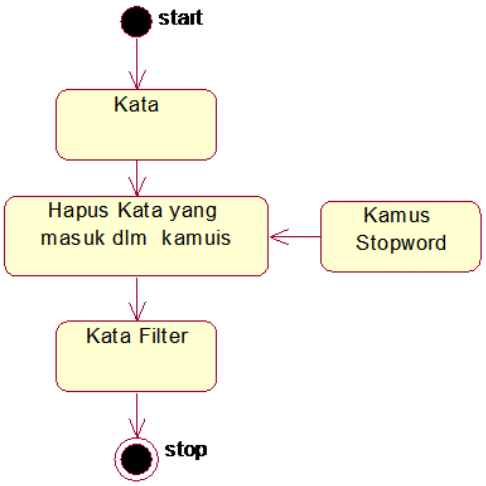
Gambar 5. *Activity Diagram Tokenizing*

Tabel 9. Hasil *Tokenizing*

Kata	Kata	Kata
aksinya	Drogba	marseille
bermain	Felipe	memanggil
berminat	Gading	memilih
big	Gaji	mendukung
blues	Gaya	mengakhiri
bridge	hengkang	mengkritik
cedera	interaktif	mesra
ceo	kariernya	miliar
chelsea	kenyon	musim
denda	Klub	nilai
dibekap	komentarnya	pantai
didier	London	pemain
dimulainya	Luiz	pemilihan
diterima	Lutut	performanya
ditunjukkan	Marah	peter
phil	Scolari	stamford
poundstering	sepekan	striker
rp	Setara	tempo
hubungan	Antara	dengan
akan	Segera	itu
kemungkinan	Akibat	yang
menurutnya	Tidak	di
ini	Dia	juga
setelah	Dan	jelas
the	Atas	telah
mungkin	bersama	sejak

3.4 *Filtering*

Pada tahap ini, dilakukan pembuangan kata-kata yang dianggap kurang penting. *Stopword* adalah kata-kata yang kurang deskriptif yang dapat dibuang dalam pendekatan *bag-of-words*. Pembuangan *stopword dilakukan dengan* mengecek kamus *stopword*. Jika terdapat kata yang sama dengan kamus maka akan dihapus. Alur *activity filtering* pada Gambar 5. Kata hasil token dicek terlebih dahulu untuk dicocokkan dengan kamus *stopword*. Jika dalam pencocokan terdapat kata yang sama dalam kamus maka kata tersebut dihilangkan. Berdasarkan Tabel 10 kata-kata yang termasuk dalam *stopword* adalah : antara, akan, dengan, hubungan, segera, itu, kemungkinan, akibat, yang, tidak, di, menurutnya, ini, dia, juga, setelah, dan, jelas, the, atas, telah, mungkin, bersama dan sejak.



