

**PERINGKASAN TEKS BERITA BAHASA INDONESIA  
MENGUNAKAN METODE  
*CROSS LATENT SEMANTIC ANALYSIS***

**SKRIPSI**

**Disusun Untuk Memenuhi Persyaratan  
Mencapai Derajat Sarjana Komputer**



**Disusun Oleh :**

**Hermawan  
1600018075**

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INDUSTRI  
UNIVERSITAS AHMAD DAHLAN  
2020**

# **LEMBAR PENGESAHAN**

## **SKRIPSI**

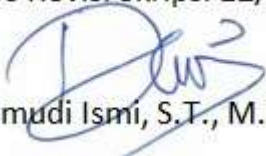
### **PERINGKASAN TEKS BERITA BAHASA INDONESIA MENGUNAKAN METODE CROSS LATENT SEMANTIC ANALYSIS**

Dipersembahkan dan disusun oleh:



**Telah disetujui oleh:**  
**Pembimbing**

Acc Revisi Skripsi 12/1/21

  
Dewi Pramudi Ismi, S.T., M.CompSc

# HALAMAN PENGESAHAN

## SKRIPSI PERINGKASAN TEKS BERITA BAHASA INDONESIA MENGUNAKAN METODE CROSS LATENT SEMANTIC ANALYSIS

yang dipersembahkan dan disusun oleh:

Hermawan  
1600018075

telah dipertahankan di depan Dewan Penguji  
pada tanggal 31 Desember 2020  
dan dinyatakan telah memenuhi syarat

### Susunan Dewan Penguji :

Ketua	: Dewi Pramudi Ismi, S.T, M.CompSc.	.....
Penguji 1	: Dewi Soyusiawati, S.T., M.T.	.....
Penguji 2	: Ahmad Azhari, S.Kom., M.Eng.	.....

Yogyakarta, 31 Desember 2020  
Dekan  
Fakultas Teknologi Industri  
Universitas Ahmad Dahlan



Sunardi, S.T., M.T., Ph.D.  
NIY. 60010313

## PERNYATAAN TIDAK PLAGIAT

Saya yang bertanda tangan di bawah ini:

Nama : Hermawan

NIM : 1600018075

Email :

hermawan1600018075@webmail.uad.ac.id

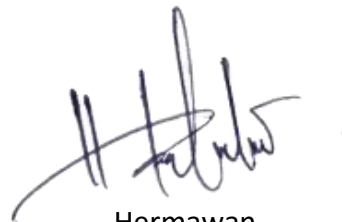
Fakultas : Teknologi Industri Program Studi : Teknik Informatika

Judul TA/Skripsi: Peringkasan Teks Berita Bahasa Indonesia Menggunakan Metode Cross Latent Semantic Analysis

Dengan ini menyatakan bahwa:

1. Hasil karya yang saya serahkan ini adalah asli dan belum pernah diajukan untuk mendapatkan gelar kesarjanaan baik di Universitas Ahmad Dahlan maupun di institusi pendidikan lainnya.
2. Hasil karya saya ini bukan saduran/terjemahan melainkan merupakan gagasan, rumusan, dan hasil pelaksanaan penelitian/implementasi saya sendiri, tanpa bantuan pihak lain, kecuali arahan pembimbing akademik dan narasumber penelitian.
3. Hasil karya saya ini merupakan hasil revisi terakhir setelah diujikan yang telah diketahui dan disetujui oleh pembimbing.
4. Dalam karya saya ini tidak terdapat karya atau pendapat yang telah ditulis atau dipublikasikan orang lain, kecuali yang digunakan sebagai acuan dalam naskah dengan menyebutkan nama pengarang dan dicantumkan dalam daftar pustaka.

Yogyakarta, 31 Desember 2020



Hermawan

## PERNYATAAN PERSETUJUAN AKSES

Saya yang bertanda tangan di bawah ini:

Nama : Hermawan

NIM : 1600018075

Email :

hermawan1600018075@webmail.uad.ac.id

Fakultas : Teknologi Industri Program Studi : Teknik Informatika

Judul TA/Skripsi: Peringkasan Teks Berita Bahasa Indonesia Menggunakan Metode Cross Latent Semantic Analysis

Dengan ini saya menyerahkan hak sepenuhnya kepada Perpustakaan Universitas Ahmad Dahlan untuk menyimpan, mengatur akses serta melakukan pengelolaan terhadap karya saya ini dengan mengacu pada ketentuan akses tugas akhir elektronik sebagai berikut:

☒ Saya mengizinkan karya tersebut diunggah ke dalam aplikasi Repository Perpustakaan Universitas Ahmad Dahlan.

Demikian pernyataan ini saya buat dengan sebenarnya.

Yogyakarta, 31 Desember 2020



Hermawan

Mengetahui,



**Dewi Pramudi Ismi, S.T, M.CompSc.**

Dosen Pembimbing

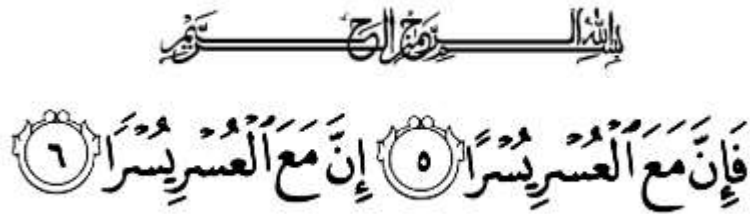
## HALAMAN PERSEMBAHAN



Dengan Rahmat Allah Yang Maha Pengasih lagi Maha Penyayang atas nikmat kekuatan, kemudahan, kelancaran yang telah Engkau berikan. Shalawat serta salam senantiasa saya haturkan kepada junjungan kita, Nabi Muhammad ﷺ atas teladannya. Dengan ini saya persembahkan karya ini untuk :

1. Kedua orang tua saya Bapak Sahirun dan Ibu Nur Baini yang senantiasa memberikan do'a, nasehat, dukungan, semangat, inspirasi dan segalanya. Terimakasih telah mendidik saya dengan sangat baik, memberikan saya kepercayaan, serta pengorbanan yang mengiringi langkah saya dari sejak saya kecil, terimakasih untuk segalanya.
2. Kakak kandungku Rahayu, Amd.Keb, Budiman, S.IP yang selalu memberi do'a, dukungan, semangat serta nasehat kepada saya.
3. Sahabat seperjuangan Arfiansyah, Luthfiantoro, Afandi, Ian, Oei, Alvin, Febrian, Habibila, Akmal, Aprizal, Ardiansyah, Gema, Erizal terimakasih untuk solidaritas yang luar biasa dari selama awal hingga akhir perkuliahan.
4. Arfiansyah dan Mas Gontang Ragil Prakasa terimakasih telah banyak membagikan ilmunya, memberikan banyak solusi dan motivasi dalam menyelesaikan skripsi ini.
5. Sahabat-sahabatku Verdi Apriansyah, Dea Nandari, Mona Riza, Ulfiana Nurfatihah sahabat kecilku hingga sama-sama kita berjuang menempuh pendidikan di Yogyakarta. Terimakasih atas dukungan, candaan, dan kebersamaannya yang memberikan warna tersendiri selama masa perkuliahan.
6. Teknik Informatika Angkatan 2016. Khususnya kelas B yang telah memberikan banyak pengajaran selama kuliah.

## MOTTO



**Artinya:**

**Dan Katakanla “Karena sesungguhnya sesudah kesulitan itu ada kemudahan(5) Sesungguhnya sesudah kesulitan itu ada kemudahan (6)”  
(Q.S At-Insyirah: 5-6)**

**“Tiap kali kita mencari dan mengejar sesuatu yang tinggi dan mulia, pasti kita akan menemui banyak kesulitan dan rintangan. Jangan kita berpikir bahwa mencapai derajat yang tinggi dan mulia adalah pekerjaan yang mudah dan tidak memerlukan usaha keras.”**

Al-Ghazali, Ihya Ulumuddin

**“All’s well that carries on well”**

Amit Abraham

## KATA PENGANTAR

Assalamu'alaikum wr.wb

Dengan mengucap Alhamdulillahirobbil'alamin, segala puji syukur kehadiran Allah Yang Maha Esa atas limpahan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi dengan judul **"PERINGKASAN TEKS BERITA BAHASA INDONEISA MENGGUNAKAN METODE *CROSS LATENT SEMANTIC ANALYSIS*"**. Skripsi ini disusun guna memenuhi persyaratan menyelesaikan derajat Sarjana di Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Ahmad Dahlan Yogyakarta.

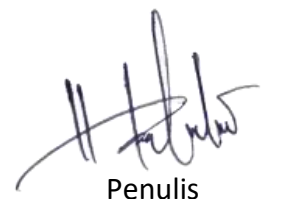
Dalam penyusunan skripsi ini tentunya penulis menyadari bahwa banyak pihak yang telah memberikan bantuannya. Oleh karena itu, penulis mengucapkan terimakasih kepada :

1. Bapak Dr. Muchlas, M.T. selaku Rektor Universitas Ahmad Dahlan.
2. Bapak Sunardi, S.T., M.T., Ph.D. selaku Dekan Fakultas Teknologi Industri Universitas Ahmad Dahlan.
3. Ibu Nur Rochmah Dyah Pujiastuti, S.T., M.Kom. selaku Kepala Program Studi Teknik Informatika.
4. Ibu Ika Arfiani, S.T., M.Cs. selaku dosen pembimbing akademik yang telah memberikan arahan selama masa perkuliahan.
5. Ibu Dewi Pramudi Ismi, S.T., M.CompSc. selaku dosen pembimbing skripsi atas bimbingan dan arahan yang telah diberikan sehingga peneliti dapat menyelesaikan penelitian ini.
6. Ibu Dewi Soyusiawati, S.T., M.T.. selaku dosen penguji I yang telah menyetujui, menerima dan memberikan saran serta kritik pada skripsi ini.
7. Bapak Ahmad Azhari, S.Kom., M.Eng. selaku dosen penguji II yang telah menyetujui, menerima dan memberikan saran serta kritik pada skripsi ini.
8. Segenap dosen Teknik Informatika Universitas Ahmad Dahlan, yang telah membagikan ilmunya sehingga skripsi ini dapat selesai.
9. Semua pihak yang tidak bisa disebutkan satu persatu.

Penulis menyadari bahwa skripsi ini masih jauh dari kata sempurna. Oleh karena itu, kritik serta saran yang membangun selalu penulis harapkan demi penelitian yang lebih baik kedepannya.

Wassalamu'alaikum wr.wb.

Yogyakarta, 31 Desember 2020



Penulis



## DAFTAR ISI

LEMBAR PENGESAHAN .....	ii
HALAMAN PENGESAHAN .....	iii
HALAMAN PERSEMBAHAN .....	vi
MOTTO .....	vii
KATA PENGANTAR.....	viii
DAFTAR ISI.....	ix
DAFTAR GAMBAR.....	xii
DAFTAR TABEL.....	xiii
DAFTAR LISTING .....	xv
ABSTRAK .....	xvi
BAB I PENDAHULUAN.....	1
A. Latar Belakang Masalah.....	1
B. Identifikasi Masalah.....	5
C. Batasan masalah .....	5
D. Rumusan Masalah.....	5
E. Tujuan Penelitian .....	6
F. Manfaat Peneltian .....	6
BAB II TINJAUAN PUSTAKA.....	7
A. Penelitian Terdahulu.....	7
B. Landasan Teori.....	19
1. <i>Text mining</i> .....	19
2. <i>Natural Language Processing (NLP)</i> .....	20
a. Komponen Utama Bahasa Alami.....	21
b. Aplikasi pengolahan bahasa alami .....	21
3. <i>Text Summarization</i> .....	22
a. Pendekatan dalam peringkasan teks .....	22
b. Fitur peringkasan teks.....	24
c. Online tools peringkasan teks.....	26
4. <i>Text preprocessing</i> .....	26
a. Parsing.....	27
b. Case folding.....	27

c. Tokenizing .....	28
d. Filtering .....	28
e. Stemming .....	29
5. <i>Term Frequency-Inverse Document Frequency (TF-IDF)</i> .....	30
6. Singular Value Decomposition .....	31
7. <i>Latent Semantic Analysis (LSA)</i> .....	32
8. <i>Cross Latent Semantic Analysis (CLSA)</i> .....	33
9. <i>Recall-Oriented Understudy for Gisting Evaluation (ROUGE)</i> .....	35
C. Studi kasus .....	37
1. Pengumpulan Data .....	37
2. <i>Preprocessing Text</i> .....	38
a. Parsing .....	38
b. Case folding .....	39
c. Tokenizing .....	39
d. Filtering .....	40
e. Stemming .....	41
3. Pembobotan TF-IDF .....	42
4. Singular Value Decomposition (SVD) .....	46
5. Latent Semantic Analysis (LSA) .....	51
6. <i>Cross Latent Semantic Analysis (CLSA)</i> .....	51
BAB III METODE PENELITIAN .....	54
A. Objek Penelitian .....	54
B. Metode Pengumpulan Data .....	54
1. Studi Pustaka .....	54
2. Teknik Dokumentasi .....	54
C. Spesifikasi Kebutuhan .....	55
1. Perangkat Keras ( <i>Hardware</i> ) .....	55
2. Perangkat Lunak ( <i>Software</i> ) .....	55
D. Tahapan Penelitian .....	56
1. Input Data .....	58
2. <i>Preprocessing</i> .....	58
a. Parsing .....	60
b. Case folding .....	60

c. Tokenizing .....	60
d. Filtering .....	60
e. Stemming .....	61
3. <i>Processing</i> .....	61
a. Pembobotan Tf-Idf .....	62
b. Pembuatan matriks term kalimat .....	62
c. Perhitungan matriks SVD .....	62
4. Kalimat ringkasan.....	62
5. Pengujian .....	63
BAB IV HASIL DAN PEMBAHASAN .....	64
A. Analisis Kebutuhan Sistem.....	64
1. Analisis Pengguna .....	64
2. Analisis Kebutuhan Sistem.....	64
B. Implementasi Sistem .....	65
1. <i>Input Data</i> .....	65
2. <i>Preprocessing</i> .....	66
3. <i>Processing</i> .....	68
a. Perhitungan TF-IDF.....	68
b. Pembentukan Matriks term kalimat .....	69
c. Perhitungan Singular Value Decomposition (SVD).....	71
d. Latent Semantic Analysis (LSA) .....	72
e. Cross Latent Semantic Analysis (CLSA).....	73
C. Tampilan muka ( <i>User interface</i> ) .....	75
1. Tampilan Index.....	75
2. Tampilan Result .....	76
3. Tampilan <i>preprocessing</i> .....	78
4. Tampilan TF-IDF .....	79
5. Tampilan LSA VS CLSA.....	80
D. Pengujian Sistem.....	82
BAB V KESIMPULAN DAN SARAN .....	88
A. Kesimpulan .....	88
B. Saran .....	88
DAFTAR PUSTAKA.....	90

## DAFTAR GAMBAR

Gambar 2. 1 Jenis dan pengaplikasian peringkasan dokumen teks.....	23
Gambar 2. 2 Proses matriks SVD.....	31
Gambar 3. 1 Tahapan Penelitian.....	57
Gambar 3. 2 Flowchart preprocessing .....	59
Gambar 3.3. Alur Processing.....	61
Gambar 4.1. Tampilan Index.....	76
Gambar 4.2. Tampilan Result.....	77
Gambar 4.3. Tampilan tab menu preprocessing .....	79
Gambar. 4.4. Tampilan tab menu TF-IDF .....	80
Gambar. 4.5. Tampilan tab menu LSA VS CLSA.....	81

## DAFTAR TABEL

Tabel 2. 1 Jurnal Penelitian Terdahulu.....	14
Tabel 2. 2. Aturan Parsing.....	27
Tabel 2. 3. Aturan case folding.....	28
Tabel 2. 4. Aturan tokenizing.....	28
Tabel 2. 5. Aturan filtering.....	29
Tabel 2. 6. Aturan stemming.....	29
Tabel 2. 7. Collect Data .....	37
Tabel 2. 8. Tahapan Parsing .....	38
Tabel 2.9. Tahapan Case folding .....	39
Tabel 2.10. Tahapan Tokenizing.....	40
Tabel 2.11. Tahapan Filtering.....	40
Tabel 2.12. Tahapan Stemming .....	41
Tabel 2.14. Matriks A .....	46
Tabel 2.15. Matriks $A^T$ .....	47
Tabel 2.16. Hasil Perkalian matriks $A^T A$ .....	47
Tabel 2.17. Matriks S (nilai eigenvalue) .....	47
Tabel 2.18. Matriks $S^{-1}$ (inverse matriks S).....	48
Tabel 2.19. Perhitungan eigenvector .....	48
Tabel 2.20. Matriks $V^T$ .....	49
Tabel 2.21. Matriks orthogonal U .....	49
Tabel 2.22. hasil perhitungan LSA.....	51
Tabel 2.23. Matriks $V^T$ CISA .....	52
Tabel 2.24. Hasil seleksi matriks $V^T$ CISA .....	52
Tabel 2.25. Hasil ringkasan CLSA.....	53
Tabel 4.1. Hasil Preprocessing .....	68
Tabel 4.2. Hasil Pembobotan TF-IDF .....	69
Tabel 4.3. Hasil Perhitungan Matriks A.....	70
Tabel 4.4 Matriks SVD .....	71
Tabel 4.5. Skor LSA .....	73
Tabel 4.6. Ringkasan LSA.....	73

Tabel 4.7. Matriks $V^T$ CLSA .....	74
Tabel 4.8. Skor CLSA.....	74
Tabel 4.9. Ringkasan CLSA.....	74
Listing 4.11. Tab menu LSA VS CLSA .....	81
Tabel 4.10. Sampel Pengujian .....	82
Tabel 4.11. Perbandingan ringkasan.....	83
Tabel 4.12. Hasil Pengujian .....	85
Tabel 4.13. Penjabaran Hasil Pengujian.....	86

## DAFTAR LISTING

Listing 4.1. Preprocessing Teks .....	67
Listing 4.2. Perhitungan TF-IDF .....	69
Listing 4.3. Perhitungan Matriks A.....	70
Listing 4.4. Perhitungan SVD .....	71
Listing 4.5. Skor LSA .....	72
Listing 4.6. Seleksi Matriks $V^T$ .....	74
Listing 4.7. Controller indexs.....	76
Listing 4.8. Controller Result .....	78
Listing 4.9. Tab menu preprocessing .....	79
Listing 4.10. Tab menu TF-IDF.....	80
Listing 4.11. Tab menu LSA VS CLSA .....	81

## PERINGKASAN TEKS BERITA BAHASA INDONESIA MENGGUNAKAN METODE *CROSS LATENT SEMANTIC ANALYSIS*

Hermawan

Teknik Informatika, Fakultas Teknologi Industri, Universitas Ahmad Dahlan  
1600018075

### ABSTRAK

Informasi yang tersedia saat ini berkembang pesat seiring dengan pertumbuhan informasi digital. Sehingga banyak bermunculan situs-situs berita terutama pada media *online*. Teks berita yang tersedia umumnya berbentuk teks bacaan yang panjang dan sulit untuk memahami inti dari berita tersebut. Untuk memahami inti dari teks berita tersebut membutuhkan waktu yang relatif lama. Oleh sebab itu, dibutuhkan suatu sistem yang mampu meringkas dokumen untuk mendapatkan inti dari dokumen asli, sehingga memudahkan pengguna dalam menemukan inti dari informasi yang dicari.

Metode atau pendekatan yang dilakukan untuk membuat peringkasan teks otomatis terus mengalami perkembangan untuk menghasilkan ringkasan yang lebih baik. Pada penelitian ini akan dilakukan peringkasan teks dengan penerapan metode *Cross Latent Semantic Analysis* (CLSA) yang merupakan pengembangan dari metode *Latent Semantic Analysis* (LSA). Metode CLSA menerapkan tahapan perhitungan Preprocessing (*Parsing*, *Case Folding*, *Tokenizing*, *Filtering* dan *Stemming*), perhitungan pembobotan kata Tf-idf, perhitungan matriks Singular Value Decomposition (SVD), dan peringkasan berita dari hasil skor tertinggi dari dokumen kalimat hasil perhitungan SVD. Tahapan pengujian sistem menggunakan ROUGE test.

Pengujian sistem ini menggunakan perbandingan antara hasil ringkasan sistem dengan hasil ringkasan manual dengan menggunakan *library* ROUGE. Pengujian diukur dengan nilai *recall*, *precision* dan *f-measure*, dengan menggunakan persentase nilai *compression rate* yaitu 50% dengan data uji yaitu berita dari website [www.kompas.com](http://www.kompas.com), [www.detik.com](http://www.detik.com), [www.krjogja.com](http://www.krjogja.com). Hasil pengujian yang dilakukan dengan menggunakan 10 dokumen berita, dimana setiap dokumen berita diringkaskan 5 orang berbeda. Dari keseluruhan berita pengujian memperoleh hasil rata-rata *f-measure* 0.6533, *precision* 0.6727, *recall* 0.6572.

