BAB 2
LANDASAN TEORI

2.1. Peringkasan Teks Otomatis

Peringkasan teks otomatis (*Automated Text Summarization*) merupakan pengertian dari sebuah aplikasi berbasis komputer yang memiliki fungsi utama untuk menghasilkan ringkasan dari sebuah artikel tanpa menghilangkan tujuan utama dari artikelnya. Tujuan utamanya adalah untuk menunjukkan beberapa kalimat utama yang digabung menjadi sebuah ringkasan, dengan harapan dapat mengurangi waktu untuk membaca dan memahami isi dari bacaan.

Sebuah sistem peringkasan teks akan diberikan masukkan berupa teks, kemudian melakukan peringkasan dan menghasilkan keluaran berupa teks yang lebih singkat dari teks aslinya. Hasil dari peringkasan tersebut mengandung poin – poin penting atau informasi utama yang ada dari teks asalnya.

Terdapat dua pendekatan pada teknik peringkasan teks, yaitu ekstraksi (*shallower approaches*) dimana pada teknik ini sistem menyalin unit – unit teks yang dianggap paling penting atau memiliki informasi – informasi utama dari teks sumber yang dijadikan bahan ringkasan dan satu lagi adalah teknik abstraksi (*deeper approaches*) dimana teknik ini mengambil intisari dari teks sumber, kemudian membuat sebuah ringkasan dengan menciptakan kalimat – kalimat baru yang merepresentasikan intisari teks sumber ke dalam bentuk berbeda dengan kalimat – kalimat pada teks sumber. Berdasarkan sumbernya, sebuah ringkasan dapat dihasilkan dari satu sumber (*single document*) atau berasal dari banyak sumber (*multy document*) [8].

Pada penelitian ini tipe peringkasan teks yang akan digunakan adalah ekstraktif dengan memanfaatkan aplikasi yang dijalankan pada komputer untuk menghasilkan sebuah ringkasan dari dokumen aslinya. Teks yang dijadikan sumber berasal dari satu sumber (*single document*).

2.2. *Preprocessing*

Preprocessing adalah sebuah tahapan yang bisa membuat teks menjadi data yang bisa diolah pada proses berikutnya. *Preprocessing* dalam peringkasan teks otomatis memiliki dua tujuan, yaitu untuk normalisasi kata-kata dalam teks serta pengurangan kosa kata yang akan diproses untuk memudahkan proses peringkasan[6].

Proses-proses yang akan digunakan pada tahap *preprocessing* pada peringkasan teks otomatis meliputi *case folding, splitting, tokenization* dan *stopword removal.*

2.2.1 Case Folding

Case folding/capitalization adalah proses mengubah semua huruf dalam teks dokumen menjadi huruf kecil atau huruf capital[7]. *Case folding* dilakukan karena dokumen mengandung berbagai variasi dari bentuk huruf. Variasi huruf harus diseragamkan untuk menghilangkan *noise* pada saat pengambilan informasi. Pada penelitian ini, *case folding* merupakan tahapan awal dari *preprocessing*.

Setelah melakukan *case folding*, dilakukan penyatuan kata entitas yang mempunyai >=2 kata pada entitas tersebut. Hal ini dilakukan supaya mepermudah untuk proses ekstraksi fitur kalimat pada urutan 4.

2.2.2. Tokenisasi kalimat

Pemisahan kalimat merupakan proses pemisahan teks pada dokumen menjadi kumpulan kalimat. Teknik yang digunakan dalam pemisahan kalimat adalah memisahkan kalimat dengan tanda titik (.) sebagai *delimeter*.

2.2.3. Filtering

Tahap *Filtering* adalah tahap yang mengacu pada proses untuk menentukan istilah yanag akan mempresentasikan atau menggambarkan isi dokumen dan membedakan dokumen satu dan yang lainnya pada suatu koleksi. Jadi *Filtering* merupakan proses dimana teks selain karakter “a” sampai “z” dan spasi akan dihilangkan dan hanya menerima spasi. Tahap-tahapan yang dilakukan yaitu:

1. Membaca data masukan yang dilakukan.

2. Mengecek apakah ada huruf selain “a” sampai ”z”, “ ” (spasi) dan “.” (titik). Jika tidak ditemukan, maka algoritma berakhir. Jika ada.
3. Hapus karakter yang bukan termasuk “a” sampai “z”, “ ” (spasi) dan “.” (titik).
4. Algoritma selesai.

2.2.4. Tokenisasi kata

Tokenisasi kata adalah proses pemotongan teks berupa kalimat untuk mendapatkan bagian-bagian atau *token-token* tertentu. Secara umum pemecahan kalimat menjadi kumpulan *token* dilakukan dengan melakukan pemindaian kalimat dengan pemisah (*delimiter*) serta *white space* (spasi, *tab*, dan *newline*)[6].