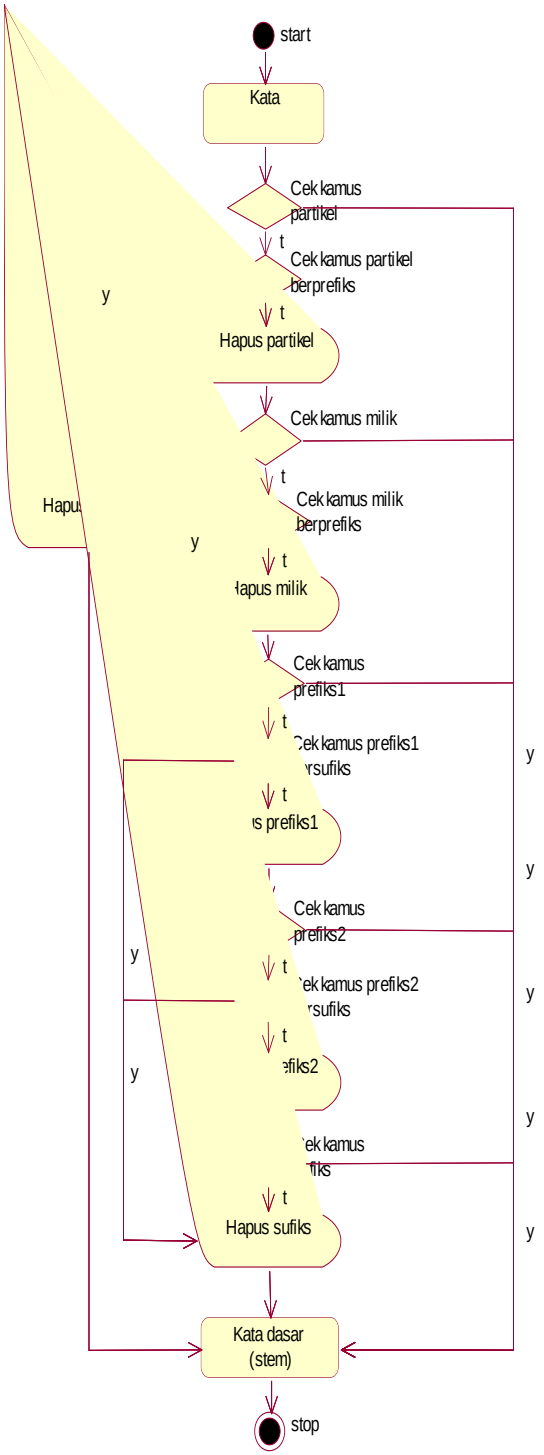
Gambar 6. *Activity Diagram Filtering*

Tabel 10. Hasil *Filtering*

Kata	Kata	Kata
aksinya	drogba	marseille
bermain	felipe	memanggil
berminat	gading	memilih
big	gaji	mendukung
blues	gaya	mengakhiri
bridge	hengkang	mengkritik
cedera	interaktif	mesra
ceo	kariernya	miliar
chelsea	kenyon	musim
denda	klub	nilai
dibekap	komentarnya	pantai
didier	london	pemain
dimulainya	luiz	pemilihan
diterima	Lutut	performanya
ditunjukkan	marah	peter
phil	scolari	stamford
poundstering	sepekan	striker
rp	setara	tempo

3.5 *Stemming*

Hasil *filtering* kemudian di-*stemming* untuk mendapatkan kata dasar (*root*). Proses *stemming* menggunakan bantuan kamus-kamus kecil dengan untuk membedakan suatu kata yang mengandung imbuhan baik prefiks maupun sufiks yang salah satu suku katanya merupakan bagian dari imbuhan, terutama dengan kata dasar yang mempunyai suku kata lebih besar dari dua. Alur *activity diagram stemming* sesuai Gambar 7.



Gambar 7. *Activity Diagram Stemming*

Dari proses stemming terdapat kata-kata yang mengalami penghilangan imbuhan, baik prefiks maupun sufiks. Kata-kata yang telah dihilangkan prefiks maupun sufiks dapat dilihat pada Tabel 11. Di antara kata-kata tersebut adalah : aksi, main, panggil, pilih, bekap,

performa, dukung, akhir, minat, kritik, tunjuk dan terima.

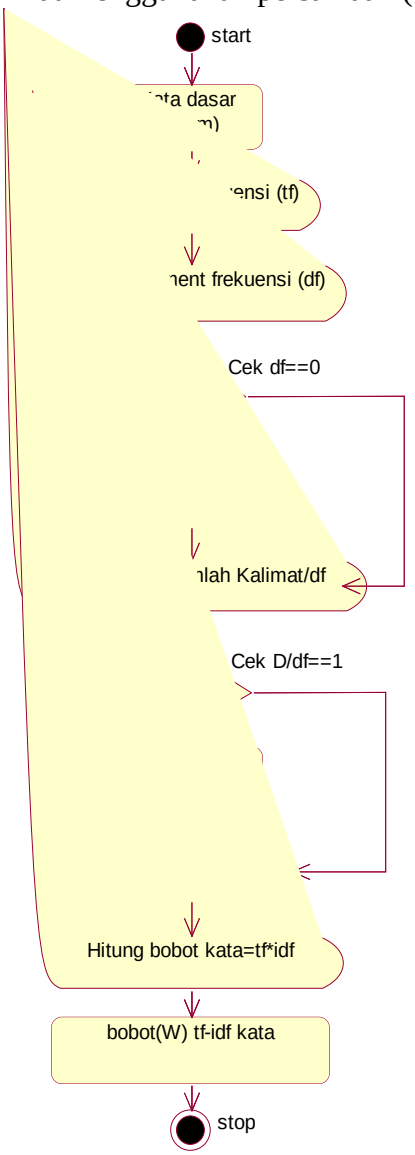
Tabel 11. Hasil *Stemming*

Kata	Kata	Kata
aksi	drogba	marseille
main	felipe	panggil
minat	gading	pilih
big	gaji	dukung
blues	gaya	akhir
bridge	hengkang	kritik
cedera	interaktif	mesra
ceo	karier	miliar
chelsea	kenyon	musim
denda	klub	nilai
bekap	komentar	pantai
didier	london	main
mulai	luiz	pilih
terima	lutut	performa
tunjuk	marah	peter
phil	scolari	stamford
poundstering	sepekan	striker
rp	setara	tempo