**Kata Kunci :** Berita, CLSA, Peringkasan, Teks, *Rouge*.



## BAB I

### PENDAHULUAN

#### A. Latar Belakang Masalah

Informasi yang tersedia saat ini berkembang pesat seiring dengan pertumbuhan informasi digital. Berita *online* merupakan salah satu media yang menyajikan informasi terkini dengan berbagai topik berita. Munculnya situs-situs *online* membuat media informasi mengalami perkembangan yang signifikan. Berdasarkan hasil survei 2018 dari Asosiasi Penyelenggara Jasa Internet Indonesia (APJI), 5.5% dari 171,17 juta pengguna internet Indonesia membaca berita di media online sebagai gaya hidup (APJI, 2018). Hal ini bisa disimpulkan bahwa berita merupakan suatu kebutuhan dan pengetahuan karena berita merupakan salah satu sajian yang diminati pengguna internet di Indonesia.

Kebutuhan untuk memperoleh informasi berita secara praktis menjadi masalah yang harus diselesaikan seiring dengan berkembangnya *volume* berita yang dapat diakses secara *online*. Semakin banyak informasi yang tersedia dalam suatu dokumen berita maka semakin panjang pula sebuah dokumen tersebut. Untuk mendapatkan inti informasi dari suatu dokumen dapat dilakukan dengan membaca isi dokumen secara keseluruhan. Hal ini membutuhkan waktu yang cukup lama jika dibandingkan dengan membaca isi ringkasan dari suatu teks. Oleh sebab itu, dibutuhkan suatu sistem yang mampu

meringkas dokumen untuk mendapatkan inti dari dokumen asli, sehingga memudahkan pengguna dalam menemukan inti dari informasi yang dicari.

Ringkasan merupakan bentuk dokumen yang lebih singkat dari sebuah teks dokumen yang dibuat dari satu dokumen atau lebih, dan terdapat informasi-informasi penting yang terdapat pada dokumen asli (Saputra, Jerry. Fachrurrozi, 2017). Peringkasan teks otomatis atau yang dikenal sebagai *Automatic Text Summarization* (ATS) merupakan bagian dari NLP (*natural language processing*) yang bisa digunakan untuk membuat sebuah sistem aplikasi yang mampu membuat ringkasan teks dokumen otomatis. Peringkasan teks otomatis dilakukan dengan menghitung nilai tiap dokumen dengan melakukan perengkingan pada setiap dokumen. Dokumen yang memuat informasi penting didalamnya akan ditempatkan pada rengking teratas yang akan dijadikan kalimat ringkasan. *Automatic Text Summarization* merupakan suatu proses pengurangan panjang dari teks asli dengan tetap mempertahankan kualitas dari hasil ringkasan tersebut tanpa menghilangkan inti dari dokumen yang diringkaskan. Solusi ini bisa digunakan dalam hal peringkasan teks dokumen.

Algoritma *latent semantic analysis* (LSA) merupakan salah satu algoritma yang digunakan dalam peringkasan teks otomatis untuk memperoleh informasi penting dari suatu dokumen. Penelitian yang dilakukan oleh (Winata, Rainarli, Informatika, & Indonesia, 2016) berupaya mengembangkan metode LSA dengan melakukan perubahan pada tahap ekstraksi kalimat. Penelitian ini

menghasilkan akurasi ringkasan sebesar 69,6%. Namun pada tahap proses *preprocessing* daftar *stop words* dan penggunaan kata dasar bahasa Indonesia yang digunakan tidak lengkap sehingga tingkat akurasinya masih kecil. Penelitian yang dilakukan oleh (Saputa, Jerry. Fachrurrozi, 2017) juga melakukan pengembangan terhadap metode LSA dengan menambahkan teknik *Steinberger & Jezek* pada tahapan pemilihan kalimat. Teknik *Steinberger & Jezek* yang digunakan untuk pemilihan kalimat ringkasan. Teknik tersebut dapat memperoleh panjang kalimat dari suatu dokumen, sehingga mampu meningkatkan nilai *f-measure* dan nilai akurasi ringkasan. Namun penelitian yang dilakukan Winata dan Rainarli tersebut masih memiliki kekurangan. Metode LSA yang digunakan pada penelitian yang dilakukan Winata dan Rainarli hanya mempertimbangkan bobot kata dari setiap kalimat menjadikan hasil ringkasannya masih bisa ditingkatkan. Dengan memperhitungkan relasi antar kata pada setiap kalimat kalimat dengan tidak hanya menghitung bobot kata.

Metode peringkasan teks otomatis terus mengalami perkembangan untuk menghasilkan ringkasan yang lebih baik. Algoritma *Cross Latent Semantic Analysis* (CLSA) merupakan salah satu algoritma yang merupakan pengembangan dari algoritma LSA untuk peringkasan teks dokumen. CLSA merubah perhitungan pada LSA dengan melakukan proses silang untuk mempercepat proses pada LSA (Mandar & Gunawan, 2017). Pada metode LSA pemanfaatan *Singular Value Decomposition* (SVD) untuk membuat serta

mengolah matriks hasil proses pembobotan kata pada dokumen untuk menemukan hubungan kesamaan antara kata dan kalimat. Penelitian yang dilakukan oleh (Steinberger, 2004) membuat ringkasan dengan memperhitungkan panjang kalimat, tidak hanya berdasarkan pada kemiripan kata dan kalimat dalam suatu dokumen. Penelitian *Steinberger* menjadi cikal bakal munculnya CLSA. Dalam penelitiannya (Mandar & Gunawan, 2017) mengungkapkan bahwa metode CLSA membuat hasil ringkasan lebih pendek dari LSA. CLSA mampu meningkatkan nilai akurasi dari metode LSA walaupun dengan hasil akurasi antara LSA dan CLSA tidak terlalu jauh berbeda. Penelitian yang dilakukan Mandar dan Gunawan masih bisa ditingkatkan dengan menambahkan *similarity* antara judul dan dokumen berita karena judul berita merupakan bagian penting dari suatu berita.

Dari beberapa hasil uraian penelitian yang telah dilakukan sebelumnya menunjukkan bahwa metode CLSA dinilai cukup baik untuk melakukan peringkasan teks otomatis dibandingkan dengan metode LSA. Namun penelitian tentang peringkasan teks otomatis dengan metode CLSA masih minim dilakukan. Dalam penelitian ini akan memperbaiki kelemahan-kelemahan pada penelitian yang telah dilakukan. Pada penelitian ini akan melakukan peringkasan teks berita bahasa Indonesia menggunakan penerapan metode *Cross Latent Semantic Analysis* (CLSA) dengan tahapan proses *preprocessing* yang lebih kompleks. Sehingga diharapkan akan menghasilkan hasil ringkasan

yang berisi intisari dari keseluruhan dokumen, mendekati ringkasan yang dihasilkan oleh manusia dengan akurat.

## **B. Identifikasi Masalah**

Berdasarkan uraian dari latar belakang masalah, maka didapatkan identifikasi masalah sebagai berikut :

1. Proses memahami intisari suatu sumber bacaan secara keseluruhan memerlukan waktu lebih lama dibandingkan dengan memahami ringkasan dari sumber bacaan.
2. Peringkasan yang dilakukan secara manual sulit dilakukan apabila teks berita yang diringkas memiliki banyak kata atau teks berita yang cukup panjang.

## **C. Batasan masalah**

Batasan masalah yang diangkat didapat berdasarkan identifikasi masalah yang dibatasi pada permasalahan:

1. Masukan yang digunakan berupa dokumen teks berita bahasa Indonesia.
2. *Dataset* dokumen yang digunakan berjumlah 300 dokumen berita berbahasa Indonesia.
3. *Dataset* yang digunakan pada pengujian sistem menggunakan *dataset* berita dari halaman [www.kompas.com](http://www.kompas.com), [www.detik.com](http://www.detik.com), [www.krjogja.com](http://www.krjogja.com).

## **D. Rumusan Masalah**

Berdasarkan uraian dari latar belakang yang telah dipaparkan diatas, permasalahan yang dapat dirumuskan adalah sebagai berikut :

1. Bagaimana membuat sistem aplikasi ATS (*Automated Text Summarization*) pada berita Bahasa Indoneisa untuk mempermudah dalam pencarian inti dari berita.
2. Bagaimana performansi dari sistem peringkasan teks Bahasa Indonesia menggunakan metode CLSA.

#### **E. Tujuan Penelitian**

Melalui penelitian ini, maka tujuan yang ingin dicapai oleh penulis adalah:

1. Membuat sistem aplikasi ATS (*Automated Text Summarization*) berita Bahasa Indoneisa untuk mempermudah dalam pencarian inti dari berita.
2. Mengetahui performansi dari sistem peringkasan teks Bahasa Indonesia menggunakan metode CLSA.

#### **F. Manfaat Peneltian**

Dengan menerapkan metode CLSA dalam peringkasan teks berita Bahasa Indonesia diharapkan dapat menghasilkan ringkasan yang berisi intisari dari keseluruhan dokumen sehingga pengguna sistem dapat membuat keputusan untuk melanjutkan membaca keseluruhan dokumen atau cukup dengan membaca ringkasan.

## BAB II

### TINJAUAN PUSTAKA

#### A. Penelitian Terdahulu

Penelitian ini mengangkat beberapa kajian penelitian terdahulu sebagai referensi dalam memperkaya bahan kajian pada penelitian yang dilakukan.

Merujuk pada penelitian sebelumnya yang terkait dengan peringkasan teks Bahasa Indonesia. Pertama adalah penelitian yang dilakukan oleh Yuliska dan Khairul Umam Syalima, dengan penelitian yang berjudul “Literatur Review Terhadap Metode, Aplikasi dan *Dataset* Peringkasan Dokumen Teks Otomatis untuk Teks Berbahasa Indonesia”. Penelitian ini memberikan pemaparan tentang metode, aplikasi, dataset dan teknik yang dapat diimplementasikan untuk *riset* di bidang peringkasan dokumen untuk teks berbahasa Indonesia. Penelitian ini juga menjelaskan berbagai teknik *text summarization*, baik *unsupervised* maupun *supervised*, *dataset* yang dapat digunakan sebagai baseline dalam pengembangan sebuah metode dan *evaluation measure* yang tepat. Menjelaskan sejauh apa perkembangan riset di bidang *text summarization* untuk dokumen berbahasa Indonesia.

Hasil penelitian ini menjelaskan bahwa peringkasan dokumen teks secara otomatis didominasi oleh teknik yang bersifat *ekstraktif*. Peringkasan dokumen teks berbahasa Indonesia juga didominasi oleh metode-metode *unsupervised*, sementara metode-metode *supervised* seperti *machine learning* dan *deep learning* masih sangat jarang ditemukan. penelitian juga dilakukan terhadap

kesalahan yang umum terjadi dalam mengevaluasi hasil ringkasan dan memberikan parameter evaluasi yang lebih tepat untuk diimplementasikan, yaitu ROUGE. Namun tidak semua metode dijelaskan melalui penelitian ini, penelitian ini memberikan pemahaman yang bagi perkembangan text *summarization* berbahasa Indonesia dan peluang risetnya.

Peringkasan dokumen yang dilakukan oleh Santun Irawan, Hermawan, dan Samsuryadi menggunakan metode LSA dengan menambahkan metode MMR (*Maximum Marginal Relevance*). Penerapan metode ini untuk mengukur relevansi antara ringkasan yang dihasilkan sistem dengan ringkasan perbandingan yang dibuat oleh ahli bahasa Indonesia. Penerapan LSA dengan membuat data *training* menjadi bentuk matriks kata, matriks kata tersebut kemudian direduksi oleh SVD. Penilaian pada metode LSA ini adalah dengan perhitungan *cosine similarity* antara vector topik dengan vektor kalimat dengan cara membandingkan vektor topik dengan vektor kalimat. Nilai dengan cosine similarity tertinggi akan dijadikan sebagai nilai akhir. Penilaian pada metode LSA ini menghasilkan *score* pada setiap kalimat yang akan digabungkan pada saat menghitung nilai *score* awal pada tahap MMR. MMR digunakan dengan kelakuan kombinasi matriks *cosine similarity* untuk merangking kalimat-kalimat sebagai tanggapan pada *query* yang masukkan oleh pengguna.

Dokumen berita yang digunakan untuk diuji terdiri dari dokumen berita ekonomi, hukum, internasional, olahraga, dan teknologi. Ringkasan yang dihasilkan berdasarkan tingkatan kompresi yang telah ditetapkan yaitu 10% dan



20%. Hasil pengujian ringkasan kategori berita ekonomi dengan jumlah kalimat awal 96 kalimat menjadi 9 kalimat dengan tingkat komperensi 10% dan 19 kalimat dengan tangka komperensi 20%. Hasil penelitian ini berupa kerangka kerja untuk peringkasan multi-dokumen. Kerangka kerja ini dapat menjadi faktor yang sangat menentukan terhadap hasil ringkasan referensi yang dibuat oleh ahli bahasa Indonesia (Irawan, 2017).

Penelitian yang dilakukan oleh Fernando winata dan Ednawati Rainarli dengan melakukan implementasi *Cross Method Latent Semantic Analysis* (CMLSA) untuk meringkas dokumen berita berbahasa Indonesia. Dengan melakukan pengembangan terhadap medote LSA dengan menambahkan perbaikan pada tahap ekstraksi kalimat ringkasan yang dilakukan untuk meningkatkan tingkat akurasi ringkasan yang dihasilkan. Pada tahapan *preprocessing* penelti menambahkan tahapan lain untuk kasus tertentu seperti *case folding* dan penghilangan kata yang jarang dimunculkan atau kata dengan frekuensi kemunculan yang kecil. Tahapan utama peringkasan ini yaitu pembuatan matriks dengan tahapan pembobotan menggunakan algoritma *Term Frequency – Inverse Document Frequency*. Hasil pembobotan dijadikan matriks lalu diolah dengan pemanfaatan SVD untuk mengubah matriks tersebut menjadi lebih kecil dan ekstraksi kalimat ringkasan dengan pemilihan kalimat dengan bobot kata terbesar yang akan dijadikan sebagai ringkasan.

Penerapan metode *Cross Method Latent Semantic Analysis* (CMLSA) terdapat pada saat tahap ekstraksi ringkasan dengan baris-baris pada matriks

atau kalimat-kalimat yang mempunyai nilai *length* yang tinggi akan dijadikan sebagai ringkasan. Data yang digunakan dalam penelitian yaitu artikel berita bertema politik yang bersumber dari portal berita [viva.co.id](http://viva.co.id) dengan jumlah artikel berita yang digunakan berjumlah enam buah dokumen. Dari hasil penelitian ini dapat dihasilkan nilai *precision* 72,25%, *recall* sebesar 66,7% dan *f-measure* sebesar 69,6%. Hasil ringkasan dengan metode CMLSA ini memiliki nilai akurasi ringkasan sebesar 69,6% (Winata et al., 2016).

Penelitian dengan topik peringkasan teks juga dilakukan oleh Jerry Satiamy Saputra, Muhammad Fachrurrozi dan Yunita dengan menggunakan metode LSA dengan menambahkan teknik Steinberger & Jezek. Metode LSA yang digunakan menambahkan teknik *Steinberger & Jezek* untuk pemilihan kalimat. Tahapan penelitian terdiri dari *preprocessing* dan *processing*, tahap-tahap yang dilakukan pada proses *preprocessing* adalah *casefolding*, *sentence segmentation*, *tokenization*, *stopword removal*, dan *stemming*. Tahapan *processing* yaitu proses utama terdiri dari implementasi LSA, pembobotan *tf-idf*, perhitungan SVD dan pemilihan kalimat dengan teknik *Steinberger&Jezek*.

Pada tahapan *preprocessing stemming* kata menggunakan *InaNLP Library* yang berfungsi sebagai pemisah kalimat, *tokenizing*, normalisasi kata, *stopword*, dan *stemmer*. *InaNLP Library* juga berfungsi sebagai sumber kamus kata dasar bahasa Indonesia dan kamus *stopword list*. Penerapan *Singular Value Decomposition* pada matriks pembobotan menggunakan *Java Matrix* (JAMA) untuk mempermudah perhitungan matriks pada penelitian yang akan

dilakukan. Dengan menggunakan teknik *Steinberger & Jezek* untuk pemilihan kalimat ringkasan sehingga dapat mengetahui panjang dari dokumen kalimat dengan menghapus kalimat-kalimat yang tidak penting atau kalimat yang tidak memiliki hubungan. Panjang dokumen kalimat kalimat yang dihasilkan mampu meningkatkan nilai dari akurasi *f-measure*. Memperoleh hasil ringkasan sebesar 50% dari teks dokumen asli dengan nilai *f-measure* tertinggi sebesar 0,71 (Saputra, Jerry. Fachrurrozi, 2017).

Penelitian yang dilakukan oleh Nurina savant widya gotami, Indrianti, Ratih Kartika dewi melakukan peringkasan ekstraktif terhadap artikel berita kesehatan berbahasa Indonesia dengan menggunakan metode LSA. Peringkasan teks dokumen dikategorikan menjadi beberapa golongan yaitu peringkasan teks *ekstraktif* dan *abstraktif*. Peringkasan ekstraktif berupaya mengekstrak hal-hal penting kalimat pada suatu dokumen asli kemudian ditampilkan kembali dengan bentuk yang lebih sederhana. Sebaliknya peringkasan abstraktif menghasilkan interpretasi dari teks asli yang ada dalam bentuk abstrak. Metode LSA merupakan salah satu metode peringkasan teks ekstraktif, penerapan metode ini untuk mengekstrak struktur semantik kalimat atau makna yang tersembunyi pada sebuah kalimat.

Metode LSA sendiri menggunakan teori *aljabar linear singular value decomopositon* (SVD) yang digunakan sebagai fitur untuk penghilang pengulangan kata pada kata tertentu, SVD membentuk matriks representasi dari asosiasi *term* kata dari hasil pembobotan *tf-idf*. Pembobotan kata pada

setiap kata pada kalimat didapatkan dari hasil perhitungan frekuensi kemunculan kata pada suatu dokumen atau disebut dengan *term frequency* (TF), kemudian dilakukan perhitungan frekuensi kemunculan dokumen yang mengandung kata atau disebut juga dengan *document frequency* (DF), kemudian melakukan perhitungan jumlah dokumen yang mengandung kata tersebut dengan melakukan perhitungan *inverse document frequency* (IDF). Penelitian ini menggunakan metode *cross method* LSA untuk memilih atau menyusun ringkasan dalam tiap kalimat hasil ekstraksi dari metode LSA sebelumnya.

Penelitian yang dilakukan oleh Gamaria Mandar dan Gunawan dengan penelitian (Mandar & Gunawan, 2017) melakukan peringkasan dokumen berita berbahasa Indonesia dengan menggunakan metode *cross latent semantic analysis* (CLSA). Penerapan metode CLSA ini untuk menguji hasil ringkasan yang diperoleh dari metode CLSA dari perbandingannya dengan hasil ringkasan dengan metode LSA dengan sama-sama menggunakan 240 artikel berita Bahasa Indonesia dari *website* kompas.com dan dua pakar sebagai perbandingan uji ke-dua metode ini. Peringkasan teks ini terbagi menjadi tiga tahapan proses yaitu tahapan *preprocessing*, pembobotan *tf-idf* dan peringkasan CLSA dan LSA. Pada tahapan *preprocessing* penelitian ini menambahkan proses *case folding* yaitu tahapan menyamakan teks dengan cara mengubah semua huruf masukan menjadi huruf kecil atau hanya menerima huruf 'a-z' dan menghilangkan beberapa karakter yang digunakan

untuk pembatas atau pemisah kata misalnya tanda koma, titik koma atau titik dua.

Pada metode CLSA perhitungan SVD mengalami perubahan dari metode LSA. Proses SVD pada peringkasan yang tidak hanya dilihat dari kemiripan antara dokumen kalimat dengan judul berita, panjang dari sebuah dokumen kalimat juga menjadi faktor yang penting dalam menentukan hasil peringkasan. Sedangkan pada LSA penggunaan SVD hanya bertugas untuk mengolah komponen matriks kata dan kalimat untuk menemukan hubungan *similarity* antara kata dan kalimat (Mandar & Gunawan, 2017). Penelitian ini menghasilkan nilai akurasi ringkasan CLSA yang lebih baik dari LSA, dengan tingkat *f-measure* 72% dari LSA dengan tingkat *f-measure* 70% walaupun perbedaannya tidak jauh berbeda. Penelitian ini menghasilkan ringkasan CLSA yang lebih pendek dari hasil ringkasan LSA.