2.2.5. Stopword Removal

Stopword removal merupakan proses menghilangkan kata - kata yang sering kali muncul dalam dokumen namun arti dari kata-kata tersebut tidak deskriptif dan tidak memiliki keterkaitan dengan tema tertentu. *Stopword* dapat berupa kata depan, kata penguhubung, dan kata pengganti contohnya adalah dan, yang, atau, ini, itu dan lainnya[8]. Cara penentuan kata yang akan di hapus yaitu dengan membawa daftar kata (*stoplist*) dari penelitian yang berkaitan dengan kata pada bahasa Indonesia yang telah dilakukan [9].

2.3. Ekstraksi Fitur

Fitur ekstraksi adalah cara untuk menghitung skor tiap-tiap kalimat dalam dokumen[5]. Untuk setiap kalimat dalam dokumen, skor kalimat dihitung berdasarkan *fitur ekstraksi* dimana nilai dari tiap-tiap *fitur* dinormalisasikan sehingga berada dalam range[0,1]. Normalisasi ini dilakukan agar nilai dari tiap-tiap *fitur ekstraksi* tidak memiliki gap atau selisih yang besar.

Fitur ekstraksi yang digunakan dalam penelitian ini berdasarkan :

1. Panjang Kalimat (f1)

Kalimat yang paling pendek tidak akan dimasukkan ke dalam kandidat ringkasan. Fitur ini di hitung dengan membagi jumlah kata-kata dalam kalimat terhadap jumlah kata dari kalimat terpanjang [7].

$$\frac{Jumlah\ kata\ pada\ kalimat\ ke{-}i}{jumlah\ kata\ terpanjang\ pada\ kalimat} \qquad (2.1)$$

2. Posisi Kalimat (f2)

Fitur ini mengasumsikan kalimat pertama pada setiap paragraf merupakan kalimat yang paling penting. Pada fitur ini akan diurutkan N kalimat pertama [7].

$$\frac{n-i}{n} \quad (2.2)$$

Ketengan :

n = jumlah kalimat keseluruhan

i = perulangan dimulai dari 0

3. Fitur Data Numerik (f3)

Biasanya kalimat yang mengandung data numerik merupakan kalimat penting dan biasanya kalimat tersebut masuk dalam ringkasan [7].

$$\frac{Banyak\ angka\ dalam\ suatu\ kalimat\ ke-i}{Banyak\ kata\ dalam\ kalimat\ ke-i} \quad (2.3)$$

4. Fitur Kata – kata *Tematic* dalam Kalimat (f4)

Fitur ini menghitung kemunculan relatif kata kunci pada suatu kalimat. Biasanya kalimat yang memiliki relatif kata kunci yang baik, merupakan kalimat ringkasan [7].

$$\frac{Jumlah\ kata\ yang\ sama\ dengan\ tematic}{Banyak\ kata\ dalam\ kalimat\ ke-i} \quad (2.4)$$

5. Fitur Kalimat yang Menyerupai dengan Judul(f5)

Kalimat yang menyerupai judul adalah kalimat yang memiliki *vocabulary overlap* antara kalimat dengan judul [7].

$$\frac{Banyak\ kata\ dalam\ kalimat\ ke-i\ yang\ menyerupai\ judul\ dokumen}{Banyak\ kata\ pada\ kalimat\ ke-i}$$

(2.5)

6. Fitur Kemiripan Kalimat dengan Kumpulan Kalimat Lain (f6)

Kemiripan kalimat dapat dilihat dari *vocabulary overlap* antara kalimat dengan kalimat yang lain, untuk mempermudah maka kata yang dilihat hanya kata kunci [7].

$$\frac{Banyak\ kata\ dalam\ kalimat\ ke-i\ yang\ sama\ dengan\ kata\ pada\ kalimat\ lain}{Banyak\ kata\ pada\ kalimat\ ke-i}$$

(2.6)

7. Fitur Ikatan Leksikal dengan Kalimat Sebelumnya (f7)

Ikatan leksikal antara kalimat dengan kalimat sebelumnya didefinisikan sebagai kata (*stem*) yang muncul dalam kedua kalimat tersebut, nilai akan 1 apabila memiliki hubungan *lexical*, 0 jika tidak punya [7].(2.7)

8. Fitur Ikatan Leksikal dengan Kalimat Sesudahnya (f8)

Ikatan leksikal antara kalimat dengan kalimat sesudahnya didefinisikan sebagai kata (*stem*) yang muncul dalam kedua kalimat tersebut, nilai akan 1 apabila memiliki hubungan *lexical*, 0 jika tidak punya [7].(2.8)