Hasil proses *text preprosessing* dilakukan pembobotan tf-idf. Pembobotan secara otomatis biasanya berdasarkan jumlah kemunculan suatu kata dalam sebuah dokumen (*term frequency*) dan jumlah kemunculannya dalam koleksi dokumen (*inverse document frequency*). Bobot kata semakin besar jika sering muncul dalam suatu dokumen dan semakin kecil jika muncul dalam banyak dokumen. Pembobotan tf-idf dilakukan untuk pembobotan tahap selanjutnya, yaitu untuk menghitung bobot *query relevance* dan bobot *similarity* kalimat. Alur *activity* pada bobot tf-idf untuk *relevance query* adalah sama seperti *activity* pada bobot tf-idf untuk *similarity* kalimat. Alur *activity* bobot tf-idf pada Gambar 8.

Perhitungan bobot *query relevance* merupakan bobot hasil perbandingan kemiripan (*similaritas*) antara *query* yang dimasukkan oleh *user* terhadap keseluruhan kalimat. Sedangkan bobot *similarity* kalimat, merupakan bobot hasil perbandingan kemiripan antar kalimat. Alur *activity diagram bobot query relevance* pada Gambar 9 dan bobot *similarity* pada Gambar 10.

Tabel 12 adalah hasil perhitungan bobot *query relevance* yaitu menghitung bobot kemiripan antara *query* dengan kalimat dalam dokumen sesuai persamaan (6). Perhitungan bobot *query relevance* ini menggunakan metode *cosine similarity* dengan menghitung *cosinus* sudut dari dua *vector*, yaitu W, bobot dari tiap kalimat dan W (bobot) *query*. Tabel 13 adalah hasil pembobotan *similarity* kalimat menggunakan persamaan (6),



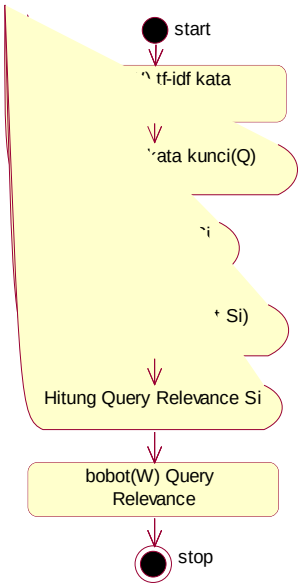
Gambar 8. *Activity Diagram* bobot tf-idf

Tabel 12. Bobot *query relevance*

S1	S2	S3	S4	S5	S6
0.202	0.109	0.235	0	0	0.205

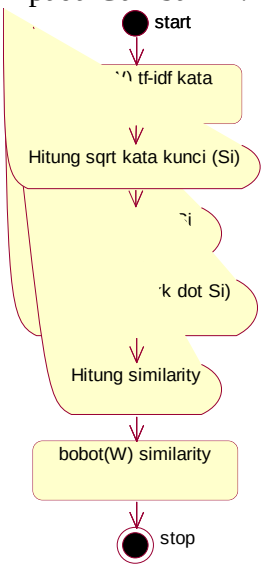
Tabel 13. Bobot *similarity* kalimat

	S1	S2	S3	S4	S5	S6
S1	**1**	0	0.023	0	0	0.071
S2	0	**1**	0.044	0.028	0	0.038
S3	0.023	0.044	**1**	0	0	0.024
S4	0	0.028	0	**1**	0	0.038
S5	0	0	0	0	**1**	0
S6	0.071	0.038	0.024	0.038	0	**1**