Tabel 2. 1 Jurnal Penelitian Terdahulu

ASPEK	PUSTAKA 1	PUSTAKA 2	PUSTAKA 3	PUSTAKA 4	PUSTAKA 5	PENELITIAN YANG DILAKUKAN
NAMA PENULIS	1. Santun Irawan 2. Hermawan 3. Samsuryadi	1. Fernando Winata 2. Ednawati Rainarli	1. Jerry Satiamy Saputra 2. Muhammad Fachrurrozi 3. Yunita	1. Nurina Savanti Widya Gotami 2. Indriati 3. Ratih Kartika Dewi	1. Gamaria Mandar 2. Gunawan	Hermawan
JUDUL JURNAL	Studi Awal Peringkasan Dokumen Bahasa Indonesia Menggunakan Metode <i>Latent Semantik Analysis</i> dan <i>Maximum Marginal Relevance</i>	Implemetasi Cross Method Latent Semantic Analysis Untuk Meringkas Dokumen Berita Berbahasa Indonesia	Peringkasan Teks Berita Berbahasa Indonesia Menggunakan Metode <i>Latent Semantic Analysis</i> (LSA) dan Teknik Steinberger & Jezek	Peringkasan Teks Otomatis Secara Ekstraktif Pada Artikel Berita Kesehatan Berbahasa Indonesia Dengan Menggunakan Metode <i>Latent Semantic Analysis</i>	Peringkasan dokumen berita Bahasa Indonesia menggunakan metode <i>Cross Latent Semantic Analysis</i>	Peringkasan Teks Berita Bahasa Indonesia Menggunakan Metode <i>Cross Latent Semantic Analysis</i>
VOLUME, NOMOR	Vol 2, No. 1	Vol. 15, No. 4	Vol. 3, No. 1	Vol. 2, No. 9	Vol 3, No 2	-
BULAN, TAHUN TERBIT	6 Desember 2016	November 2016	2017	September 2018	2 Juni 2018	-

PROSIDING	<i>Annual Research Seminar (ARS) - UNSRI</i>	LPPM Universitas Dian Nuswantoro Semarang	<i>Annual Research Seminar (ARS) - UNSRI</i>	JPTIHK - Universitas Brawijaya	<i>Department of Information Systems, Faculty of Science and Technology, Universitas Pesantren Tinggi Darul Ulum (Unipdu) Jombang, East Java, Indonesia</i>	-
MASALAH PENELITIAN	Permasalahan dalam penelitian ini yaitu kurangnya suatu mekanisme yang mampu menyajikan informasi secara efektif.	Metode LSA yang dianggap masih bisa dikembangkan dengan memanfaatkan metode CMLSA untuk memenuhi kebutuhan akan peringkasan teks otomatis yang dapat menghasilkan ringkasan dengan cepat dan akurat. Serta dapat menghasilkan ringkasan dengan tingkat akurasi tinggi.	Pada dokumen teks berita terdapat berita dengan dokumen berita yang Panjang dan sulit untuk mendapatkan inti dari informasi yang dicari jika dilakukan dengan membaca isi dokumen secara keseluruhan. Membutuhkan waktu yang cukup lama untuk mendapatkan inti dari informasi yang dicari. Untuk itu dibutuhkan dokumen yang lebih ringkas dari dokumen asli, sehingga	Informasi yang semakin banyak tersedia membuat banyaknya data teks dokumen salah satunya adalah artikel berita. Proses pencarian informasi peting dari artikel berita selalu mengalami kesulitan karna menggunakan pencarian manuals. Sehingga dibutuhkan <i>Ekstraksi</i> informasi pada suatu dokumen agar dapat mempermudah pencarian informasi	Kemudahan dalam pengaksesan berita yang bisa diperoleh dari berbagai situs <i>online</i> membuat peningkatan dalam pencarian berita, namun berita yang didapatkan sulit untuk menemukan informasi karena teks dokumen yang panjang sehingga membutuhkan waktu yang banyak untuk memahami informasi yang dibaca.	Semakin banyak informasi yang tersedia dalam suatu dokumen berita maka semakin panjang pula sebuah dokumen tersebut. Untuk mendapatkan inti informasi dari suatu dokumen dapat dilakukan dengan membaca isi dokumen secara keseluruhan. Hal ini membutuhkan waktu yang cukup lama jika

			memudahkan dalam hal menemukan informasi yang akan dicari.	dari sebuah artikel berita dengan waktu yang relative lebih cepat.		dibandingkan dengan membaca isi ringkasan dari suatu teks.
TUJUAN PENELITIAN	Membuat suatu sistem yang mampu menghasilkan ringkasan yang baik dan memuat isinya yang mewakili seluruh informasi yang ada pada dokumen asli. Ringkasan yang dibuat lebih singkat dari pada dokumen aslinya sehingga mempermudah pembaca dalam memahami isi dan garis besar dokumen tanpa harus membaca seluruh isi dokumen.	Menciptakan sistem yang mampu membuat ringkasan dengan cepat dan tingkat akurasi yang tinggi dengan melakukan pengembangan pada metode LSA dengan menggunakan CMLSA sebagai metode peringkasan untuk menghasilkan ringkasan yang mendekati hasil ringkasan manusia.	Menciptakan sistem yang mampu membuat ringkasan dari dokumen yang panjang menjadi ringkasan yang memuat intisari dari dokumen asli. Serta mengetahui tingkat akurasi pada peringkasan dokumen berita berbahasa Indoneisa menggunakan metode LSA dengan teknik Steinberger & Jezek pada tahapan pemilihan kalimat ringkasan.	Membantu pembaca atau pengguna untuk mempermudah dalam proses <i>ekstraksi</i> informasi yang ada pada dokumen berita dengan waktu yang cepat. Menciptakan sistem yang mampu membuat ringkasan dari dokumen panjang atau banyak menjadi bentuk singkat tanpa menghilangkan makna penting dari isi dokumen asli.	Menciptakan suatu sistem yang mampu memberikan ide pokok atau informasi penting dari dokumen asli, berupa sebuah teks yang lebih ringkas dengan hasil yang cepat. Sehingga pengguna tidak perlu membaca teks berita secara keseluruhan. Penelitian ini juga membandingkan hasil ringkasan dengan metode LSA dan CLSA.	Menghasilkan suatu sistem yang mampu membuat ringkasan dokumen berita yang akurat. Ringkasan yang memuat intisari dari dokumen asli dengan melakukan perbaikan terhadap kelemahan-kelemahan pada penelitian sebelumnya.
METODE / TEORI YANG	Dokumen uji akan dilakukan tahap <i>preprocessing</i> dengan tahapan <i>tokenisasi</i> ,	Pada penelitian ini teknik peringkasan teks yang digunakan adalah	Metode yang digunakan terdiri dari <i>preprocessing</i> dan <i>processing</i> .	Tapahap peringkasan diawali dengan tahapan <i>preprocessing</i> yaitu	Peringkasan dilakukan dengan menggunakan algoritma LSA dan	Penelitian ini dilakukan dengan menggunakan metode CLSA



DIGUNAKAN	<p><i>stopword</i> dan <i>stemming</i>. Hasil <i>preprocessing</i> berbentuk <i>term</i> kata yang akan di lakukan pembobotan dengan menggunakan <i>tf-idf</i>. Tahap peringkasan dilakukan dengan metode LSA dan MMR.</p>	<p>teknik ekstraksi dengan tahapan <i>preprocessing</i> yang terdiri dari pemecahan kalimat, <i>case folding</i>, <i>tokenisasi</i>, <i>stop words</i> dan <i>stemming</i>. Pembobotan kata menggunakan metode <i>Term Frequency –Inverse Document Frequency</i> (TF-IDF) dan implementasi <i>Cross Method</i> LSA untuk menghasilkan ringkasan.</p>	<p><i>Preprocessing</i> terdiri dari tahapan <i>case folding</i>, pemisahan kalimat, <i>tokenization</i>, <i>stopword removal</i> dan <i>stemming</i>. Tahapan <i>processing</i> LSA terdiri dari pembobotan <i>tf-idf</i>, pembuatan matriks term-kalimat, perhitungan SVD dan pemilihan kalimat hasil perhitungan SVD menggunakan teknik <i>Steinberger&amp;Jezek</i>.</p>	<p><i>parsing</i>, <i>tokenizing</i>, <i>filterisasi</i> dan <i>stemming</i>. Kemudian tahapan <i>processing</i> yang dilakukan dengan tahapan pembobotan <i>tf-idf</i>, perhitungan LSA dengan menambahkan metode perhitungan <i>cross method</i> LSA sebagai penyusunan ringkasan atau ekstraksi ringkasan.</p>	<p>CLSA yang bertujuan untuk membandingkan antara ke-dua algoritma tersebut. Proses peringkasan terdiri dari <i>preprocessing</i> yang terdiri dari penguraian teks dokumen, <i>case folding</i>, <i>tokenisasi</i>, <i>stopword</i>, <i>stemming</i>, dan perhitungan <i>term matriks</i>. Pembobotan kata menggunakan metode <i>tf-idf</i>. Pada proses perhitungan SVD algoritma CLSA melakukan perubahan dari perhitungan SVD algoritma LSA.</p>	<p>dengan tahapan <i>preprocessing</i> yang lebih kompleks seperti <i>parsing</i>, <i>tokenizing</i>, <i>filtering</i>, dan <i>stemming</i>. Melakukan pembobotan kata <i>tf-idf</i>. Menambahkan kesamaan <i>similarity</i> antara judul berita dan dokumen juga memperhitungkan relasi antara kata pada kalimat.</p>
HASIL PENELITIAN	<p>Keluaran dari hasil ringkasan sistem berbentuk dalam bentuk kerangka kerja. Ringkasan dibuat</p>	<p>Penggunaan metode</p>	<p>Pengujian tingkatan akurasi berdasarkan perbandingan antara hasil ringkasan sistem dan hasil ringkasan</p>	<p>Hasil pengujian dari penggunaan LSA dengan penggunaan <i>Cross method</i> LSA sebagai pengestraksi</p>	<p>Hasil pengujian sistem dari uji perbandingan antara metode LSA dan CLSA dengan sama-sama</p>	<p>Sistem memperoleh hasil pengujian dengan menggunakan 10 artikel berita yang</p>

	<p>berdasarkan tingkat kompresi yang berbeda, yaitu 10% dan 20%. Hasil uji sistem dengan dokumen ekonomi yang mempunyai jumlah 96 kalimat hasil <i>kompresi</i> 10% menjadi 9 kalimat dan <i>kompersi</i> 20% menjadi 19 kalimat</p>	<p><i>cross method latent semantic analysis</i> pada penelitian ini mampu membuat sistem menghasilkan ringkasan dengan dokumen uji artikel politik berbahasa Indonesia yang memiliki rata-rata <i>f-measured</i> dengan nilai 69,6%, <i>Recall</i> dengna nilai sebesar 66,7% dan performansi <i>precision</i> sebesar 72,25%.</p>	<p>manual yang dengan perhitungan <i>recall</i>, <i>precision</i>, dan <i>f-measure</i>. Penggunaan metode LSA dengan penambahan teknik <i>Steinberger&amp;Jezek</i> ini mampu menghasilkan ringkasan dengan nilai <i>recall</i> tertinggi yang sebesar 0.71, nilai <i>precision</i> tertinggi 0.75, dan nilai <i>f-measure</i> sebesar 0.71.</p>	<p>ringkasan yang akan dipilih. Dari data uji berupa artikel berita kesehatan. Meode ini menghasilkan ringkasan dengan nilai <i>compression rate</i> 40% dan 50% yang didapati nilai <i>precession</i>, <i>recall</i>, <i>f-measure</i> tertinggi adalah sebesar 0.75 dan 0.743.</p>	<p>menggunakan data uji yang sama menghasilkan hasil <i>f-measure</i> dari CLSA lebih baik dari LSA walaupun perbandinganya tidak jauh berbeda dengan nilai 72% dan 70%. Waktu pada proses perhitungan nilai length CLSA lebih cepat dibandingkan LSA. dan hasil ringkasan CLSA lebih pendek.</p>	<p>dibandingkan dengan ringkasan sistem. Hasil ringkasan memperoleh nilai rata-rata Rouge-1 (f-measure 0,5811, precession 0,5757, recal 0,6194), Rouge-2 (f-measure 0,47022, precession 0,4673, recal 0,5005), Rouge-L (f-measure 0,5936, precession 0,5809, recal 0,6332).</p>
--	--	--	---	--	---	---

## B. Landasan Teori

### 1. *Text mining*

*Text mining* (penambangan teks) adalah salah satu teknik yang dapat digunakan untuk melakukan klasifikasi dimana, *text mining* merupakan variasi dari *data mining* yang berusaha menemukan pola yang menarik dari sekumpulan data *tekstual* yang berjumlah besar. Selain klasifikasi, *text mining* juga digunakan untuk menangani masalah *clustering*, *information extraction*, dan *information retrieval* (IR) (Feldman & Sanger, 2007).

Dalam *text mining* berbeda dengan dengan *data mining* dimana Pada *data mining* data yang diekstrak berasal dari pola-pola tertentu dan terstruktur, sedangkan *text mining* sumber data yang digunakan berasal dari teks yang relatif tidak terstruktur karena menggunakan tata bahasa manusia atau biasa disebut (*natural language*).

Menurut (Indrawati, 2010) *text mining* adalah penambang data yang berupa teks dimana sumber data biasanya didapatkan dari suatu dokumen, dengan tujuan untuk mencari kata-kata yang dapat mewakili isi dari dokumen asli sehingga dapat dilakukan analisa keterhubungan antara dokumen. *Text mining* juga merupakan proses menemukan informasi atau *trend* baru yang sebelumnya tidak terungkap atau belum diketahui dengan memproses dan menganalisa data-data dalam jumlah yang besar.

Dalam penerapannya pada peringkasan teks, penerapan *text mining* digunakan sebagai tahapan untuk menentukan seberapa jauh

keterhubungan dengan kata atau *term*, yang dimana kata atau *term* tersebut dalam bentuk dokumen yang akan diproses, namun dokumen-dokumen tersebut sebelumnya belum terstruktur. Jadi dibutuhkan tahapan untuk menambang *term* atau kata tersebut yang terdapat dalam dokumen sehingga bisa memperoleh informasi dengan lebih tepat, jelas serta akurat.

## **2. *Natural Language Processing (NLP)***

Pemrosesan Bahasa Alami (*Natural language processing/NLP*) merupakan cabang kecerdasan buatan yang membantu komputer memahami, menafsirkan, dan memanipulasi bahasa manusia. NLP menarik dari banyak disiplin ilmu, termasuk ilmu komputer dan linguistik komputasional, dalam usahanya untuk mengisi kesenjangan antara komunikasi manusia dan pemahaman komputer (Pustejovsky & Stubbs, n.d.).

*Natural language processing (NLP)* pada pengaplikasiannya berkaitan dengan bagaimana suatu komputer dapat digunakan untuk memahami atau mengerti dalam memanipulasi teks bahasa alami (*natural language*) untuk mendapatkan informasi tertentu. Dengan perantaraan bahasa alami (*natural language*) inilah, manusia bisa berinteraksi dengan komputer. NLP digunakan dalam pemrosesan dokumen karena *user* menentukan keterkaitan dari dokumen dengan membaca dan menganalisis nya. Jika sistem dapat melakukan analisis dokumen secara otomatis, maka proses pencarian dokumen yang relevan akan lebih mudah (Indrawati, 2010).

#### **a. Komponen Utama Bahasa Alami**

Pengolahan bahasa alami terdiri dari tiga bagian utama, yaitu :  
*parser*, sistem representasi pengetahuan dan *output translator*.

##### **1) Parser**

Kemampuan suatu sistem yang mengambil kalimat *input* bahasa alami dan menguraikannya ke dalam beberapa bagian gramatikal (kata benda, kata kerja, kata sifat, dan lain-lain).

##### **2) Sistem Representasi Pengetahuan**

Suatu sistem yang menganalisis *output parser* untuk menentukan maknanya.

##### **3) Output Translator**

Suatu terjemahan yang merepresentasikan sistem pengetahuan dan melakukan langkah-langkah yang bisa berupa jawaban atas bahasa alami atau *output* khusus yang sesuai dengan program komputer lainnya.

#### **b. Aplikasi pengolahan bahasa alami**

*Natural language processing* (NLP) mempunyai aplikasi yang sangat luas. Beberapa diantaranya yaitu *summarization*. Pembuatan ringkasan dari sekumpulan konten dokumen atau Dengan menggunakan aplikasi

ini, user bisa dibantu untuk mengkonversikan dokumen teks yang besar ke dalam bentuk slide presentasi.

### **3. Text Summarization**

*Text summarization* (peringkasan teks) adalah suatu proses penyulingan sebagian besar informasi penting dari sumber (beberapa sumber) untuk menghasilkan suatu ringkasan bagi pemakai atau pekerjaan tertentu (Klein, Hirschman, Firmin, & Diego, 1998). Menurut (Saputra, Jerry. Fachrurrozi, 2017) ringkasan adalah bentuk singkat dari sebuah teks yang dibuat dari satu dokumen atau lebih, dan mengambil informasi-informasi penting yang terdapat pada dokumen asli. Suatu ringkasan yang dihasilkan berisi informasi yang memuat inti dari isi dokumen asli dengan jumlah kalimat yang tak tidak lebih dari setengah teks aslinya.

#### **a. Pendekatan dalam peringkasan teks**

Pendekatan yang umum dalam membuat ringkasan teks digolongkan menjadi dua, keduanya yaitu *ekstraktif* dan *abstraktif*.

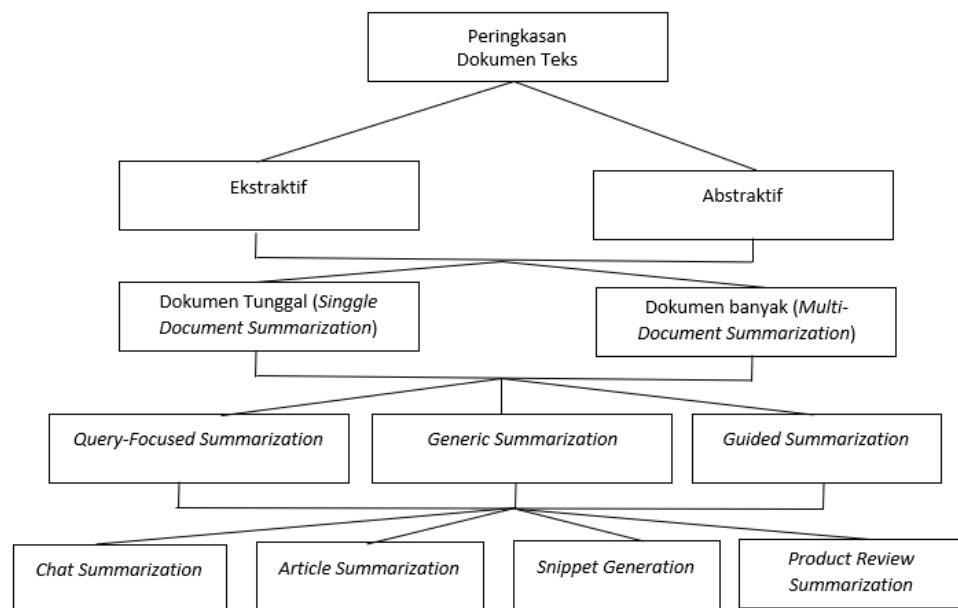
##### **1) Metode *ekstraktif***

Metode peringkasan *ekstraktif* berupaya mengekstrak hal-hal penting kalimat pada suatu dokumen asli kemudian ditampilkan kembali dengan bentuk yang lebih sederhana. Mudahnya dokumen kalimat yang memuat informasi penting diekstrak untuk mengumpulkan informasi penting sehingga mendapatkan gambaran umum tentang konten dari dokumen asli.

## 2) Metode *abstraktif*

Peringkasan teks abstraktif merupakan ringkasan yang menghasilkan interpretasi dari teks asli yang ada dalam bentuk abstrak. Kalimat yang terdapat pada dokumen ditransformasikan kembali menjadi kalimat yang lebih singkat. Peringkasan *abstraktif* membuat ringkasan dari kalimat ataupun frase yang berbeda, tetapi tetap memiliki intisari arti yang sama dari sumber dokumen.

Penjabaran jenis dan pengaplikasian peringkasan dokumen teks menurut (Yuliska & Syaliman, 2020) dijelaskan pada gambar 2.1 berikut.



Gambar 2. 1 Jenis dan pengaplikasian peringkasan dokumen teks

Jenis peringkasan dokumen teks dapat dibagi menjadi peringkasan dokumen tunggal atau *single document summarization* dan peringkasan

dokumen jamak atau *multi-document summarization*. Peringkasan dokumen dapat dilakukan berdasarkan pada fakta-fakta tertentu (*Guided summarization*), berdasarkan pada informasi penting yang relevan dengan *query* pengguna (*query-focused summarization*), atau hanya berdasarkan pemilihan informasi yang dianggap penting (*Generic summarization*). Dokumen yang akan diringkaskan dapat berupa dokumen *chat*, *article*, *snippet generation* dan *product review*.

## **b. Fitur peringkasan teks**

Untuk menentukan kalimat ringkasan maka digunakan beberapa fitur sebagai dasar dalam pembuatan ringkasan.