2.4. Simple Additive Weighting

Metode *Simple Additive Weighting* sering juga dikenal dengan istilah metode penjumlahan terbobot. Konsep dasar metode *Simple Additive Weighting* adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut. Metode *Simple Additive Weighting* disarankan untuk menyelesaikan masalah penyeleksian dalam sistem pengambilan keputusan multi proses. Metode *Simple Additive Weighting* adalah metode yang banyak digunakan dalam pengambilan keputusan yang memiliki banyak atribut. Metode *Simple Additive Weighting* membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang didapat diperbandingkan dengan semua rating alternatif yang ada. Formula untuk melakukan normalisasi ditunjukan pada Persamaan 2.9 (Nofriansyah, 2014) :

$$r_{ij} \begin{cases} \frac{x_{ij}}{Max\ x_{ij}} ; Jika\ j\ adalah\ atribut\ keuntungan \\ \frac{Min\ x_{ij}}{x_{ij}} ; Jika\ j\ ada\ lah\ atribut\ benefit \end{cases} \quad (2.9)$$

Keterangan :

$Max\ xij$= Nilai terbesar dari setiap kriteria j.

$Min\ xij$= Nilai terkecil dari setiap kriteria j.

xij= Nilai atribut yang dimiliki dari setiap kriteria.

Benefit = jika nilai terbesar adalah terbaik.

Cost = jika nilai terkecil adalah terbaik.

Nilai preferensi untuk setiap alternatif (Vi) ditunjukan pada Persamaan 2

$$V_i = \sum_{j=1}^{n} w_j r_{ij} \quad (2.10)$$

Keterangan :

Vi= Peringkat untuk setiap alternatif

wj= Nilai bobot (dari setiap kriteria).

rij= Nilai rating kinerja ternormalisasi

Nilai Vi lebih besar mengindikasikan bahwa alternatif Ai lebih terpilih.

Keunggulan dari metode *Simple Additive Weighting* dibandingkan dengan metode sistem keputusan yang lain terletak pada kemampuannya dalam melakukan penilaian secara lebih tepat karena didasarkan pada nilai kriteria dan bobot tingkat kepentingan yang dibutuhkan. Dalam metode *SAW* juga dapat menyeleksi alternatif terbaik dari sejumlah alternatif yang ada kemudian dilakukannya proses perangkingan yang jumlah nilai bobot dari semua kriteria dijumlahkan setelah menentukan nilai bobot dari setiap kriteria. Intinya bahwa pada metode *SAW* ini menentukan nilai bobot pada setiap kriteria untuk menentukan alternatif yang paling optimal.

2.5. UML (*Unified Modeling Language*)

UML *(Unified Modeling Language)* adalah notasi yang lengkap untuk membuat visualisasi model suatu sistem. Sistem berisi informasi dan fungsi, tetapi secara normal digunakan untuk memodelkan sistem komputer. Di dalam pemodelan objek guna menyajikan sistem yang berorientasi objek kepada orang lain, akan sangat sulit di lakukan dalam bentuk kode bahasa pemrograman [10].

UML disebut sebagai bahasa pemodelan bukan metode. Bahasa pemodelan (sebagian besar grafik) merupakan notasi model yang digunakan untuk mendesain secara cepat. Bahasa pemodelan merupakan bagian terpenting dari metode. UML merupakan bahasa standar untuk penulisan *blueprint software* yang digunakan untuk visualisasi, spesifikasi, pembentukan dan pendokumentasian alat-alat dari sistem perangkat lunak. UML biasanya disajikan dalam diagram atau gambar yang meliputi *class* beserta atribut dan operasinya, serta hubungan antar kelas. UML terdiri dari banyak diagram diantaranya *use case diagram, activity diagram*, *class diagram* dan *sequence diagram.*

2.5.1. Use Case Diagram

Dalam konteks UML , tahap konseptualisasi dilakukan dengan pembuatan *use case diagram* yang sesungguhnya merupakan deskripsi peringkat tinggi bagaimana perangkat lunak (aplikasi) akan digunakan oleh penggunanya.

Use case diagram merupakan deskripsi lengkap tentang interaksi yang terjadi antara para actor denga sistem [10]. Dalam hal ini, setiap objek yang berinteraksi dengan sistem merupakan actor untuk sistem, sementara *use case* merupakan deskripsi lengkap tentang bagaimana sistem berperilaku kepada actornya. Actor dalam *use case diagram* digambarkan sebagai ikon yang berbentuk manusia, sementara *use case* digambarkan sebagai elips yang berisi nama *use case*yang bersangkutan. Untuk mempermudah pemahaman, actor biasanya dituliskan sebagai kata benda, sementara *use case* biasanya dituliskan sebagai kata kerja.