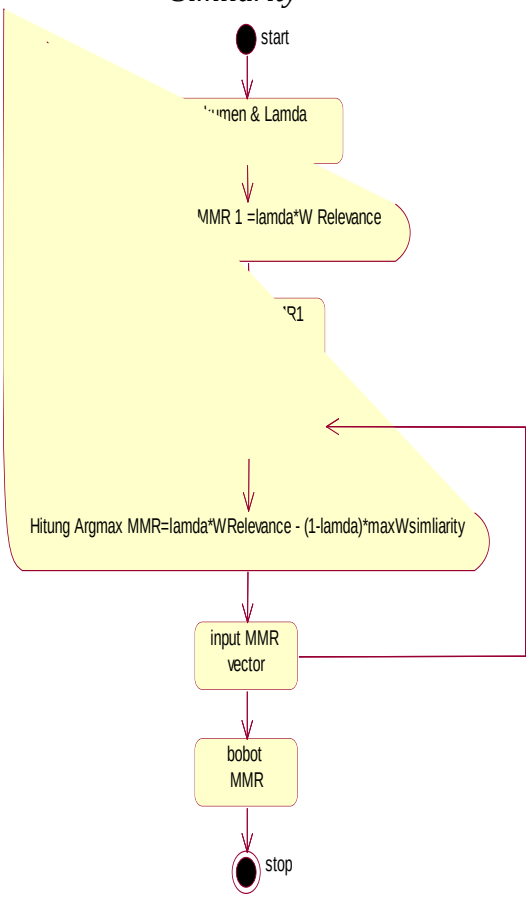


Gambar 9. *Activity Diagram* Bobot *Query Relevance*

Pembobotan *MMR* kalimat menggunakan algoritma *maximum marginal relevance*, kalimat dirangking sebagai tanggapan terhadap *query* yang diberikan oleh *user*. Perhitungan *MMR* dilakukan dengan perhitungan iterasi antara kombinasi dua matrik *cosine similarity* yaitu *query relevance* dan *similarity* kalimat. Pengguna yang menginginkan ruang sampel informasi disekitar *query*, maka harus menetapkan λ pada nilai yang lebih rendah. Sedangkan bagi pengguna yang ingin fokus untuk memperkuat dokumen-dokumen lebih relevan, maka harus menetapkan λ pada nilai yang lebih dekat dengan λ. Kalimat dengan nilai *MMR* tertinggi dari setiap perhitungan iterasi akan diambil, kemudian dipilih sebagai ringkasan. Iterasi berhenti pada saat hasil *MMR* maksimum sama dengan 0. Alur *activity* bobot *MMR* pada Gambar 11.



Gambar 10. *Activity Diagram* Bobot *Similarity*



Gambar 11. Activity Diagram Bobot MMR

Tabel 14. Hasil Iterasi MMR

Iterasi ke	S1	S2	S3	S4	S 5	S6
1	0.141	0.076	**0.165**	0	0	0.144
2	0.134	0.063	-	0	0	**0.137**
3	**0.120**	0.065	-	0	0	-
4	-	**0.063**	-	0	0	-
5	-	-	-	- 0.011	**0**	-

Tabel 15. Hasil bobot MMR maksimum iterasi MMR

Iterasi ke	Kalima t	Bobot ArgMax MMR
MMRMAX(1)	S3	0.165
MMRMAX(2)	S6	0.137
MMRMAX(3)	S1	0.120
MMRMAX(4)	S2	0.063

Tabel 16. Hasil Ekstraksi

Kalimat ke	Kalimat
S3	Nilai denda itu setara gaji sepekan yang diterima Drogba
S6	Atas aksinya ini klub telah memanggil Drogba yang mungkin memilih mengakhiri kariernya bersama Chelsea yang dimulainya sejak 2004 setelah hengkang dari Marseille
S1	**Tempo***interaktif*, London: hubungan mesra antara Chelsea dengan Didier Drogba tampaknya akan segera berakhir
S2	Striker pantai gading itu kemungkinan akan mendapat denda 100,000 poundstering (rp 1,5 miliar) akibat mengkritik klub yang menurutnya tidak mendukung performanya di Stamford Bridge musim ini

Tahap terakhir adalah ekstraksi ringkasan dari hasil bobot *MMR* kalimat. Dari hasil perhitungan mmr diketahui kalimat yang menjadi ringkasan berdasarkan urutan bobot *MMR* kalimat tertinggi dapat dilihat pada tabel 16.

4. HASIL UJI COBA

Data yang digunakan sebagai uji coba sejumlah 30 berita. Data uji coba diambil secara acak dari koran berita *online* Tempo Interaktif yang diunduh dari bulan Januari 2009 sampai Juni 2009. Uji coba dilakukan dengan menguji tiap teks berita. Pengujian dilakukan dengan memasukkan isi dari teks berita dan *query*. *Query* merupakan judul dari berita. Kalimat-kalimat yang terambil sebagai ringkasan merupakan kalimat yang merepresentasikan *query*, karena memiliki kesamaan kata-kata pada kalimat *query*, dan memiliki bobot *MMR* maksimum antara nilai bobot maksimum 1 hingga bobot minimum 0. Semakin banyak kata-kata yang sama dengan *query* maka semakin besar peluang kalimat terambil sebagai ringkasan. Tabel 17 menunjukkan hasil keseluruhan uji coba yang dilakukan pada sistem peringkasan teks otomatis.