### **1) Frekuensi**

Kata yang dianggap penting adalah kata yang sering muncul dalam sebuah dokumen. Semakin sering muncul, maka perhitungan skor untuk kata tersebut semakin tinggi. Pengukuran yang umum digunakan untuk menghitung frekuensi kata adalah TF-IDF.

### **2) Lokasi**

Kalimat utama dalam suatu paragraf biasanya terdapat pada bagian awal dan akhir dari sebuah paragraf, sehingga kalimat ini memiliki kesempatan yang lebih besar untuk diikutsertakan dalam sebuah ringkasan daripada kalimat yang berada di tengah paragraph.

### **3) Cue Method**



Pentingnya suatu ide biasanya tersirat dari kalimat: *in summary*, *in conclusion*, *the paper describes*, atau kesimpulannya adalah, ringkasannya.

#### 4) Judul/Kepala Berita

Kata yang ada pada judul dan kepala/pokok berita besar kemungkinannya berhubungan dengan ringkasan. Kata-kata yang ada pada sebuah judul juga mengindikasikan topik dari suatu dokumen.

#### 5) Panjang Kalimat

Pada umumnya, kalimat yang terlalu panjang ataupun pendek tidak cocok digunakan dalam sebuah ringkasan.

#### 6) Kemiripan

Kemiripan dapat dikalkulasi dengan pengetahuan linguistik. Hal ini mengindikasi kemiripan kalimat yang digunakan dalam judul dan dalam isi dokumen.

#### 7) Kata Benda

Penggunaan kata benda yang tepat harus diperhatikan. Ringkasan harus menggunakan kata benda yang tepat, misalnya nama seseorang, nama tempat ataupun organisasi.

#### 8) Kedekatan

Jarak antara kata dalam sebuah *entity* menjadi sebuah faktor untuk membuat relasi antara *entity*.

### **c. *Online tools* peringkasan teks**

Online tools peringkasan teks merupakan *website* penyedia layanan peringkasan dokumen teks. Terdapat beberapa fitur pada layanan peringkasan teks salah satunya user bisa memilih berapa kalimat ringkasan yang diinginkan. Beberapa *website* peringkasan teks dengan berbagai metode peringkasan yang dapat di gunakan sebagai acuan atau referensi untuk pengembangan sistem. *Online tools* peringkasan teks tersebut sebagai berikut.

- 1) <http://textsummarization.net/>
- 2) <https://indo-texsum.herokuapp.com/>
- 3) <https://www.tools4noobs.com/summarize/>
- 4) <https://autosummarizer.com/>
- 5) <https://resoomer.com/en/>

### **4. *Text preprocessing***

*Text pre-processing* (pemrosesa awal) merupakan tahapan utama dalam pemrosesan teks. *Pre-processing* bertujuan untuk menghasilkan indeks kata dari dokumen teks yang dilakukan agar dokumen dapat diproses ke tahap selanjutnya (Savanti, Gotami, & Dewi, 2018). Mudahnya *text pre-processing* merupakan tahapan dimana teks dokumen yang sebelumnya tidak terstruktur diolah menjadi data yang terstruktur yang bertujuan untuk membuat teks dokumen lebih mudah diolah. Tahapan *pre-processing*

menghasilkan teks yang mempunyai skor atau nilai sehingga bisa digunakan dalam acuan pada proses selanjutnya untuk melakukan peringkasan teks. Pada penelitian ini terdapat beberapa tahapan pre-processing yaitu sebagai berikut.

**a. *Parsing***

*Parsing* merupakan tahapan pemecahan disetiap kalimat pada suatu dokumen dengan simbol titik sebagai pembatas kalimat. Setiap dokumen yang telah dipecah akan dimasukkan sebagai list kalimat. Keluaran dari hasil *parsing* berupa kumpulan kalimat terpisah dari setiap dokumen yang akan digunakan pada proses selanjutnya. Contoh aturan terdapat pada tabel 2.2. Aturan *parsing* sebagai berikut.

Tabel 2. 2. Aturan *Parsing*

Kondisi	Aksi
Inputan data terdapat titik(.).	Maka akan memecah kalimat menjadi dokumen baru.

**b. *Case folding***

*Case folding* merupakan tahapan yang mengubah semua huruf kapital dalam dokumen menjadi huruf kecil. Hanya huruf yang diterima yaitu 'a' sampai dengan huruf 'z'. Karakter selain huruf akan dihilangkan dan dianggap sebagai delimiter. Contoh penggunaan terdapat pada tabel 2.3. Aturan *case folding* sebagai berikut.

Tabel 2. 3. Aturan *case folding*

Kondisi	Aksi
Inputan data latih memiliki huruf kapital [A.....Z].	Maka akan mengubah semua inputan tersebut menjadi huruf kecil [a.....z] semua.
Inputan data latih memiliki karakter Symbol	Maka akan menghapus karakter symbol tersebut dari inputan
Inputan data latih memiliki huruf kecil	Tidak ada aksi
Inputan data latih memiliki spasi	Tidak ada aksi

### c. *Tokenizing*

*Tokenizing* adalah tahap pemotongan tiap kata yang menyusunnya, atau memisah tiap-tiap kata dalam satu dokumen yang dilakukan pada seluruh kalimat. Selain itu, spasi digunakan untuk memisahkan antara kata tersebut. Contoh penggunaan *tokenizing* terdapat pada tabel 2.4. Aturan *tokenizing* sebagai berikut.

Tabel 2. 4. Aturan *tokenizing*

Kondisi	Aksi
Jika inputan data uji bertemu spasi	Maka akan memecah dari deskripsi data uji menjadi bab-bab per bagian kata atau string.
Jika Inputan data latih memiliki Huruf	Tidak ada aksi

### d. *Filtering*

Tahap *filtering* adalah tahap mengambil kata-kata penting dari hasil *tokenizing*. Proses *filtering* dapat menggunakan algoritma *stoplist* (membuang kata yang kurang penting) atau *wordlist* (menyimpan kata penting). *Stoplist* /

*stopword* adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan *bag-of-words*. Pada dokumen berbahasa Indonesia contoh *stopword* adalah “yang”, “dan”, “di”, “dari” dan lain – lain. Contoh penggunaan *filtering* terdapat pada tabel 2.5. Atruran *filtering* sebagai berikut.

Tabel 2. 5. Aturan *filtering*

Kondisi	Aksi
Jika inputan data uji terdapat kata-kata yang tidak deskriptif	Maka akan melakukan pembuangan pada kata yang tidak deskriptif
Jika Inputan data latih memiliki kata-kata yang tidak deskriptif	Tidak ada aksi

#### e. *Stemming*

*Stemming* adalah tahapan untuk mencari *root* kata dari tiap kata hasil *filtering* atau proses untuk memecah kata menjadi kata dasar. Pada dokumen berbahasa Indonesia untuk memecah kata menjadi kata dasar harus sesuai dengan aturan Bahasa Indonesia salah satu algoritma yang digunakan dalam pemecahan kata berbahasa Indonesia adalah algoritma Nazief & Adriani. Contoh penggunaan *stemming* terdapat pada tabel 2.6. Aturan *stemming* sebagai berikut.

Tabel 2. 6. Aturan *stemming*

Kondisi	Aksi
Jika kata baku atau kata singkatan dari kata dasar terdapat dalam kamus atau <i>database</i> algoritma Nazief & Adriani.	Maka akan dilakukan pemotongan kata menjadi kata dasar
Jika kata tidak baku atau kata singkatan dari kata dasar tidak terdapat dalam	Tidak ada aksi

kamus atau <i>database</i> algoritma Nazief & Adriani.	
--	--

## 5. *Term Frequency-Inverse Document Frequency (TF-IDF)*

*Term Frequency-Inverse Document Frequency (TFIDF)* adalah suatu metode yang merupakan integrasi antar *term frequency* (TF), dan *inverse document frequency* (IDF) (Widyasanti, Gede, Putra, Kadek, & Rusjyanthi, 2018). *Term Frequency* (TF) merupakan jumlah frekuensi kemunculan suatu *term* pada suatu kalimat, sedangkan *Inverse Document Frequency* (IDF) adalah perhitungan logaritma pembagian antar total jumlah kalimat dengan frekuensi kalimat yang memuat suatu *term* (Saputra, Jerry. Fachrurrozi, 2017). Pada penelitian peringkasan teks yang dilakukan oleh (Savanti et al., 2018) tf-idf sebagai pembobotan kata dihitung dengan menggunakan persamaan berikut.

$$TF-IDF = TF * IDF \quad (2.1)$$

$$TF = 1 + \log tf \quad (2.2)$$

$$IDF = \log \frac{D}{df(t)} \quad (2.3)$$

Keterangan :

$TF$  = banyaknya *term* yang terdapat pada sebuah dokumen

$IDF$  = hubungan antara banyaknya dokumen yang memuat *term* dengan jumlah dokumen kalimat.

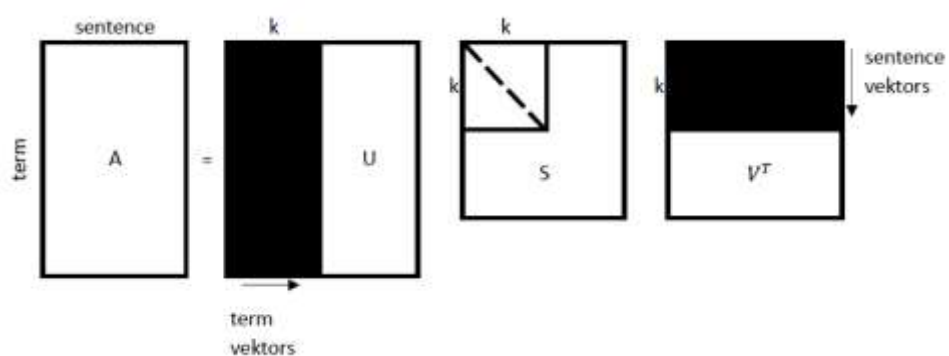
$D$  = total dari dokumen

$df$  = banyaknya dokumen yang mengandung *term*.

Fungsi pembobotan kata dengan TFIDF adalah untuk mencari representasi nilai dari tiap-tiap dokumen dari suatu kumpulan *data training* (*training set*).

## 6. Singular Value Decomposition

Singular Value Decomposition (SVD) merupakan salah satu teknik reduksi pada *aljabar linear* yang bermanfaat untuk memperkecil nilai kompleksitas dalam pemrosesan kata pada matriks. SVD melakukan proses penguraian (*dekomposisi*) suatu matriks menjadi tiga buah matriks baru, yaitu matriks *orthogonal*  $U$ , matriks *diagonal*  $S$ , dan *transpose* matriks *orthogonal*  $V$  (Saputra, Jerry. Fachrurrozi, 2017). Proses dekomposisi akan memfaktorkan sebuah matriks menjadi lebih dari satu matriks. Gambaran proses SVD dapat dilihat pada gambar 2.1 berikut.



Gambar 2. 2 Proses *matriks* SVD

Rumus SDV yang dipaparkan (Saputra, Jerry. Fachrurrozi, 2017) pada penelitiannya sebagai berikut.

$$Am^{\times n} = Um^{\times n}.Sn^{\times n}.V^Tn^{\times n} \quad (2.4)$$

Keterangan :

$Am^{\times n}$  = matriks A dengan nilai  $m \geq n$

$Um^{\times n}$  = matriks berupa vector sebelah kiri berukuran  $m \times n$

$Sn^{\times n}$  = matriks *diagonal* berukuran  $n \times n$ , dengan elemen matriks positif atau nol

$V^Tn^{\times n}$  = matriks berupa vector sebelah kiri berukuran  $n \times n$ , yang merupakan transpose matriks V.

Untuk menentukan nilai SVD dilakukan pencarian nilai *eigen value* dan *eigen vector* dari matriks  $A^T A$ . Eigen value dari matriks  $A^T A$  membentuk matriks S atau nilai *singular*. Nilai *singular* meruakan elemen-elemen diagonal dari S dan disusun dengan urutan menurun. Eigen vector dari  $A^T A$  membentuk matriks V.

## 7. Latent Semantic Analysis (LSA)

Latent sematic analysis (LSA) menurut bahasa terbagi atas beberapa kata yaitu *latent* dan *semantic*. *Latent* yang berarti tersembunyi atau sesuatu yang masih belum ditemukan atau belum terlihat, dan *semantic* yang berarti hubungan kata dengan konsepsi makna pada suatu kata atau kalimat (Mandar & Gunawan, 2017). LSA menurut (Savanti et al., 2018) merupakan metode yang digunakan untuk menganalisa hubungan antara sebuah *frase* ataupun kalimat dengan sekumpulan dokumen. Jadi mudahnya LSA adalah peruraian makna atau analisa makna yag masih tersembunyi dari suatu



bahasa yang bertujuan untuk memperoleh informasi yang penting dari suatu dokumen.

Konsep LSA menerapkan dua fitur yaitu matriks dan SVD, hasil pembobotan kata akan dijadikan matriks. Matriks hasil pembobotan tersebut selanjutnya diolah dengan pemanfaatan SVD untuk menemukan kesamaan atau hubungan antar kata dalam suatu kalimat. Langkah-langkah LSA dalam penelitian yang dilakukan (Mandar & Gunawan, 2017) sebagai berikut.

- a. Membentuk matriks  $A_{mn}$  yaitu matriks yang memuat *term* pada suatu dokumen.
- b. Membuat matriks *right singular vektor* ( $V$ ) dan nilai dari *eigenvalue*, matriks  $V$  terbentuk dari hasil *eigenvector* matriks  $A^T A$ .
- c. Membentuk matriks  $S$  yaitu hasil dari akar nilai tertinggi *eigenvalue* matriks  $A^T A$ .
- d. Menghitung *length* pada setiap nilai matriks  $V^T$  dengan rumus berikut.

$$S_k = \sqrt{\sum_i^n (v^T)_{k1}^2 \cdot S_1^2} \quad (2.5)$$

- e. Hasil ringkasan ditentukan nilai skor tertinggi dari hasil perhitungan matriks kalimat.

## 8. Cross Latent Semantic Analysis (CLSA)

CLSA atau *cross latent semantic analysis* merupakan salah satu metode peringkasan teks yang tercipta dari hasil pengembangan dari metode *latent semantic analysis*. Cross pada CLSA yang berarti persilangan atau proses

silang dari perhitungan LSA, dengan mengubah beberapa tahapan-tahapan pada proses perhitungan matriks SVD. Proses silang ini bertujuan untuk mempercepat peringkasan. CLSA dan LSA sama-sama menggunakan fitur matriks dan SVD dalam proses peringkasan teks. Berikut ini merupakan langkah langkah CLSA pada penelitian yang dilakukan oleh (Mandar & Gunawan, 2017) pada tahapan pemilihan kata ringkasan.

- a. Membuat matriks berukuran  $A_{m \times n}$  atau  $A_{mn}$  yaitu matriks hasil pembobotan *tf-idf* yang diubah kedalam bentuk matriks dokumen.
- b. Mencari nilai *eigenvector* ( $V$ ) dan nilai *eigenvalue* dari perhitungan matriks  $A^T A$ .
- c. Membuat matriks  $S$  dengan cara menemukan nilai *singular* dengan cara mengurutkan nilai yang tertinggi dari *eigenvalue* yang diakarkan.
- d. *Transpose* pada *eigenvector* atau matriks  $V$  untuk menghasilkan matriks  $V^T$ .
- e. Melakukan perhitungan nilai rata-rata matriks  $V^T$ .
- f. Melakukan penyeleksian matriks  $V^T$ , jika nilai suatu *term* lebih kecil dari nilai rata-rata pada dokumen kalimat, maka nilai pada matriks  $V^T$  diubah menjadi 0 dan membentuk matriks  $V$  yang baru.
- g. Menghitung panjang matriks  $V^T$  dengan rumus (2.5).

Dimana *length* adalah panjang vektor  $k$  pada kalimat yang dimodifikasi oleh *laten vektor*.  $n$  adalah jumlah ruang dimensi baru. Hasil dari *length* terbesar pada setiap dokumen kalimat akan dijadikan ringkasan.

- h. Baris - baris pada matriks atau kalimat - kalimat yang mempunyai nilai *length* yang tinggi akan dijadikan sebagai ringkasan.

### 9. Recall-Oriented Understudy for Gisting Evaluation (ROUGE)

Rouge atau *Recall-Oriented Understudy for Gisting Evaluation* adalah suatu teknik untuk mengevaluasi perbandingan antara hasil dari ringkasan yang dihasilkan dari ATS (*Automated Text Summarization*) dengan hasil ringkasan yang dibuat manusia. Pengujian diukur dengan perhitungan dari nilai *precision*, nilai *recall*, nilai *f-measure* dan nilai dari akurasi. Untuk menghitung nilai tersebut digunakan rumus berikut.

$$Precision (P) = \frac{tp}{(tp+fp)} \quad (2.6)$$

$$Recall (R) = \frac{tp}{(tp+fn)} \quad (2.7)$$

$$F - measure = \frac{2 RP}{(P+R)} \quad (2.8)$$

$$Akurasi = \frac{tp+tn}{(tp+fp+tn+fn)} \quad (2.9)$$

Keterangan :

*True positive* (tp) = yaitu kalimat yang terdapat pada hasil ringkasan manual yang juga muncul pada hasil ringkasan sistem.

*False positive* (fp) = yaitu kalimat hasil ringkasan yang terdapat pada sistem tetapi tidak ada pada ringkasan manual.

*False negative* (fn) = yaitu kalimat ringkasan yang terdapat pada hasil ringkasan manual tetapi tidak terdapat pada hasil ringkasan sistem.

*True negative* (tn) = yaitu kalimat tidak ada pada hasil ringkasan manual maupun hasil ringkasan sistem.

ROUGE juga memiliki lima macam ukuran berbeda sebagai berikut :

**a. *N-gram Co-Occurrence Statistics (ROUGE-N)***

ROUGE- N merupakan perhitungan rouge dengan penarikan n-gram antara ringkasan kandidat dan kumpulan ringkasan referensi. Dimana metode ini menggunakan setiap kata yang cocok antara *candidate summary* (ringkasan yang dihasilkan ATS) dengan *references summary* (ringkasan ideal yang dibuat oleh manusia) sebagai ukurannya dan tidak memperhatikan urutan kata-kata yang terdapat pada ringkasan.

**b. *Longest Common Subsequence (ROUGE-L)***

ROUGE-L adalah perhitungan rouge dengan langsung membandingkan setiap ringkasan yang dihasilkan ATS (*Automated Text Summarization*) dengan kalimat yang ada diringkasan referensi. ROUGE-L tidak membutuhkan kecocokan berurutan tetapi kecocokan berurutan yang mencerminkan urutan kata tingkat kalimat sebagai n-gram. ROUGE-L secara otomatis menyertakan n-gram umum berurutan terpanjang, oleh karena itu panjang n-gram yang telah ditentukan tidak diperlukan.

**c. *Weighted Longest Common Subsequence (ROUGE-W)***

ROUGE-W merupakan perkembangan dari metode ROUGE-L dengan memandang kalimat-kalimat yang ada pada *candidate summary* sebagai rangkaian kata-kata yang nantinya akan dicocokkan pada *references summary* menggunakan pemrograman dinamis.

**d. Skip-Bigram Co-Occurrence Statistics (ROUGE-S)**

ROUGE-S adalah pasangan kata apa pun dalam urutan kalimat yang memungkinkan adanya kesalahan. Skip-bigram mengukur rasio antara terjemahan kandidat dari satu set terjemahan referensi. Skip-bigram melakukan perhitungan terhadap semua pasangan kata yang cocok secara berurutan.

**C. Studi kasus**

**1. Pengumpulan Data**

Data yang digunakan sebagai pengujian hitungan manual diambil dari berita pada halaman [www.kompas.com](http://www.kompas.com), [www.detik.com](http://www.detik.com), [www.krjogja.com](http://www.krjogja.com) sebanyak 300 berita sebagai data uji. Berita tersebut kemudian dikumpulkan dalam bentuk file excel. Setiap data uji tersebut akan di ujikan satu persatu. Terkhusus untuk penelitian ini diambil salah satu dari 300 berita dari dataset tersebut. Data berita tersebut dapat dilihat pada Tabel 2.7. *Collect Data* sebagai berikut.

Tabel 2. 7. *Collect Data*

Dokumenen Asli
Berita
Ikatan Dokter Anak Indonesia (IDAI) memberikan sejumlah rekomendasi untuk mencegah penularan Covid-19 kepada anak di masa pandemi. Salah satu yang direkomendasikan adalah agar anak tidak keluar rumah selama situasi Covid-19 di Indonesia belum memenuhi kriteria epidemiologi badan kesehatan dunia (WHO). "Kami merekomendasikan agar anak-anak jangan keluar rumah dulu. Termasuk untuk kegiatan tatap muka di sekolah," ujar anggota Tim Satgas Penanganan Covid-19 IDAI, Yogi Prawira. Rekomendasi untuk tidak keluar rumah ini, lanjut dia, berlaku hingga daerah tempat tinggal anak-anak dianggap sudah dapat mengatasi penularan Covid-19 lewat transmisi

lokal. Namun, kata Yogi, rekomendasi ini dikecualikan jika ada keperluan mendesak yang membuat anak untuk keluar rumah.