2.5.2. Activity Diagram

Activity diagram pada dasarnya menggambarkan skenario secara grafis. Serta *activity diagram* cukup serupa dengan diagram alir (*flow chart),* yang membedakan mungkin hanya *swimlane* yang menunjukkan suatu *state* berada pada objek/kelas tertentu. Keunggulan dari *activity diagram* adalah bahwa diagram tersebut lebih mudah dipahami dibanding skenario. Selain itu, dengan menggunakan *activity diagram,* kita dapat melihat dibagian manakah sistem dari suatu skenario akan berjalan [10].

2.5.3. Class Diagram

Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Class diagram menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti containment , pewarisan, asosiasi, dan lain-lain.

2.5.4. Sequence Diagram

Diagram sekuen menggambarkan kelakuan/perilaku objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Objek-objek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui

objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

2.6. Teknik Evaluasi Peringkasan Teks

Evaluasi peringkasan teks adalah cara untuk mengetahui hasil dari sebuah peringkasan itu baik oleh peringkasan secara manual ataupun memiliki tingkat akurasi yang baik atau tidak untuk peringkasan teks otomatis. Sebuah hasil ringkasan dapat dikatakan sudah tepat oleh sebagian orang, tetapi menurut sebagian orang lagi tidak demikian. Secara umum teradapat dua jenis metode evaluasi ringkasan, yaitu [11]:

1. Ekstrinsik yaitu kualitas sebuah ringkasan yang diukur berdasarkan bagaimana ringaksan tersebut dapat membantu penyelesaian tugas orang yang membaca ringkasan tersebut.
2. Intrinsik yaitu kualitas sebuah ringkasan yang diukur dari kualitas keluaran ringaksan yang dihasilkan.

Evaluasi peringkasan yang digunakan adalah evaluasi intrinsik. Salah satu metode untuk mengukur hasil ringkasannya adalah *ROUGE*. *ROUGE* mengukur kualitas hasil ringaksan dengan menghitung unit – unit yang *overlap* seperti urutan kata dan pasangan – pasangan kata antara ringkasan kandidat dan ringkasan hasil yang menjadi referensi *ROUGE* [12]. Evaluasi yang diukur adalah *overlap* dari isi atau disebut *recall* dan *precission* kalimat. Kombinasi antara nilai *recall* dan *precission* menhasilkan f-measure.

2.6.1. Recall

Recall merupakan istilah yang digunakan untuk kalimat terpanggil yang relevan dengan pernyataan atau vector *query*. Perbandingan jumlah informasi relevan yang didapatkan sistem dengan jumlah seluruh informasi relevan yang ada dalam koleksi informasi. *Recall* berhubungan dengan kemampuan dalam menemukan kalimat yang relevan [13]. Hal ini berarti *recall* adalah bagian dari proses evaluasi yang dapat digunakan sebagai alat ukur relevannya sebuah ringkasan.

$$\boldsymbol{recall}=\frac{hits}{hits+misses} \quad (14)$$

Dimana $hits$ = Jumlah kalimat yang berhasil diekstrak sistem yang sesuai dengan hasil ekstrak manusia
$misses$= Jumlah kalimat yang diekstrak manusia tetapi tidak terdapat dalam hasil ekstrak sistem

2.6.2. Precission

Precission adalah tingkat ketepatan hasil ringkasan. Perbandingan jumlah informasi relevan yang didapatkan sistem dengan jumlah seluruh informasi yang terambil oleh sistem baik yang relevan maupun tidak.

$$Precission=\frac{hits}{hits+noice} \quad (15)$$

Dimana $hits$ = Jumlah kalimat yang berhasil diekstrak sistem yang sesuai dengan hasil ekstrak manusia
$noice$ = Jumlah kalimat yang diekstrak sistem tetapi tidak terdapat dalam kalimat yang diekstrak manusia

2.6.3. f-measure

f-measure adalah hubungan antara *recall* dan *precission* yang memperlihatkan hasil akurasi sistem. Nilai *recall* dan *precission* pada suatu keadaan dapat memiliki bobot yang berbed. Ukuran yang menampilkan timbal balik antara *recall* dan *precission* adalah *f-measure* yang merupakan bobot *harmonic mean* dari *recall* dan *precission* [14].

$$\boldsymbol{f\text{-}measure}=\frac{2*precission*recall}{precission+recall} \quad (16)$$

Dimana *precission* = hasil dari perhitungan *precission*
Recall = hasil dari perhitungan *recall*