Kualitas ringkasan sistem diukur dengan membandingkan dengan ringkasan manual. Ringkasan manual diperoleh dari mayoritas kalimat yang dipilih oleh enam orang responden. Kualitas ringkasan dihitung dengan nilai *recall*, *precision* dan *f-measure* menggunakan persamaan (9), (10) dan (11).

Tabel 17. Hasil Ringkasan Sistem

Nomor Berita	Jumlah Kalimat	Ringkasan (No. Kalimat)
1	4	1,5,6,7
2	4	1,4,6,11
3	8	1,2,3,4,7,8,9,13
4	7	1,2,3,4,8,10,11
5	3	1,7,10
6	5	1,4,7,12,13
7	3	1,3,6
8	2	1,10
9	4	1,2,4,10
10	5	1,2,3,6,9
11	4	1,2,5,8
12	6	1,2,3,4,5,9
13	7	1,2,3,4,5,10,11
14	8	1,2,3,5,6,7,8,9
15	2	1,3
16	7	1,3,4,6,7,8,12
17	4	1,2,3,5
18	7	1,3,4,5,8,10,17
19	4	1,2,5,6
20	6	1,6,7,9,12,14
21	9	1,3,4,5,9,10,11,12,13
22	4	1,2,3,6
23	4	1,3,4,9
24	8	1,2,3,5,10,12,13,16
25	3	1,7,11
26	5	2,3,5,6,7
27	8	1,2,3,4,6,7,9,10
28	4	1,2,7,8
29	6	2,7,8,9,13,15
30	3	1,2,5

Hasil perhitungan evaluasi diurutkan berdasarkan nilai *recall, precision* dan *f-measure* dari persentase yang tertinggi ke urutan terendah. Dari tabel tersebut menunjukkan bahwa berita ketiga belas memiliki nilai *recall, precision* dan *f-measure* persentase paling tinggi yaitu dengan nilai *recall* 100%, *precision* 100% dan *f-measure* 100%, sedangkan persentase terendah pada berita kedelapan dengan nilai *recall* 10%, *precision* 50% dan *f-measure* 17%. Berdasarkan hasil pengujian dan evaluasi ringkasan dapat diketahui bahwa hasil evaluasi antara ringkasan sistem dengan ringkasan manual menghasilkan nilai rata-rata *recall* sebesar 60%, *precision* 77%, dan *f-measure* sebesar 66%. Pada hasil evaluasi antara ringkasan sistem dengan ringkasan manual, terdapat lima berita dengan nilai persentase *f-measure* rendah dibawah 50% yaitu berita nomor 9 dengan 46%, berita nomor 5 dengan 46%, berita nomor 6 dengan 31%, berita nomor 15 dengan 25% dan nomor 8 dengan 17%. Hal ini disebabkan oleh jumlah kalimat yang sama (*overlap*) adalah kecil atau sedikit sehingga menyebabkan hasil *f-measure*nya rendah. Jika semakin *overlap* kalimatnya yang terpilih banyak maka hasil dari *f-measure*nya tinggi.

5. KESIMPULAN

Kesimpulan yang didapat dari penelitian adalah : Metode *maximum marginal relevance* dapat digunakan untuk meringkas *single* dokumen secara otomatis dengan menggunakan judul artikel berita sebagai *query*, hasil dari uji coba yang dilakukan menghasilkan rata-rata *recall* 60%, *precision* 77%, dan *f-measure* 66% berdasarkan perbandingan sistem dengan ringkasan manual.

6. SARAN

Hasil ringkasan merupakan kalimat yang memiliki kemiripan dengan *query* dan berdasarkan urutan bobot *MMR*. Hasil perbandingan terhadap ringkasan manual terdapat beberapa artikel berita yang memiliki nilai *f-measure* rendah, karena *query* yang dimasukkan tidak menggambarkan isi, sehingga alimat yang terambil tidak sesuai urutan kalimat yang baik. Pengembangan lebih lanjut disarankan untuk menggunakan *generator* judul sebagai *query* untuk mendapatkan *f-measure* yang tinggi, kalimat ringkasan yang ditampilkan urut berdasarkan sistematika yang baik.