## 2. *Preprocessing Text*

Data sampel yang masih berbentuk paragraf akan diolah untuk mendapatkan bobot tiap kata-kata pada data sampel, untuk itu di lakukan tahapan *preprocessing*. Pada bagian *preprocessing text* terbagi menjadi beberapa tahapan di antaranya sebagai berikut :

### a. Parsing

Tahap pertama dalam *preprocessing* ini adalah Parsing atau tahapan pemecahan kalimat. Karena data sampel yang berupa paragraph maka akan di pisah dengan memecahnya menjadi beberapa dokumen dengan simbol titik sebagai pemisah dokumen. Setiap dokumen yang telah dipecah akan dimasukkan sebagai list kalimat. Berikut adalah tahapan *parsing* bisa di lihat pada Tabel 2.8. Tahapan *Parsing* sebagai berikut.

Tabel 2. 8. Tahapan Parsing

Parsing	
d1	Ikatan Dokter Anak Indonesia (IDAI) memberikan sejumlah rekomendasi untuk mencegah penularan Covid-19 kepada anak di masa pandemi.
d2	Salah satu yang direkomendasikan adalah agar anak tidak keluar rumah selama situasi Covid-19 di Indonesia belum memenuhi kriteria epidemiologi badan kesehatan dunia (WHO).
d3	"Kami merekomendasikan agar anak-anak jangan keluar rumah dulu
d4	Termasuk untuk kegiatan tatap muka di sekolah," ujar anggota Tim Satgas Penanganan Covid-19 IDAI, Yogi Prawira.
d5	Rekomendasi untuk tidak keluar rumah ini, berlaku hingga daerah tempat tinggal anak-anak dianggap sudah dapat mengatasi penularan Covid-19 lewat transmisi lokal

d6	Namun, kata Yogi, rekomendasi ini dikecualikan jika ada keperluan mendesak yang membuat anak untuk keluar rumah.
----	--

### b. Case folding

Tahapan *preprocessing* selanjutnya yaitu case folding. Pada tahapan case folding data akan diubah menjadi huruf kecil. Hanya huruf yang diterima yaitu 'a' sampai dengan huruf 'z'. data yang berupa simbol akan dihilangkan dan dianggap sebagai delimiter. Berikut adalah tahapan case folding bisa di lihat pada Tabel 2.9. Tahapan *Case folding* sebagai berikut.

Tabel 2.9. Tahapan *Case folding*

Case Folding	
d1	ikatan dokter anak indonesia idai memberikan sejumlah rekomendasi untuk mencegah penularan covid19 kepada anak di masa pandemi
d2	salah satu yang direkomendasikan adalah agar anak tidak keluar rumah selama situasi covid19 di indonesia belum memenuhi kriteria epidemiologi badan kesehatan dunia who
d3	kami merekomendasikan agar anak anak jangan keluar rumah dulu
d4	termasuk untuk kegiatan tatap muka di sekolah ujar anggota tim Satgas Penanganan covid19 idai yogi prawira
d5	rekomendasi untuk tidak keluar rumah ini berlaku hingga daerah tempat tinggal anak anak dianggap sudah dapat mengatasi penularan covid19 lewat transmisi lokal
d6	namun kata yogi rekomendasi ini dikecualikan jika ada keperluan mendesak yang membuat anak untuk keluar rumah

### c. Tokenizing

Tahapan *preprocessing* selanjutnya yaitu tokenizing. *Tokenizing* merupakan tahapan pemotongan tiap kata yang menyusunnya, atau memisah tiap-tiap kata dalam satu dokumen yang dilakukan pada seluruh

kalimat. Berikut adalah tahapan Tokenisasi bisa di lihat pada Tabel 2.10.

Tahapan *Tokenizing* sebagai berikut.

Tabel 2.10. Tahapan *Tokenizing*

Tokenizing					
d1	d2	d3	d4	d5	d6
ikatan	salah	kami	termasuk	rekomendasi	namun
dokter	satu	merekomendasikan	untuk	untuk	kata
anak	yang	agar	kegiatan	tidak	yogi
indonesia	direkomendasikan	anak	tatap	keluar	rekomendasi
idai	adalah	anak	muka	rumah	ini
memberikan	agar	jangan	di	ini	jika
sejumlah	anak	keluar	sekolah	dia	ada
rekomendasi	tidak	rumah	ujar	berlaku	keperluan
untuk	keluar	dulu	anggota	hingga	mendesak
mencegah	rumah		tim	daerah	yang
penularan	selama		satgas	tempat	membuat
covid19	situasi		penanganan	tinggal	anak
kepada	covid19		covid19	anak	untuk
anak	di		idai	anak	keluar
di	indnesia		yogi	dianggap	rumah
masa	belum		prawira	sudah	
pandemi	memenuhi			dapat	
	kriteria			mengatasi	
	epidemilogi			penularan	
	badan			covid19	
	kesehatan			lewat	
	dunia			transmisi	
	who			lokal	

#### d. *Filtering*

Tahapan *preprocessing* selanjutnya yaitu Filteing. Proses *filtering* dapat menggunakan algoritma *stoplist* (membuang kata yang kurang penting) atau *wordlist* (menyimpan kata penting). Berikut adalah tahapan *Filtering* bisa di lihat pada Tabel 2.11. Tahapan *Filtering* sebagai berikut.

Tabel 2.11. Tahapan *Filtering*.

Filtering					
d1	d2	d3	d4	d5	d6
ikatan	salah	merekomendasikan	termasuk	rekomendasi	kata
dokter	satu	anak	kegiatan	keluar	yogi
anak	direkomendasikan	anak	tatap	rumah	rekomendasi
indonesia	anak	jangan	muka	berlaku	perlu



idai	keluar	keluar	sekolah	hingga	desak
memberikan	rumah	rumah	ujar	daerah	buat
sejumlah	selama	dulu	anggota	tempat	anak
rekomendasi	situasi		tim	tinggal	keluar
mencegah	covid19		satgas	anak	rumah
penularan	indnesia		penanganan	anak	
covid19	memenuhi		covid19	dianggap	
kepada	kriteria		idai	mengatasi	
anak	epidemilogi		yogi	penularan	
masa	badan		prawira	covid19	
pandemi	kesehatan			lewat	
	dunia			transmisi	
	who			lokal	

#### e. Stemming

Proses terakhir setelah *filtering* adalah *stemming*. *Stemming* merupakan tahapan untuk mencari *root* kata dari tiap kata hasil *filtering* atau proses untuk memecah kata menjadi kata dasar atau proses menghilangkan kata yang berimbuhan sehingga menjadi kata dasar. Hasil *stemming* seperti yang terlihat pada Tabel 2.12. Tahapan *Stemming* sebagai berikut.

Tabel 2.12. Tahapan *Stemming*

Stemming					
d1	d2	d3	d4	d5	d6
ikat	salah	rekomendasi	masuk	rekomendasi	kata
dokter	satu	anak	giat	keluar	yogi
anak	rekomendasi	anak	tatap	rumah	rekomendasi
indonesia	anak	jangan	muka	laku	perlu
idai	keluar	keluar	sekolah	hingga	desak
beri	rumah	rumah	ujar	daerah	buat
jumlah	lama	dulu	anggota	tempat	anak
rekomendasi	situasi		tim	tinggal	keluar
cegah	covid19		satgas	anak	rumah
tular	indnesia		tangan	anak	
covid19	penuh		covid19	anggap	
anak	kriteria		idai	atasi	
masa	epidemilogi		yogi	tular	
pandemi	badan		prawira	covid19	
	sehat			lewat	

	dunia			transmisi	
	who			lokal	

### 3. Pembobotan TF-IDF

Proses selanjutnya yang harus dilakukan setelah melalui *text preprocessing* adalah melakukan pembobotan TF-IDF. Perhitungan dilakukan dengan mencari pembobotan setiap term pada setiap dokumen. Term merupakan kata dasar yang diperoleh dari *text preprocessing* sebelumnya. Proses pembobotan dokumen secara lengkap ditunjukkan pada Tabel 2.13. Pembobotan dokumen dihitung dengan menggunakan algoritma TF-IDF dengan rumus berikut:

$$TF - IDF(t_i, d_j) = tf(t_i, d_j) * \log \frac{N}{N(t_i)} \quad (2.10)$$

Tabel 2.13. Tahapan Tf-idf

Term	TF						df	D/df	IDF	idf+1	TF.IDF Term weigth (W)					
	Term Frequency								ln(D/df)		W = tf * (idf+1)					
	d1	d2	d3	d4	d5	d6					D1	D2	D3	D4	D5	D6
anak	2	1	1	0	1	1	5	1.2	0.182322	1.182322	2.364643	1.182322	1.182322	0	1.182322	1.182322
anggap	0	0	0	0	1	0	1	6	1.791759	2.791759	0	0	0	0	2.791759	0
anggota	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0
atasi	0	0	0	0	1	0	1	6	1.791759	2.791759	0	0	0	0	2.791759	0
badan	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
beri	1	0	0	0	0	0	1	6	1.791759	2.791759	2.791759	0	0	0	0	0
buat	0	0	0	0	0	1	1	6	1.791759	2.791759	0	0	0	0	0	2.791759
cegah	1	0	0	0	0	0	1	6	1.791759	2.791759	2.791759	0	0	0	0	0
covid19	1	1	0	1	1	0	4	1.5	0.405465	1.405465	1.405465	1.405465	0	1.405465	1.405465	0
daerah	0	0	0	0	1	0	1	6	1.791759	2.791759	0	0	0	0	2.791759	0
desak	0	0	0	0	0	1	1	6	1.791759	2.791759	0	0	0	0	0	2.791759
dokter	1	0	0	0	0	0	1	6	1.791759	2.791759	2.791759	0	0	0	0	0
dulu	0	0	1	0	0	0	1	6	1.791759	2.791759	0	0	2.791759	0	0	0
dunia	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
epidemiologi	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
giat	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0
hingga	0	0	0	0	1	0	1	6	1.791759	2.791759	0	0	0	0	2.791759	0
idai	1	0	0	1	0	0	2	3	1.098612	2.098612	2.098612	0	0	2.098612	0	0
ikat	1	0	0	0	0	0	1	6	1.791759	2.791759	2.791759	0	0	0	0	0
indonesia	1	0	0	0	0	0	1	6	1.791759	2.791759	2.791759	0	0	0	0	0
jangan	0	0	1	0	0	0	1	6	1.791759	2.791759	0	0	2.791759	0	0	0
jumlah	1	0	0	0	0	0	1	6	1.791759	2.791759	2.791759	0	0	0	0	0

kata	0	0	0	0	0	1	1	6	1.791759	2.791759	0	0	0	0	0	2.791759
kecuali	0	0	0	0	0	1	1	6	1.791759	2.791759	0	0	0	0	0	2.791759
keluar	0	1	1	0	1	0	3	2	0.693147	1.693147	0	1.693147	1.693147	0	1.693147	0
kriteria	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
laku	0	0	0	0	1	0	1	6	1.791759	2.791759	0	0	0	0	2.791759	0
lama	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
lewat	0	0	0	0	1	0	1	6	1.791759	2.791759	0	0	0	0	2.791759	0
lokal	0	0	0	0	1	0	1	6	1.791759	2.791759	0	0	0	0	2.791759	0
masa	1	0	0	0	0	0	1	6	1.791759	2.791759	2.791759	0	0	0	0	0
masuk	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0
muka	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0
pandemi	1	0	0	0	0	0	1	6	1.791759	2.791759	2.791759	0	0	0	0	0
penuh	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
perlu	0	0	0	0	0	1	1	6	1.791759	2.791759	0	0	0	0	0	2.791759
prawira	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0
rekomendasi	1	1	1	0	1	1	5	1.2	0.182322	1.182322	1.182322	1.182322	1.182322	0	1.182322	1.182322
rumah	0	1	1	0	1	1	4	1.5	0.405465	1.405465	0	1.405465	1.405465	0	1.405465	1.405465
salah	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
satgas	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0
satu	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
sehat	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
sekolah	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0
situasi	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
tangan	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0
tatap	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0
tempat	0	0	0	0	1	0	1	6	1.791759	2.791759	0	0	0	0	2.791759	0
tim	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0

tinggal	0	0	0	0	1	0	1	6	1.791759	2.791759	0	0	0	0	2.791759	0
transmisi	0	0	0	0	1	0	1	6	1.791759	2.791759	0	0	0	0	2.791759	0
tular	1	0	0	0	1	0	2	3	1.098612	2.098612	2.098612	0	0	0	2.098612	0
ujar	0	0	0	1	0	0	1	6	1.791759	2.791759	0	0	0	2.791759	0	0
who	0	1	0	0	0	0	1	6	1.791759	2.791759	0	2.791759	0	0	0	0
yogi	0	0	0	1	0	1	2	3	1.098612	2.098612	0	0	0	2.098612	0	2.098612
<b>Total Bobot Dokumen</b>											31.48373	37.57807	11.04677	36.31204	36.88493	19.82752

Keterangan :

Term : Kata dasar yang diperoleh dari proses *text preprocessing*

TF : Frekuensi kemunculan term dalam dokumen

df : Dokumen frekuensi

D/df : Dokumen / frekuensi dokumen yang mengandung term

DF : Frekuensi dokumen yang mengandung term

IDF : hasil dari  $\log D/DF$

idf+1 : hasil dari  $\log D/DF + 1$

$W = tf * (idf+1)$  : Bobot dokume

#### 4. Singular Value Decomposition (SVD)

Proses selanjutnya setelah data di lakukan pembobotan TF-IDF adalah melakukan perhitungan *singular value decomposition* (SVD), perhitungan *svd* untuk memperkecil nilai kompleksitas dalam pemrosesan kata pada matriks. SVD melakukan proses penguraian (*dekomposisi*) suatu matriks menjadi tiga buah matriks baru, yaitu matriks *orthogonal* U, matriks *diagonal* S, dan *transpose* matriks *orthogonal* V (Saputra, Jerry. Fachrurrozi, 2017). Rumus SVD dapat di lihat pada rumus 1 berikut.

$$A = USV^t \quad (2.7)$$

Hasil Pembobotan TF-IDF membentuk matriks A yang akan di olah untuk mencari nilai singular value decomposition.

Tabel 2.14. Matriks A

	A					
	d1	d2	d3	d4	d5	d6
T1	2.36464	1.182322	1.182322	0	1.182322	1.182322
T2	0	0	0	0	2.791759	0
T3	0	0	0	2.791759	0	0
T4	0	0	0	0	2.791759	0
T5	0	2.791759	0	0	0	0
T6	2.791759	0	0	0	0	0
↓						
T54	0	0	0	2.098612	0	2.098612

Tabel 2.15. Matriks  $A^T$ 

	$A^T$						
	T1	T2	T3	T4	T5	T6	T54
d1	2.36464	0	0	0	0	2.791759	0
d2	1.182322	0	0	0	2.791759	0	0
d3	1.182322	0	0	0	0	0	0
d4	0	0	2.791759	0	0	0	2.098612
d5	1.182322	2.791759	0	2.791759	0	0	0
d6	1.182322	0	0	0	0	0	2.098612

Tabel 2.16. Hasil Perkalian matriks  $A^T A$ 

$A^T A$					
76.73472	10.57316	4.193653	6.379506	10.57316	4.193653
10.57316	98.85907	6.746433	1.975332	8.721765	6.746433
4.193653	6.746433	22.33427	0	6.746433	6.746433
6.379506	1.975332	0	96.51681	1.975332	4.404174
10.57316	8.721765	6.746433	1.975332	91.06515	6.746433
4.193653	6.746433	6.746433	4.404174	6.746433	42.32629

Proses selanjutnya setelah mendapatkan hasil perkalian dari matriks  $A^T A$  adalah mencari nilai *eigenvalue*. Temukan determinan sehingga  $|A^T A - \lambda I| = 0$ . Hasil determinan akan digunakan untuk mendapatkan *eigenvalue* yang selanjutnya untuk membentuk matriks S. Hasil perhitungan nilai *eigenvalue* ditunjukkan pada Tabel 2.17. Matriks S sebagai berikut.

Tabel 2.17. Matriks S (nilai eigenvalue)

Matriks S / Nilai Eigenvalue					
115.5021	0	0	0	0	0
0	95.92212	0	0	0	0
0	0	85.82928	0	0	0
0	0	0	69.13717	0	0
0	0	0	0	41.77809	0
0	0	0	0	0	19.66759

Setelah mendapatkan nilai eigenvalue dan membentuk matriks S, selanjutnya akan dilakukan inverse pada matriks S. Matriks S inverse

nantinya akan digunakan untuk perhitungan untuk mendapatkan nilai matriks orthogonal U. Hasil inverse matriks S dapat dilihat pada Tabel 2.18.

Matriks  $S^{-1}$  (*inverse matriks S*) sebagai berikut

Tabel 2.18. Matriks  $S^{-1}$  (inverse matriks S)

Matriks S inverse / S-1					
0.008658	0	0	0	0	0
0	0.010425	0	0	0	0
0	0	0.011651	0	0	0
0	0	0	0.014464	0	0
0	0	0	0	0.023936	0
0	0	0	0	0	0.050845

Proses selanjutnya setelah ditemukan nilai eigenvalue yaitu mencari nilai *eigenvector* dengan mengevaluasi  $|A^T A - \lambda_i I| = 0$ . Dari setiap *eigenvector* tersebut dilakukan normalisasi dengan membagi setiap nilai dari tiap *eigenvector* dengan panjang tiap vektor. Dari hasil normalisasi akan membentuk matriks V. Hasil perhitungan *eigenvector* dapat dilihat pada Tabel 2.19. perhitungan *eigenvector* sebagai berikut.

Tabel 2.19. Perhitungan *eigenvector*

Matriks V / Nilai Eigen Vektor					
-0.401214	-0.046844	0.118352214	0.904437183	0.060103	0.034736
-0.670581	0.328489	-0.620387227	-0.208363768	0.107695	0.050224
-0.115385	0.038772	0.026247473	7.97461E-05	-0.235914	-0.963763
-0.295196	-0.931957	-0.108875711	-0.168266587	0.061182	-0.020106
-0.512961	0.139943	0.766702418	-0.331333937	0.128386	0.05647
-0.160514	-0.016403	0.027147629	-0.021956615	-0.953367	0.252664

Setelah didapatkan nilai dari *eigenvector* dan membentuk matriks V, selanjutnya akan dilakukan transpose pada matriks V. Transpose matriks V nantinya akan digunakan sebagai perhitung pada metode CLSA. Hasil



transpose matriks  $V$  dapat dilihat pada Tabel 2.20. Matriks  $V^T$  sebagai berikut.

Tabel 2.20. Matriks  $V^T$ 

Matriks $V^T$					
-0.401214	-0.670581	-0.115384735	-0.295195937	-0.512961	-0.160514
0.046844	-0.328489	-0.038771824	0.931957242	-0.139943	0.016403
-0.118352	0.620387	-0.026247473	0.108875711	-0.766702	-0.027148
0.904437	-0.208364	7.97461E-05	-0.168266587	-0.331334	-0.021957
0.060103	0.107695	-0.235913993	0.061182038	0.128386	-0.953367
-0.034736	-0.050224	0.963762813	0.020105991	-0.05647	-0.252664

Setelah nilai *eigenvalue* yang ditunjukkan pada matriks  $S$  dan nilai *eigenvector* yang ditunjukkan pada matriks  $V$  telah diperoleh maka langkah selanjutnya adalah membentuk matriks *Ortogonal*  $U$  dengan menggunakan persamaan berikut:

$$U = AVS^{-1} \quad (2.8)$$

Perhitungan matriks *Ortogonal* akan bermanfaat untuk mencari vektor dari setiap dokumen. Perhitungan nilai ortogonal di mana hasilnya akan ditunjukkan pada Tabel 2.21. Matriks *orthogonal*  $U$  sebagai berikut.

Tabel 2.21. Matriks orthogonal  $U$ 

Matriks orthogonal $U$ ( $U = A*V*S^{-1}$ )					
-0.023153	0.004895	0.006011731	0.02133	-0.023574	-0.032158
-0.012399	0.004073	0.024938445	-0.013379	0.008579	0.008016
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.012399	0.004073	0.024938445	-0.013379	0.008579	0.008016
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.009698	-0.001363	0.003849629	0.036521	0.004016	0.004931
-0.00388	-0.000477	0.000883028	-0.000887	-0.063707	0.035865
-0.009698	-0.001363	0.003849629	0.036521	0.004016	0.004931
-0.022876	-0.007478	0.002551108	0.003994	0.012022	0.00867
-0.012399	0.004073	0.024938445	-0.013379	0.008579	0.008016
-0.00388	-0.000477	0.000883028	-0.000887	-0.063707	0.035865
-0.009698	-0.001363	0.003849629	0.036521	0.004016	0.004931

Matriks ortogonal U ( $U = A \cdot V \cdot S^{-1}$ )					
-0.002789	0.001128	0.000853749	3.22E-06	-0.015765	-0.136803
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.012399	0.004073	0.024938445	-0.013379	0.008579	0.008016
-0.012653	-0.021415	0.00023171	0.022346	0.006092	0.001561
-0.009698	-0.001363	0.003849629	0.036521	0.004016	0.004931
-0.019474	0.006162	-0.012275262	0.021129	0.008429	0.009066
-0.002789	0.001128	0.000853749	3.22E-06	-0.015765	-0.136803
-0.009698	-0.001363	0.003849629	0.036521	0.004016	0.004931
-0.00388	-0.000477	0.000883028	-0.000887	-0.063707	0.035865
-0.017759	0.007191	0.003270281	-0.011416	-0.032067	-0.043191
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.012399	0.004073	0.024938445	-0.013379	0.008579	0.008016
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.012399	0.004073	0.024938445	-0.013379	0.008579	0.008016
-0.012399	0.004073	0.024938445	-0.013379	0.008579	0.008016
-0.009698	-0.001363	0.003849629	0.036521	0.004016	0.004931
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.009698	-0.001363	0.003849629	0.036521	0.004016	0.004931
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.00388	-0.000477	0.000883028	-0.000887	-0.063707	0.035865
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.019046	0.005472	0.004381397	0.005863	-0.025275	-0.034246
-0.017759	0.007191	0.003270281	-0.011416	-0.032067	-0.043191
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.012399	0.004073	0.024938445	-0.013379	0.008579	0.008016
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.012399	0.004073	0.024938445	-0.013379	0.008579	0.008016
-0.012399	0.004073	0.024938445	-0.013379	0.008579	0.008016
-0.01661	0.002037	0.021640477	0.017396	0.009468	0.009732
-0.007135	-0.027124	-0.003541388	-0.006795	0.004088	-0.002854
-0.016208	0.00956	-0.020179267	-0.008414	0.007197	0.007129
-0.00828	-0.020749	-0.001998334	-0.005774	-0.044817	0.024815

Dari hasil perhitungan *Singular Value Decomposition* maka terbentuk matriks S, V, Dan U. Nilai dari matriks-matriks SVD nantinya akan digunakan sebagai perhitungan pada metode CLSA.

## 5. Latent Semantic Analysis (LSA)

Setelah didapatkan nilai dari tiap-tiap matriks Singular Value Decomposition (SVD). Selanjutnya menerapkan metode Latent Semantic Analysis, dimana metode ini melakukan perhitungan nilai length pada setiap matriks SVD dengan menggunakan rumus (2.5).

Dari hasil perhitungan nilai length tersebut maka didapatkan score dari tiap-tiap dokumen kalimat.

Tabel 2.22. hasil perhitungan LSA

LSA						
D1	46.34107	0	0	0	0	0
D2	0	31.50936	0	0	0	0
D3	0	0	2.252801623	0	0	0
D4	0	0	0	11.63347494	0	0
D5	0	0	0	0	5.363733	0
D6	0	0	0	0	0	4.969294

Dari tabel 2.22 di atas didapatkan score dari hasil perhitungan rumus (2.5). Score dari tiap dokumen tersebut akan dipilih dokumen dengan score tertinggi yang akan di panggil sebagai ringkasan.

## 6. Cross Latent Semantic Analysis (CLSA)

Berbeda dengan LSA pada proses CLSA, setelah didapatkan nilai dari tiap-tiap matriks Singular Value Decomposition (SVD). Selanjutnya menerapkan metode Cross Latent Semantic Analysis, dimana metode ini

pelakukan proses silang pada matriks SVD. Matriks  $V^T$  akan dilakukan seleksi dengan mencari rata-rata tiap dokumen kalimat. Apabila nilai dari dokumen kalimat lebih kecil dari nilai rata-rata tiap dokumen kalimat maka akan di rubah menjadi 0 dan membentuk matriks baru. Untuk lebih jelasnya matriks  $V^T$  CLSA bisa dilihat pada Tabel 2.23. Matriks  $V^T$  CLSA sebagai berikut.

Tabel 2.23. Matriks  $V^T$  CLSA

	Matriks $V^T$						Rata-rata
D1	-0.401214	-0.670581	-0.115384735	-0.295195937	-0.512961	-0.160514	-0.35931
D2	0.046844	-0.328489	-0.038771824	0.931957242	-0.139943	0.016403	0.081333
D3	-0.118352	0.620387	-0.026247473	0.108875711	-0.766702	-0.027148	-0.03486
D4	0.904437	-0.208364	7.97461E-05	-0.168266587	-0.331334	-0.021957	0.029099
D5	0.060103	0.107695	-0.235913993	0.061182038	0.128386	-0.953367	-0.13865
D6	-0.034736	-0.050224	0.963762813	0.020105991	-0.05647	-0.252664	0.098296

Pada tabel 2.23 di atas terdapat tiap-tiap dokumen dan hasil rata-rata tiap dokumen. Tahapan CLSA melakukan seleksi pada tiap-tiap kolom pada dokumen, dimana kolom dari tiap dokumen dengan nilai di bawah rata-rata maka akan di buat nol (0). Sedangkan nilai di atas rata-rata akan di tampilkan seperti yang di tampilkan pada tabel 2.24 sebagai berikut.

Tabel 2.24. Hasil seleksi matriks  $V^T$  CLSA

$V^T$ CLSA					
0	0	-0.115384735	-0.295195937	0	-0.160514
0	0	0	0.931957242	0	0
0	0.620387	-0.026247473	0.108875711	0	-0.027148
0.904437	0	0	0	0	0
0.060103	0.107695	0	0.061182038	0.128386	0
0	0	0.963762813	0	0	0

Setelah mendapatkan hasil dari matriks  $V^T$ , selanjutnya yaitu melakukan perhitungan nilai length pada matriks  $V^T$  CLSA dengan menggunakan rumus (2.5).

Selanjutnya menentukan hasil ringkasan berdasarkan skor tertinggi dari setiap dokumen Kalimat.

Tabel 2.25. Hasil ringkasan CLSA

CLSA						
D1	0	0	0	0	0	0
D2	0	0	0	0	0	0
D3	0	0	2.252801623	0	0	0
D4	0	0	0	0	0	0
D5	0	0	0	0	5.363733	0
D6	0	0	0	0	0	0

Dari tabel 2.25 di atas didapatkan score dari hasil perhitungan rumus (2.5). Score dari tiap dokumen tersebut akan dipilih dokumen dengan score tertinggi yang akan di panggil sebagai ringkasan.

## **BAB III**

### **METODE PENELITIAN**

#### **A. Objek Penelitian**

Objek pada penelitian ini adalah teks berita atau dokumen berita berbahasa Indonesia dari halaman [www.kompas.com](http://www.kompas.com), [www.detik.com](http://www.detik.com), [www.krjogja.com](http://www.krjogja.com) yang terdiri dari teks berita bahasa Indonesia berjumlah 300 dokumen berita. Data teks berita berbahasa Indonesia yang diambil adalah berita dengan rentang tahun 2019-2020.

#### **B. Metode Pengumpulan Data**

Proses pengumpulan data dilakukan dengan penacrian data dan informasi sebagai referensi yang dibutuhkan dalam melakukan penelitian. Beberapa metode yang dilakukan pada penelitian ini adalah sebagai berikut.

##### **1. Studi Pustaka**

Studi pustaka merupakan metode yang dilakukan untuk mendapatkan informasi serta referensi. Metode ini dilakukan dengan membaca serta menelaah penelitian-penelitian terdahulu, artikel, buku serta pustaka-digital yang terkait dengan penelitian yang hendak dilakukan.

##### **2. Teknik Dokumentasi**

Teknik dokumentasi adalah metode yang bertujuan untuk memperoleh *dataset* berupa dokumen berita yang dipublikasi dari *website* [www.kompas.com](http://www.kompas.com), [www.detik.com](http://www.detik.com), [www.krjogja.com](http://www.krjogja.com). Proses

pengumpulan data dilakukan dengan mengumpulkan berita berbahasa Indonesia berjumlah 300 berita yang didapatkan dari *website* [www.kompas.com](http://www.kompas.com), [www.detik.com](http://www.detik.com), [www.kriogja.com](http://www.kriogja.com) yang akan dijadikan sebagai *dataset* berita.

### C. Spesifikasi Kebutuhan

Spesifikasi kebutuhan pada penelitian ini terbagi menjadi kebutuhan perangkat keras (*hardware*) dan kebutuhan perangkat lunak (*software*). Berikut merupakan spesifikasi kebutuhan yang digunakan pada penelitian ini.

#### 1. Perangkat Keras (*Hardware*)

Spesifikasi perangkat keras (*hardware*) yang digunakan dalam penelitian ini yaitu *personal computer* dengan spesifikasi sebagai berikut.

- a. HP Notebook 14-am125TX
- b. *Processor* Intel Core(TM) i5-72000 2.7Ghz
- c. Memori RAM 4096MB
- d. *Hardisk* 500MB
- e. 64-bit Operating System

#### 2. Perangkat Lunak (*Software*)

Spesifikasi perangkat lunak yang digunakan pada penelitian ini yaitu sebagai berikut.

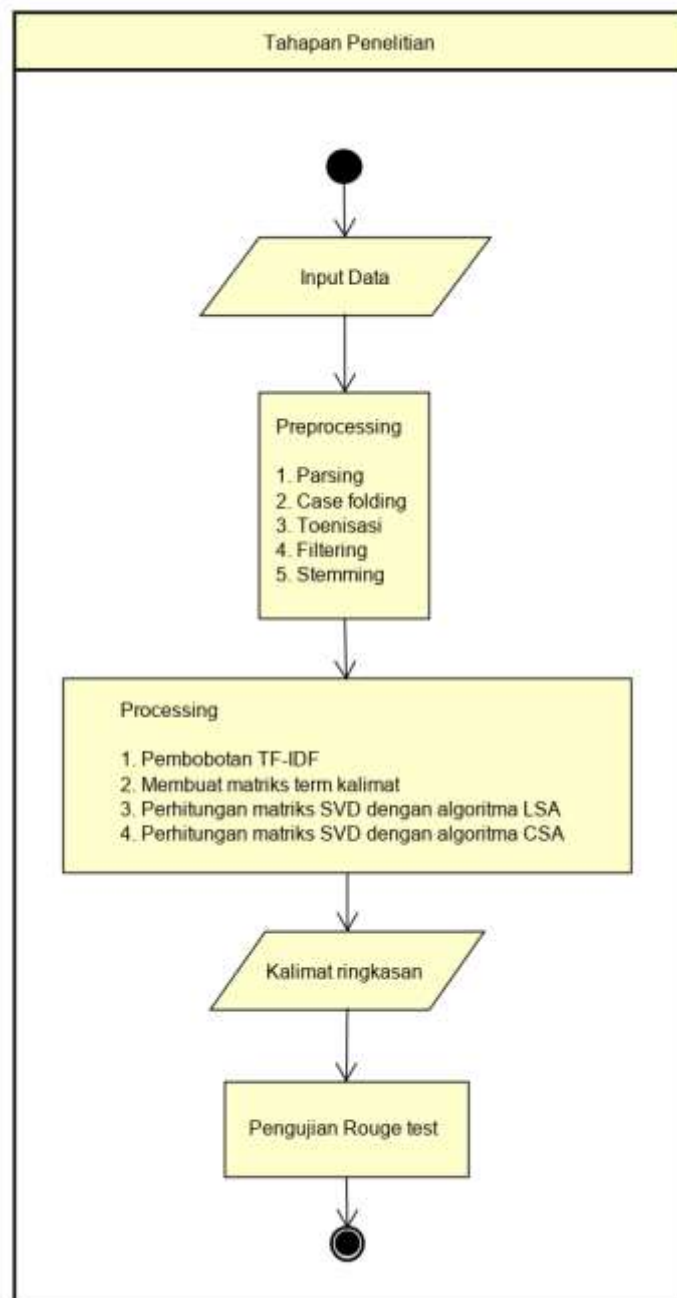
- a. Sistem Operasi Windows 10 pro 64-bit (10.0, Build 1734)
- b. Sublime Text Versi 3.1.1, build 3176
- c. Anaconda 3 dengan python 3.6

- d. Google Chrome Browser
- e. Microsoft Office 2016
- f. Library Python berupa :
  - 1) Sastrawi
  - 2) NLTK
  - 3) Sklearn
  - 4) Numpy
  - 5) Indosum
  - 6) Scypy
  - 7) Rouge
- g. Data 300 artikel berita dari halaman [www.kompas.com](http://www.kompas.com),  
[www.detik.com](http://www.detik.com), [www.krjogja.com](http://www.krjogja.com).

#### **D. Tahapan Penelitian**

Adapun tahapan-tahapan penelitaian yang dilakukan pada penelitian dapat dilihat pada gambar 3.1. Tahapan Penelitian sebagai berikut.





Gambar 3. 1 Tahapan Penelitian

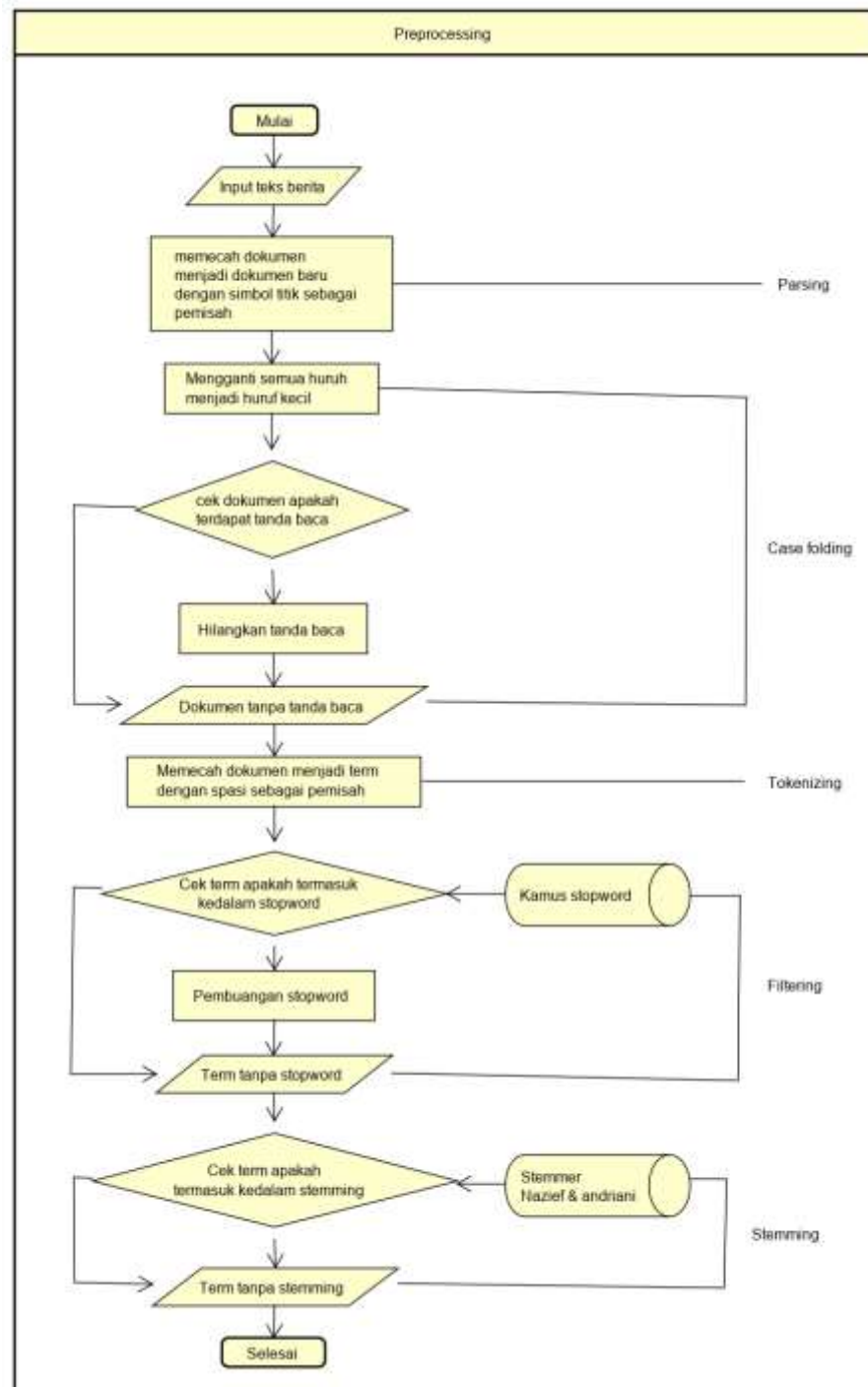
Adapun tahapan-tahapan penelitian dari gambar 3.1 diatas adalah sebagai berikut.

## 1. Input Data

Proses input data merupakan tahapan dimana dilakukan pengumpulan data berupa artikel berita untuk memperoleh informasi yang dilakukan dengan melakukan studi pustaka dalam rangka mencapai tujuan penelitian. Data yang dikumpulkan dalam penelitian ini merupakan data berita berbahasa Indonesia yang didapatkan dari *website* [www.kompas.com](http://www.kompas.com), [www.detik.com](http://www.detik.com), [www.krijogja.com](http://www.krijogja.com) dari rentang tahun 2018-2019. Data tersebut kemudian menjadi data uji atau sebagai data inputan.

## 2. *Preprocessing*

Proses *preprocessing* bertujuan untuk mengubah bentuk kalimat pada dokumen awal menjadi data kalimat dokumen yang terstruktur. Proses *preprocessing* terbagi menjadi beberapa tahapan seperti pada gambar 3.2. *Flowchart preprocessing* berikut.



Gambar 3. 2 Flowchart preprocessing

**a. *Parsing***

*Parsing* merupakan proses yang akan memecah setiap kalimat pada suatu dokumen dengan *symbol* titik sebagai pembatas kalimat. Setiap dokumen yang telah dipecah akan dimasukkan sebagai list kalimat.

**b. *Case folding***

*Case folding* merupakan tahapan yang mengubah semua huruf kapital dalam dokumen menjadi huruf kecil. Hanya huruf yang diterima yaitu 'a' sampai dengan huruf 'z'. Proses *case folding* hanya menghilangkan karakter seperti titik, koma, dan karakter-karakter sejenisnya yang dianggap delimiter akan dihilangkan.

**c. *Tokenizing***

*Tokenizing* adalah tahap pemotongan tiap kata yang menyusunnya, atau memisah tiap-tiap kata dalam satu dokumen yang dilakukan pada seluruh kalimat. Selain itu, spasi digunakan untuk memisahkan antar kata tersebut.

**d. *Filtering***

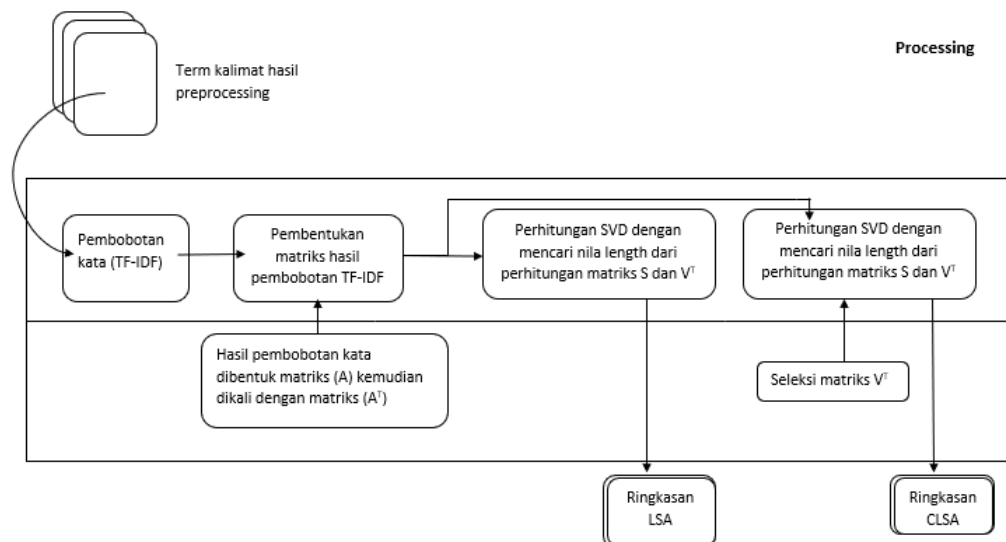
Tahap *filtering* adalah tahapan pemilihan kata dari hasil *tokenizing*. Proses *filtering* menggunakan algoritma *stopword* (menghilangkan kata yang tidak penting) atau *wordlist* (mengambil kata penting). Pada dokumen berbahasa Indonesia penggunaa *stopword* dalam tahapan *preprocessing* adalah menghilangkan kata seperti kata “yang”, “dan”, “di”, “dari” dan lain-lain.