Tabel 18. Hasil Perbandingan Sistem dengan Ringkasan Manual

Nomor berita	Overlap Kalimat	*Recall*	*Precision*	*F-measure*
1	4	80%	100%	89%
2	3	43%	75%	55%
3	7	88%	88%	88%
4	6	86%	86%	86%
5	3	30%	100%	46%
6	3	38%	60%	46%
7	2	50%	67%	57%
8	1	10%	50%	17%
9	3	33%	75%	46%
10	5	63%	100%	77%
11	4	80%	100%	89%
12	4	80%	67%	73%
13	7	100%	100%	100%
14	6	86%	75%	80%
15	1	17%	50%	25%
16	5	71%	71%	71%
17	3	75%	75%	75%
18	5	50%	71%	59%
19	3	60%	75%	67%
20	5	63%	83%	71%
21	6	75%	67%	71%
22	3	75%	75%	75%
23	3	60%	75%	67%
24	6	60%	75%	67%
25	3	43%	100%	60%
26	4	67%	80%	73%
27	5	71%	63%	67%
28	3	60%	75%	67%
29	4	40%	67%	50%
30	2	50%	67%	57%

DAFTAR PUSTAKA

Das and Martins. 2007. *A Survey on Automatic Text Summarization.* Language Technologies Institute Carnegie Mellon University

Erwin A.H., Muhammad. 2005. *Sistem Pengidentifikasi Otomatis Pokok Kalimat Suatu Paragraf Dalam Dokumen Ekspositori Dengan Model Ruang Vektor*. Laboratorium Pemrograman dan Informatika Teori. Yogyakarta: Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Islam Indonesia.

Firmin, T and M.J Chrzanowski. 1999. *An Evaluation of Automatic Text Summarization System*. The MIT Press: Cambrige

Golstein, Jade and Carbonell, Jaime. 1998. *Summarization: Using MMR for Diversity Based-Reranking and Evaluating Summaries*. Langauge Technologies Istitute. Carnegie Mellon University.

Golstein, Jade. 2008. *Genre Oriented Summarization*. Thesis. Pittsburgh: Language Technologies Institute School of Computing Carnegie Mellon University.

Garcia, E. 2006. *The Classic Vector Space Model***.** http://www.miislita.com/information-retrieval-tutorial/. Diakses tanggal 25 Maret 2011.

Grossman, D., dan Ophir, F. 1998. *Information Retrieval: Algorithm and Heuristics.* Kluwer Academic Publisher.

Hovy, E. and Lin, C. Y. (1999). *Automated text summarization in summarist*. In Mani, I. and Maybury, M. T., editors, Advances in Automatic Text Summarization, pages 81-94. MIT Press

Husni, Muchammad dan Zaman, Badrus. 2005. *Perangkat lunak Peringkas Dokumen Berbahasa Indonesia dengan Hybrid Stemming*. Surabaya: Teknik Informatika Fakultas Teknologi Informasi Institut Teknologi Sepuluh Nopember.

Jones, K.S, dan Galliers, J.R. 1996. *Evaluating Natural Language Processing System : An Analysis and Review*. New York: Springer.

Larson and Hearst. 2000. *Computing Relevance, Similarity: The Vector Space Model.* UC Berkeley. http://www.sims.berkeley.edu/courses/is202/f00/. Diakses tanggal 25 Maret 2011.

Mani, I. and Maybury. 1999. *Advance in Automatic Text Summarization*. The MIT Press: Cambrige.

Mani, Inderjeet. 2001. *Summarization Evaluation: An Overview*. The MITRE Corporation, W640 11493 Sunset Hills Road Reston, VA 20190-5214 USA.

Radlinski, Filip. 2008. *Learning to Rank (part 2).* In NESCAI 2008 Tutorial. Computer Science Cornell University. http://radlinski.org/papers/LearningToRank_NESCAI08.pdf. Diakses tanggal 20 Maret 2011.

Sartuni, Rasjid dkk. 1984. *Bahasa Indonesia untuk Perguruan Tinggi*. Jakarta: Nina Dinamika

Tala, Fadillah Z. 2003. *A Study of Stemming Efects on Information Retrieval in Bahasa Indonesia.* Institute for Logic, Language and Computation Universite itvan Amsterdam The Netherlands. www.illc.uva.nl/publications/ResearchReport/Mol-200302.text.pdf. Diakses tanggal 25 Februari 2011.

Xie, Shasha. 2010. *Automatic Extractive Summarization Meeting Corpus*. Dissertation. Dallas: The University of Texas at Dallas.