### e. Stemming

*Stemming* merupakan tahapan untuk mencari *root* kata dari tiap kata hasil *filtering* atau proses untuk memecah kata menjadi kata dasar. Algoritma yang digunakan dalam pemecahan kata berbahasa Indonesia adalah algoritma Nazief & Adriani.

### 3. Processing

*Processing* adalah tahapan dimana dilakukan implementasi metode *cross latent semantic analysis* (CLSA). Melakukan pembobotan *tf-idf*, pembuatan *matriks term* dokumen dan melakukan perhitungan *singular value decomposition* (SVD) terhadap matriks inputan hasil pembobotan *tf-idf*. Adapun tahapan processing dapat dilihat pada gambar 3.3. Alur processing berikut.



Gambar 3.3. Alur Processing

**a. Pembobotan *Tf-Idf***

Pembobotan *Tf-Idf* adalah proses dimana data hasil *preprocessing* akan dilakukan pembobotan dengan melakukan perhitungan bobot tiap *term* pada suatu dokumen. Dokumen yang memuat *term* akan dihitung banyaknya *frekuensi* munculnya *term* tersebut dalam suatu dokumen.

**b. Pembuatan matriks *term* kalimat**

Hasil pembobotan *Tf-Idf* adalah sebagai *dataset* yang akan dilakukan untuk membentuk *matriks term* kalimat atau matriks  $A_{mn}$ . Matriks  $A_{mn}$  yang nantinya akan dilakukan perhitungan SVD untuk memperoleh kalimat ringkasan.

**c. Perhitungan matriks SVD**

Proses SVD adalah proses untuk menguraikan (*dekomposisi*) suatu matriks menjadi tiga buah matriks baru, yaitu matriks orthogonal  $U$ , matriks *diagonal*  $S$ , dan *transpose* matriks *orthogonal*  $V$  dari matriks *term* kalimat  $A_{mn}$  sebelumnya. Perhitungan dari matriks SVD menghasilkan nilai kalimat pada suatu dokumen yang akan digunakan sebagai acuan ringkasan.

**4. Kalimat ringkasan**

Penentuan hasil ringkasan dokumen diperoleh berdasarkan perhitungan matriks SVD dengan metode CLSA yang mengubah beberapa tahapan untuk menghasilkan ringkasan yang baik. Nilai skor tertinggi dari hasil perhitungan CLSA akan dijadikan sebagai kalimat ringkasan.

## 5. Pengujian

Proses pengujian kalimat hasil ringkasan menggunakan perbandingan antara dokumen ringasan hasil ringaks sistem dan dokumen ringakasan manual. Pengujian diukur dengan menggunakan metode *Rouge Test* yang menghasilkan nilai *precision*, nilai *recall* dan nilai *f-measure* sebagai parameter pengujian. Pengujian pada sistem ini akan menggunakan 3 tahapan rouge yaitu ROUGE-1 yang mengukur unigram, ROUGE-2 yang mengukur bigram dan ROUGE-L untuk membandingkan tiap kata dalam suatu kalimat.

## **BAB IV**

### **HASIL DAN PEMBAHASAN**

#### **A. Analisis Kebutuhan Sistem**

Pada tahap analisis sistem terdapat dua analisis kebutuhan untuk merancang sistem peringkasan teks berita, pemaparan kedua analisis kebutuhan tersebut sebagai berikut.

##### **1. Analisis Pengguna**

Tahapan analisis pengguna yaitu untuk mengetahui pengguna seperti apa yang bisa menggunakan sistem ini nantinya. Sistem ini diperuntukkan untuk semua jenis kalangan. Sistem ini bersifat umum serta tidak ada batasan pengguna untuk sistem ini.

##### **2. Analisis Kebutuhan Sistem**

Pada tahap analisis kebutuhan sistem terdapat 2 kebutuhan yaitu kebutuhan fungsional dan non fungsional, pemaparan kedua kebutuhan tersebut sebagai berikut.

###### **a. Kebutuhan Fungsional**

- 1) Sistem mampu menampilkan ringkasan berita sesuai dengan inputan berita asli.
- 2) Sistem mampu menampilkan batasan untuk kalimat ringkasan sesuai inputan user.

###### **b. Kebutuhan non-Fungsional**



- 1) Sistem bisa dijalankan di berbagai *web browser* seperti Google chrome dan Mozilla firefox.
- 2) Sistem memiliki tampilan atau *user interface* yang mudah di pahami serta di gunakan oleh pengguna.

## **B. Implementasi Sistem**

Sistem peringkasan teks berita otomatis ini menerapkan metode *Cross latent semantic analysis* (CLSA) sebagai metode utama. Dimana CLSA merupakan pengembangan dari metode terdahulu yaitu Latent semantic analysis (LSA). Pada implementasi sistem ini akan dijabarkan bagaimana metode CLSA diterapkan untuk membangun sistem peringkasan teks otomatis. Implementasi sistem ini terdiri dari berbagai tahapan sebagai berikut.

### **1. Input Data**

Input Data merupakan tahapan awal sistem dimana user akan diharuskan untuk memasukkan data berupa artikel berita yang nantinya artikel inputan akan diperoleh ringkasannya. Pada tahapan ini di gunakan contoh satu potongan artikel berita yang akan diolah untuk mendapatkan ringkasan.

Teks berita :

“Ikatan Dokter Anak Indonesia (IDAI) memberikan sejumlah rekomendasi untuk mencegah penularan Covid-19 kepada anak di masa pandemi. Salah satu yang direkomendasikan adalah agar anak tidak keluar rumah selama situasi Covid-19 di Indonesia belum memenuhi kriteria epidemiologi badan kesehatan dunia (WHO). Kami merekomendasikan agar anak-anak jangan

keluar rumah dulu. Termasuk untuk kegiatan tatap muka di sekolah, ujar anggota Tim Satgas Penanganan Covid-19 IDAI, Yogi Prawira. Rekomendasi untuk tidak keluar rumah ini, berlaku hingga daerah tempat tinggal anak-anak dianggap sudah dapat mengatasi penularan Covid-19 lewat transmisi lokal. Namun, kata Yogi, rekomendasi ini dikecualikan jika ada keperluan mendesak yang membuat anak untuk keluar rumah.”

Sc. Kompas.com

Teks diatas merupakan teks potongan berita yang diambil dari halaman [www.kompas.com](http://www.kompas.com) dengan judul berita “Rekomendasi IDAI agar Anak Terlindungi Saat Pandemi Covid-19”.

## 2. *Preprocessing*

Tahapan *Preprocessing* merupakan proses pengubahan dokumen awal menjadi data kalimat dokumen yang terstruktur, agar kemudian dokumen bisa di proses. Tahapan *preprocessing* terdiri dari tahapan *parsing* , *case folding*, *tokenisasi*, *filtering* dan *stemming*. Setelah teks berita yang akan diringkas sudah di inputkan *user*, selanjutnya akan dilakukan tahapan-tahapan *preprocessing*. Implementasi program untuk *preprocessing* dapat dilihat pada Listing 4.1. *Preprocessing* teks berikut.

<code>import pandas as pd</code>
<code>from nltk.corpus import stopwords</code>
<code>from nltk.stem import PorterStemmer</code>
<code>from Sastrawi.Stemmer.StemmerFactory import StemmerFactory</code>
<code>from Sastrawi.StopWordRemover.StopWordRemoverFactory</code>
<code>import StopWordRemoverFactory</code>
<code>from nltk.stem import PorterStemmer</code>

```

def splitParagraphIntoSentences(paragraph):
    ''' break a paragraph into sentences
        and return a list '''
    sentenceEnders = re.compile('[.!?]')
    sentenceList = sentenceEnders.split(paragraph)
    return sentenceList

stemmer = StemmerFactory().create_stemmer()
remover =
StopWordRemoverFactory().create_stop_word_remover()
translator = str.maketrans('', '', string.punctuation)

def stemmerEN(text):
    porter = PorterStemmer()
    factory = StopWordRemoverFactory()
    stopwords = factory.get_stop_words()
    text = text.lower()
    text = [i for i in text.lower().split() if i not in
stopwords]
    text = ' '.join(text)
    preprocessed_text = text.translate(translator)
    text_stem = porter.stem(preprocessed_text)
    return text_stem

def preprocessing(text):
    text = text.lower()
    text_clean = remover.remove(text)
    text_stem = stemmer.stem(text_clean)
    text_stem = stemmerEN(text_stem)
    print(text_stem)
    return text_stem

simpanisiberita = input()

sentences = splitParagraphIntoSentences(simpanisiberita)
list_isi_berita = []
berita_asli = []
for s in sentences:
    list_isi_berita.append(preprocessing(s.strip()))
    berita_asli.append(s.strip())
print(list_isi_berita)

```

Listing 4.1. Preprocessing Teks

Hasil *preprocessing* dari implementasi code pada listing 4.1 menghasilkan dokumen kalimat yang nantinya akan dilakukan perhitungan bobot kata pada tiap-tiap dokumen kalimat. Dokumen kalimat hasil *preprocessing* dapat dilihat pada tabel 4.1 berikut.

Tabel 4.1. Hasil *Preprocessing*

'ikat dokter anak indonesia ida beri jumlah rekomendasi cegah tular covid19 anak masa pandemi'
'salah satu rekomendasi anak keluar rumah lama situasi covid19 indonesia penuh kriteria epidemiologi badan sehat dunia who'
'rekomendasi anak jangan keluar rumah dulu', 'masuk giat tatap muka sekolah ujar anggota tim satgas tangan covid19 ida yogi prawira'
'rekomendasi keluar rumah laku hingga daerah tempat tinggal anak anggap atas tular covid19 lewat transmisi lok'
'kata yogi rekomendasi perlu desak buat anak keluar rumah'

### 3. *Processing*

*Processing* adalah tahapan dimana dokumen hasil *preprocessing* sudah menjadi dokumen yang terstruktur. Dokumen yang sudah terstruktur tersebut bisa diproses untuk dilakukan implementasi metode *cross latent semantic analysis* (CLSA). Adapun tahapan *processing* adalah sebagai berikut.

#### a. Perhitungan TF-IDF

Setelah melakukan tahapan-tahapan *preprocessing*, Langkah selanjutnya melakukan perhitungan bobot kata pada tiap dokumen kalimat. Hasil perhitungan bobot kata dapat dilihat pada tabel 4.1. Hasil Pembobotan TF-IDF.

Tabel 4.2. Hasil Pembobotan TF-IDF

anak	2.364643	1.182322	1.182322	0	1.182322	1.182322
anggap	0	0	0	0	2.791759	0
anggota	0	0	0	2.791759	0	0
atasi	0	0	0	0	2.791759	0
badan	0	2.791759	0	0	0	0
beri	2.791759	0	0	0	0	0
buat	0	0	0	0	0	2.791759
cegah	2.791759	0	0	0	0	0
covid19	1.405465	1.405465	0	1.405465	1.405465	0
Dst.						

Listing code yang untuk perhitungan tf-idf dapat dilihat pada listing 4.2.

Perhitungan TF-IDF sebagai berikut.

```

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import
TfidfTransformer, TfidfVectorizer
factory = StopWordRemoverFactory()
stopwords = factory.get_stop_words()
tfidf_vectorizer =
TfidfVectorizer(stop_words=stopwords, smooth_idf=False,
norm=None)
X = tfidf_vectorizer.fit_transform(list_isi_berita)
pd.DataFrame(X.toarray(),
columns=tfidf_vectorizer.get_feature_names())

```

Listing 4.2. Perhitungan TF-IDF

Pada listing 4.2. menjelaskan bagaimana proses perhitungan bobot kata menggunakan *library tfidfvectorizer* dari *sklearn*. Proses perhitungan menggunakan kalimat yang sebelumnya sudah di *preprocessing* yang disimpan di *list\_isi\_berita*.

#### b. Pembentukan Matriks term kalimat

Sebelum melakukan perhitungan SVD, hasil pembobotan kata akan dibentuk matriks terlebih dahulu agar kemudian bisa diolah untuk mendapatkan nilai matriks SVD. Matriks hasil pembobotan TF-IDF kita

sebut dengan matriks A, matriks A kemudian dilakukan transpose. Matriks A hasil pembobotan TF-IDF dan matriks A transpose kemudian di kalikan dan mendapatkan matriks baru kita sebut dengan matriks  $A \cdot A^T$ . Implementasi code untuk pengolahan matriks A dapat dilihat pada listing

#### 4.3. Perhitungan Matriks A.

```
import numpy
matrixAT=X.toarray()
matrixA=numpy.transpose(matrixAT)
a = np.array(matrixAT)
b = np.array(matrixA)
from numpy import *
A = array(matrixAT)
B = array(matrixA)
aTa = dot(A,B)
print ('[A]*[B]')
print (aTa)
```

Listing 4.3. Perhitungan Matriks A

Hasil perhitunga matriks A pada implementasi code pada listing 4.3 menghasilkan matriks baru yang kemudian akan diolah untuk mencari nilai SVD. Hasil perhitungan matriks A dapat dilihat pada tabel 4.3. Hasil Perhitungan Matriks A.

Tabel 4.3. Hasil Perhitungan Matriks A

[76.73472064 10.5731585 4.19365279 6.37950571 10.5731585 4.19365279]
[10.5731585 98.85906885 6.74643287 1.97533217 8.72176504 6.74643287]
[ 4.19365279 6.74643287 22.33427474 0. 6.74643287 6.74643287]
[ 6.37950571 1.97533217 0. 96.51680952 1.97533217 4.40417354]
[10.5731585 8.72176504 6.74643287 1.97533217 91.06514792 6.74643287]
[ 4.19365279 6.74643287 6.74643287 4.40417354 6.74643287 42.32629014]

### c. Perhitungan Singular Value Decomposition (SVD)

Setelah didapatkan matriks  $A \cdot A^T$ , selanjutnya melakukan perhitungan SVD yang akan menghasilkan matriks S, matriks V dan matriks U. Implementasi code untuk perhitungan SVD dapat dilihat pada listing

#### 4.4. Perhitungan SVD.

```
import scipy as sp
import numpy as np
A = np.array(aTa)
s, v = np.linalg.svd(A, full_matrices = False)
s = sp.diag(s)
```

Listing 4.4. Perhitungan SVD

Perhitungan SVD pada implementasi code pada listing 4.4 menggunakan *library* `numpy.linalg`. Hasil perhitungan SVD akan membentuk matriks matriks S, matriks V dan matriks U. Namun pada sistem ini hanya menggunakan matriks S dan matriks V yang akan olah untuk peringasan LSA dan CLSA. Hasil perhitungan SVD dapat dilihat pada tabel 4.4. Matriks SVD berikut.

Tabel 4.4 Matriks SVD

Singular values:				
[	115.50206633	0.	0.	0.
0.]				
[	0.	95.92212	0.	0.
0.]				
[	0.	0.	85.82927798	0.
0.]				
[	0.	0.	0.	69.13716588
0.]				
[	0.	0.	0.	0.
41.77809255				
0.]				
[	0.	0.	0.	0.
19.66758907]]				

Right singular vectors:
[[ -4.01214180e-01 -6.70580830e-01 -1.15384735e-01 - 2.95195937e-01
-5.12961364e-01 -1.60514463e-01]]
[ 4.68440161e-02 -3.28488964e-01 -3.87718238e-02 9.31957242e-01
-1.39942931e-01 1.64030135e-02]]
[[ -1.18352214e-01 6.20387227e-01 -2.62474726e-02 1.08875711e-01
-7.66702418e-01 -2.71476293e-02]]
[ 9.04437183e-01 -2.08363768e-01 7.97460864e-05 - 1.68266587e-01
-3.31333937e-01 -2.19566149e-02]]
[ 6.01029964e-02 1.07694606e-01 -2.35913993e-01 6.11820380e-02
1.28386267e-01 -9.53366569e-01]]
[[ -3.47359304e-02 -5.02240070e-02 9.63762813e-01 2.01059912e-02
-5.64695113e-02 -2.52664100e-01]]]

#### d. Latent Semantic Analysis (LSA)

Tahap selanjutnya yaitu pengimplementasian metode LSA, dimana metode ini menggunakan matriks S dan matriks V hasil perhitungan SVD. Tahapanya yaitu menghitung *length* pada setiap nilai matriks  $V^T$ . Implementasi code untuk metode LSA dapat dilihat pada listing 4.5. Skor LSA.

<code>loop = s[0]</code>
<code>for i in np.arange(np.size(loop)):</code>
<code>    for j in np.arange(np.size(loop)):</code>
<code>        print(np.sqrt(dot(np.square(s[i][j]), np.square(v[i][j]))))</code>
<code>        print('\n')</code>

Listing 4.5. Skor LSA

Hasil implementasi code pada listing 4.5 menghasilkan skor dari hasil perhitungan matriks kalimat. Nilai score kalimat untuk metode LSA bisa dilihat pada tabel 4.5. Skor LSA.



Tabel 4.5. Skor LSA

	0	1	2	3	4	5
0	46.341067	0.000000	0.000000	0.000000	0.000000	0.000000
1	0.000000	31.509358	0.000000	0.000000	0.000000	0.000000
2	0.000000	0.000000	2.252802	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000	11.633475	0.000000	0.000000
4	0.000000	0.000000	0.000000	0.000000	5.363733	0.000000
5	0.000000	0.000000	0.000000	0.000000	0.000000	4.969294

Hasil perhitungan matriks kalimat menghasilkan skor LSA. Kalimat ringkasan akan dipilih sesuai dengan skor tertinggi dari hasil perhitungan matriks kalimat, sesuai dengan berapa jumlah kalimat inputan user. Hasil ringkasan dengan contoh 3 kalimat inputan user menghasilkan ringkasan yang bisa dilihat pada tabel 4.6. Ringkasan LSA.

Tabel 4.6. Ringkasan LSA

Ikatan Dokter Anak Indonesia ( IDAI) memberikan sejumlah rekomendasi untuk mencegah penularan Covid 19 kepada anak di masa pandemi.
Salah satu yang direkomendasikan adalah agar anak tidak keluar rumah selama situasi Covid 19 di Indonesia belum memenuhi kriteria epidemiologi badan kesehatan dunia (WHO).
Termasuk untuk kegiatan tatap muka di sekolah," ujar anggota Tim Satgas Penanganan Covid 19 IDAI, Yogi Prawira.

#### e. Cross Latent Semantic Analysis (CLSA)

Berbeda dengan LSA tahapan CLSA melakukan seleksi untuk mencari nilai rata-rata dari matriks  $V^T$ . implementasi code untuk seleksi matriks  $V^T$  dapat dilihat pada listing 4.6 . Seleksi Matriks  $V^T$ .

aa =v[0]
for i in np.arange(np.size(aa)):
av = np.average(v[i])
for j in np.arange(np.size(aa)):
if v[i][j] < av :
v[i][j] = 0

```
pd.DataFrame(v)
```

Listing 4.6. Seleksi Matriks  $V^T$ 

Hasil implementasi code pada listing 4.6. menghasilkan matriks  $V^T$  yang baru, dimana nantinya akan dihitung nilai length pada setiap matriks  $V^T$ .

Hasil seleksi matriks  $V^T$  dapat dilihat pada tabel 4.7. Matriks  $V^T$  CLSA.

Tabel 4.7. Matriks  $V^T$  CLSA

	0	1	2	3	4	5
0	0.000000	0.000000	-0.115385	-0.295196	0.000000	-0.160514
1	0.000000	0.000000	0.000000	0.931957	0.000000	0.000000
2	0.000000	0.620387	-0.026247	0.108876	0.000000	-0.027148
3	0.904437	0.000000	0.000000	0.000000	0.000000	0.000000
4	0.060103	0.107695	0.000000	0.061182	0.128386	0.000000
5	0.000000	0.000000	0.963763	0.000000	0.000000	0.000000

Matriks  $V^T$  hasil seleksi akan dilakukan perhitungan nilai length pada setiap nilai matriks  $V^T$ . Hasil perhitungan nilai length untuk metode CLSA dapat dilihat pada tabel 4.8. Skor CLSA.

Tabel 4.8. Skor CLSA

	0	1	2	3	4	5
0	0.0	0.0	0.000000	0.0	0.000000	0.0
1	0.0	0.0	0.000000	0.0	0.000000	0.0
2	0.0	0.0	2.252802	0.0	0.000000	0.0
3	0.0	0.0	0.000000	0.0	0.000000	0.0
4	0.0	0.0	0.000000	0.0	5.363733	0.0
5	0.0	0.0	0.000000	0.0	0.000000	0.0

Kalimat ringkasan akan dipilih sesuai dengan skor tertinggi dari hasil perhitungan matriks kalimat, sesuai dengan berapa jumlah kalimat inputan user. Hasil ringkasan dengan contoh 3 kalimat inputan user menghasilkan ringkasan yang bisa dilihat pada tabel 4.9. Ringkasan CLSA.

Tabel 4.9. Ringkasan CLSA

Kami merekomendasikan agar anak anak jangan keluar rumah dulu
Rekomendasi untuk tidak keluar rumah ini, berlaku hingga daerah tempat tinggal anak anak dianggap sudah dapat mengatasi penularan Covid 19 lewat transmisi lokal

### C. Tampilan muka (*User interface*)

Tahapan ini merupakan tahapan dimana perancangan dan konstruksi untuk membangun sebuah sistem peringkasan dengan menggunakan bahasa pemrograman PHP dan *framework Flask*. Pada bagian *controller* bahasa pemrograman yang digunakan adalah python yang berekstensi py dan untuk tampilan menggunakan bahasa markup HTML. Tampilan-tampilan untuk sistem ini sebagai berikut.

#### 1. Tampilan Index

Tampilan ini merupakan tampilan utama sistem dimana terdapat input teks dan input jumlah kalimat ringkasan. User diharuskan untuk menginput teks dan jumlah ringkasan yang diinginkan. Untuk menampilkan hasil ringkasan user diarahkan untuk mengklik botton proses. Tampilan index dapat dilihat pada gambar 4.1. Tampilan Index.



Gambar 4.1. Tampilan Index

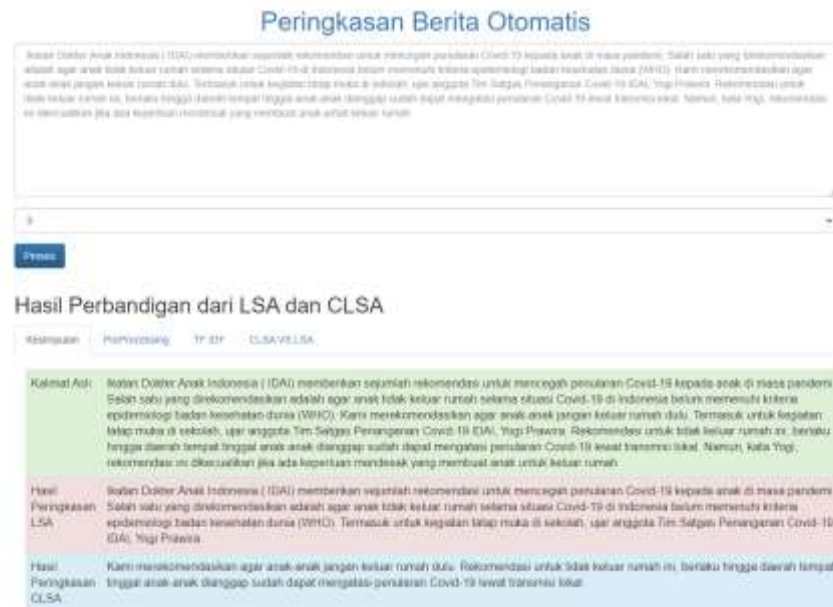
Tampilan pada gambar 4.1 dihasilkan dari fungsi `index` pada controller. Listing code untuk controller `index` dapat dilihat pada listing 4.6 berikut.

<code>@app.route('/')</code>
<code>def index():</code>
<code>    return render_template('index.html')</code>

Listing 4.7. Controller `index`

## 2. Tampilan Result

Tampilan Result merupakan tampilan dimana hasil proses inputan user. Disini terdapat kalimat inputan, hasil ringkasan LSA dan hasil ringkasan CLSA. Pada tampilan result juga terdapat menu seperti hasil *preprocessing*, TF-IDF dan matriks hasil perhitungan LSA dan CLSA dari proses inputan user. Tampilan result bisa dilihat pada gambar 4.2. Tampilan Result.



Gambar 4.2. Tampilan Result

Tampilan result pada gambar 4.2 dihasilkan melalui fungsi result atau proses pada controller. Listing code untuk controller result atau proses dapat dilihat pada listing 4.7 berikut.

```
@app.route('/proses', methods=['POST', 'GET'])
def proses():
    kalimat = str(request.form['kalimat'])
    params_angka = request.form['input_kalimat']

    pemisal_kalimat = splitParagraphIntoSentences(kalimat)
    simpan_sementara_isi_berita = list()
    berita_asli = list()
    for per_kalimat in pemisal_kalimat:

        simpan_sementara_isi_berita.append(preprocessing(per_kalimat.strip()))
        berita_asli.append(per_kalimat.strip())
    sesudah = ' '.join(simpan_sementara_isi_berita)

    factory = StopWordRemoverFactory()
    stopwords = factory.get_stop_words()
    tfidf_vectorizer =
    TfidfVectorizer(stop_words=stopwords, smooth_idf=False,
    norm=None)
    X =
    tfidf_vectorizer.fit_transform(simpan_sementara_isi_berita)
```

<code>return_TFIDF = pd.DataFrame(X.toarray(), columns=tfidf_vectorizer.get_feature_names()).T</code>
<code>return_LSA = pd.DataFrame(LSA(X)).T</code>
<code>return_CLSA = pd.DataFrame(CLSA(X)).T</code>
<code>rank_LSA = sum_frame_by_column(return_LSA, 'total_score_document', [i for i in range(len(return_CLSA[0]))])</code>
<code>rank_CLSA = sum_frame_by_column(return_CLSA, 'total_score_document', [i for i in range(len(return_CLSA[0]))])</code>
<code>docs = [str(x) for x in simpan_sementara_isi_berita]</code>
<code>documentNames = list()</code>
<code>for i, simpan_sementara_isi_berita in enumerate(docs):</code>
<code>documentNames.append("Document_{}".format(i+1))</code>
<code>return_LSA['documentNames'] = documentNames</code>
<code>return_LSA['rank'] = return_LSA['total_score_document'].rank(method='first', ascending=False).astype(int)</code>
<code>return_CLSA['documentNames'] = documentNames</code>
<code>return_CLSA['rank'] = return_CLSA['total_score_document'].rank(method='first', ascending=False).astype(int)</code>
<code>aftersort_LSA = rank_LSA.sort_values(['total_score_document'], ascending=False)</code>
<code>aftersort_LSA['rank'] = range(1, len(aftersort_LSA) + 1)</code>
<code>aftersort_CLSA = rank_CLSA.sort_values(['total_score_document'], ascending=False)</code>
<code>aftersort_CLSA['rank'] = range(1, len(aftersort_CLSA) + 1)</code>
<code>sentences_lsa = summary_sentence(berita_asli, X, params_angka, types='lsa')</code>
<code>sentences_clsa = summary_sentence(berita_asli, X, params_angka, types='clsa')</code>

Listing 4.8. Controller Result

### 3. Tampilan *preprocessing*

Tampilan *preprocessing* terdapat pada tabs menu pada tampilan *result*.

*Preprocessing* berisi tampilan pengubahan bentuk kalimat asli menjadi kalimat

yang lebih terstruktur untuk kemudian bisa dilakukan proses penerapan metode CLSA. Tampilan *preprocessing* dapat dilihat pada gambar 4.3 berikut.



Gambar 4.3. Tampilan tab menu preprocessing

Tampilan tabs menu *preprocessing* pada gambar 4.3 dihasilkan melalui fungsi *preprocessing* pada controller. Listing code untuk controller *preprocessing* dapat dilihat pada listing 4.8 berikut.

```
return render_template('proses.html'  
    sebelum_preprocessing=request.form['kalimat']  
    sesudah_preprocessing=sesudah  
)
```

Listing 4.9. Tab menu preprocessing

#### 4. Tampilan TF-IDF

Tampilan TF-IDF juga terdapat pada tabs menu pada tampilan result. TF-IDF berisi tampilan tabel skor tiap kalimat. Tampilan TF-IDF dapat dilihat pada gambar 4.4 berikut.

**Peringkasan Berita Otomatis**

Berita Online yang Indonesia (2016) menampilkan informasi perkembangan terkini mengenai pandemi Covid-19 kepada user di masa pandemi. User yang menggunakan aplikasi agar dapat lebih mudah akses berita Covid-19 di Indonesia. Berita-berita yang disajikan pada aplikasi ini adalah berita-berita yang relevan dengan topik yang sedang dibahas. Untuk menghasilkan berita-berita yang relevan, aplikasi ini menggunakan algoritma TF-IDF. TF-IDF adalah algoritma yang digunakan untuk mengukur relevansi kata dalam dokumen. TF-IDF adalah algoritma yang digunakan untuk mengukur relevansi kata dalam dokumen. TF-IDF adalah algoritma yang digunakan untuk mengukur relevansi kata dalam dokumen.

3

**Proses**

**Hasil Perbandingan dari LSA dan CLSA**

Result: TF-IDF TF-IDF TF-IDF TF-IDF TF-IDF TF-IDF

	0	1	2	3	4	5
0	1.364041	1.181112	1.452322	0.000000	1.181112	1.181112
1	0.000000	0.000000	0.000000	0.000000	1.791759	0.000000
2	0.000000	0.000000	0.000000	2.791759	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000	1.791759	0.000000
4	0.000000	1.791759	0.000000	0.000000	0.000000	0.000000
5	1.791759	0.000000	0.000000	0.000000	0.000000	0.000000
6	0.000000	0.000000	0.000000	0.000000	0.000000	2.791759
7	1.791759	0.000000	0.000000	0.000000	0.000000	0.000000
8	1.452322	1.452322	0.000000	1.452322	1.452322	0.000000
9	0.000000	0.000000	0.000000	0.000000	1.791759	0.000000
10	0.000000	0.000000	0.000000	0.000000	0.000000	2.791759
11	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
12	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
13	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
14	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
15	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
16	0.000000	1.791759	0.000000	0.000000	0.000000	0.000000
17	0.000000	1.791759	0.000000	0.000000	0.000000	0.000000
18	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
19	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
20	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
21	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
22	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
23	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
24	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
26	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
27	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
28	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
29	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
30	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
31	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
32	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
33	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
34	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
35	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
36	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
37	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
38	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
39	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
40	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
41	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
42	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
43	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
44	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
45	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
46	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
47	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
48	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
49	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
51	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
52	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
53	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
54	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
55	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
56	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
57	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
58	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
59	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
60	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
61	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
62	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
63	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
64	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
65	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
66	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
67	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
68	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
69	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
70	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
71	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
72	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
73	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
74	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
76	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
77	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
78	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
79	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
80	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
81	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
82	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
83	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
84	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
85	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
86	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
87	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
88	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
89	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
90	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
91	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
92	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
93	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
94	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
95	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
96	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
97	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
98	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
99	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Gambar. 4.4. Tampilan tab menu TF-IDF

Tampilan tabs menu TF-IDF pada gambar 4.4 dihasilkan melalui fungsi TF-IDF pada controller. Listing code untuk controller TF-IDF dapat dilihat pada listing 4.9 berikut.

```
return render_template('proses.html'
    tables_TFIDF=[return_TFIDF.to_html()])
```

Listing 4.10. Tab menu TF-IDF

## 5. Tampilan LSA VS CLSA

Tampilan LSA VS CLSA juga terdapat pada tabs menu pada tampilan result. LSA VS CLSA berisi tampilan tabel skor matriks dari proses ke-dua algoritma LSA dan CLSA. Matriks tersebut berisi skor tiap dokumen kalimat yang kemudian dirutkan sesuai dengan skor tertinggi tiap dokumen kalimat. Skor tertinggi tiap kalimat tersebut kemudian akan menjadi ringkasan sesuai dengan berapa banya inputan ringkasan yang diinginkan user. Tampilan LSA VS CLSA dapat dilihat pada gambar 4.5 berikut.



[illegible]

Gambar. 4.5. Tampilan tab menu LSA VS CLSA

Tampilan tabs menu LSA VS CLSA pada gambar 4.5 dihasilkan melalui fungsi LSA VS CLSA pada controller. Listing code untuk controller LSA VS CLSA dapat dilihat pada listing 4.10 sebagai berikut.

```

return render_template('proses.html',
    sum_tables_LSA = [aftersort_LSA.to_html()],
    sum_tables_CLSA = [aftersort_CLSA.to_html()],
    sentences_lsa = sentences_lsa,
    sentences_clsa = sentences_clsa)

```

Listing 4.11. Tab menu LSA VS CLSA

#### D. Pengujian Sistem

Pada penelitian ini proses pengujian sistem dilakukan dengan menggunakan *library Rouge*. Proses pengujian dilakukan dengan membandingkan hasil ringkasan sistem dan hasil ringkasan manusia. Proses pengujian penelitian ini menggunakan 10 dokumen berita. Setiap berita diringkas oleh 5 orang yang berbeda. Ringkasan manual di buat dengan memberi batasan seperti tidak membuat kalimat baru, hanya meringkas dengan mengambil kalimat atau kata penting yang terdapat pada dokumen asli. Hasil ringkasan dari setiap orang yang berbeda untuk kemudian dibandingkan dengan ringkasan sistem.

Contoh salah satu berita yang gunakan dalam pengujian ini dapat dilihat pada tabel 4.10. Sampel Pengujian.

Tabel 4.10. Sampel Pengujian

Dokumen Asli
Berita
KARANGANYAR, KRJOGJA.com – Larangan perusakan hutan Lawu diikuti sanksi hukuman berat bagi pelanggarnya. Kerusakan itu seperti pembalakan, aktivitas memicu kebakaran sampai merusak jalur pendakian dengan menaiki kendaraan bermotor. Administratur Muda Perum Perhutani KPH Surakarta, Sugi Purwanta mengatakan penegakan hukum di hutan Lawu telah disepakati bersama. Polres Karanganyar selaku perangkat penindakan siap menjalankan tugasnya. “Kita semua saling mengingatkan untuk menjaga gunung Lawu dari kerusakan dan perusakan,” jelasnya di sela apel siaga antisipasi kebakaran hutan di bukit Mongkrang, Tawangmangu, Senin (31/8). Perum Perhutani KPH Surakarta mengumumkan orang yang nekat membakar hutan akan diancam hukuman penjara maksimal 15 tahun dan denda maksimal Rp5 miliar. Perhutani memasang spanduk berisi informasi itu di beberapa tempat di area hutan Gunung Lawu. Dasar hukum yang digunakan adalah Undang-Undang (UU) No.41/1999 tentang Kehutanan. Ancaman hukuman dan denda itu untuk pelaku pembakaran hutan. Perum Perhutani KPH Surakarta gencar melakukan tindakan pencegahan kebakaran hutan yang sering terjadi di musim kemarau. Salah satu bentuk pencegahan dengan melarang pembuatan arang tradisional. Apel siaga tersebut diikuti jajaran TNI, Polri, pemerintah kecamatan, BPBD, serta para sukarelawan. Sugi Purwanta menyampaikan, seluruh komponen di apel siaga memiliki komitmen sama untuk menjaga kelestarian Gunung Lawu dari kebakaran akibat faktor alam maupun kesengajaan. Hutan Lawu patut dilestarikan karena selain

mengkonservasi air, juga menyuguhkan pemandangan indah untuk pariwisata. Pemanfaatannya sekarang mutlak mengikuti aturan. Asper Perhutani KPH Lawu Utara, Widodo mengaku melibatkan masyarakat desa hutan untuk mengawasi. “Hanya polisi hutan saja tidak cukup. Kami mengajak masyarakat desa hutan ikut mengawasi. Supaya hutan dan lahan tetap lestari,” katanya. Belum lama ini, komunitas offroad nekat menjelajah dengan motor trail dan jip tanpa izin perhutani. Akibatnya, jalur pendakian rusak. “Sementara ini mereka belum mengajukan izin. Mereka tidak boleh melakukan apa pun di hutan sebelum ada izin. Saya konfirmasi ke mereka katanya survei. Mau buka wisata jip. Mereka di petak 7 RPH Nglerak BKPH Lawu Utara. Apa pun kegiatan di kawasan hutan mau survei atau lainnya harus izin dulu secara tertulis,” ungkap dia. (Lim)

Dokumen berita asli pada tabel 4.10 kemudian dilakukan peringkasan sistem dan peringkasan yang dibuat oleh manusia. Hasil ringkasan sistem menggunakan *compression rate* 50% dari teks dokumen asli. Perbandingan ringkasan sistem dan ringkasan manual dapat dilihat pada tabel 4.11. Perbandingan ringkasan.

Tabel 4.11. Perbandingan Ringkasan

Ringkasan Sistem	Ringkasan Manual
<p>Larangan perusakan hutan Lawu diikuti sanksi hukuman berat bagi pelanggarnya.</p> <p>“Kita semua saling mengingatkan untuk menjaga gunung Lawu dari kerusakan dan perusakan,” jelasnya di sela apel siagaantisipasi kebakaran hutan di bukit Mongkrang, Tawangmangu, Senin (31/8).</p> <p>Perum Perhutani KPH Surakarta mengumumkan orang yang nekat membakar hutan akan diancam hukuman penjara maksimal 15 tahun dan denda maksimal Rp5 miliar.</p> <p>Perhutani memasang spanduk berisi informasi itu di beberapa tempat di area hutan Gunung Lawu.</p> <p>Dasar hukum yang digunakan adalah Undang-Undang (UU) No. 41/1999 tentang Kehutanan.</p>	<p>Larangan perusakan hutan Lawu diikuti sanksi hukuman berat bagi pelanggarnya.</p> <p>Kerusakan itu seperti pembalakan, aktivitas memicu kebakaran sampai merusak jalur pendakian dengan menaiki kendaraan bermotor.</p> <p>Perum Perhutani KPH Surakarta mengumumkan orang yang nekat membakar hutan akan diancam hukuman penjara maksimal 15 tahun dan denda maksimal Rp5 miliar.</p> <p>Perum Perhutani KPH Surakarta gencar melakukan tindakan pencegahan kebakaran hutan yang sering terjadi di musim kemarau.</p> <p>Perhutani memasang spanduk berisi informasi itu di beberapa tempat di area hutan Gunung Lawu.</p>

<p>Salah satu bentuk pencegahan dengan melarang pembuatan arang tradisional.</p> <p>Apel siaga tersebut diikuti jajaran TNI, Polri, pemerintah kecamatan, BPBD, serta para sukarelawan.</p> <p>Hutan Lawu patut dilestarikan karena selain mengkonservasi air, juga menyuguhkan pemandangan indah untuk pariwisata.</p> <p>Asper Perhutani KPH Lawu Utara, Widodo mengaku melibatkan masyarakat desa hutan untuk mengawasi.</p> <p>Kami mengajak masyarakat desa hutan ikut mengawasi.</p> <p>Belum lama ini, komunitas offroad nekat menjelajah dengan motor trail dan jip tanpa izin perhutani.</p> <p>Saya konfirmasi ke mereka katanya survei.</p> <p>Mereka di petak 7 RPH Nglerak BKPH Lawu Utara.</p>	<p>Salah satu bentuk pencegahan dengan melarang pembuatan arang tradisional.</p> <p>Hutan Lawu patut dilestarikan karena selain mengkonservasi air, juga menyuguhkan pemandangan indah untuk pariwisata.</p> <p>Belum lama ini, komunitas offroad nekat menjelajah dengan motor trail dan jip tanpa izin perhutani.</p> <p>Apa pun kegiatan di kawasan hutan mau survei atau lainnya harus izin dulu secara tertulis.</p>
--	---

Hasil ringkasan sistem dan hasil ringkasan manual kemudian dilakukan pengujian dengan menggunakan menggunakan tahapan ROUGE-N dan ROUGE-L.

ROUGE-N pada penelitian ini adalah perhitungan *recall* berdasarkan pada perbandingan *n-gram* antara peringkasan manual dan teks hasil peringkasan mesin. Jumlah *n-gram* yang digunakan  $n=1$  (ROUGE-1) dan  $n=2$  (ROUGE-2). Misal  $p$  adalah jumlah *n-gram* yang sama antara peringkasan manual dan teks hasil peringkasan mesin, dan  $q$  adalah jumlah *n-gram* pada peringkasan manual. ROUGE-N dapat dihitung dengan rumus sebagai berikut:

$$ROUGE - N = \frac{p}{q} \quad (4.1)$$

ROUGE-L pada penelitian ini mengevaluasi ringkasan teks dengan cara membandingkan *longest common subsequence* (LCS) atau rangkaian kata terpanjang yang sama antara hasil ringkasan teks mesin dan peringkasan manual. Misal  $m$  adalah jumlah kata pada peringkasan manual, ROUGE-L dapat dihitung dengan rumus sebagai berikut:

$$ROUGE - L = \frac{LCS}{m} \quad (4.2)$$

Hasil pengujian pada penelitian ini didapatkan rata-rata *f-measure* 0.6533, *precision* 0.6727, *recall* 0.6572. Hasil pengujian dapat dilihat pada tabel 4.12. Hasil Pengujian.

Tabel 4.12. Hasil Pengujian

	F-Measure	Precision	Recall
Rouge-1	0.6741	0.6996	0.674
Rouge-2	0.592	0.6148	0.5919
Rouge-L	0.6938	0.7039	0.7057
Rata-Rata	0.6533	0.6727667	0.6572

Penjabaran hasil pengujian pada penelitian ini dapat dilihat pada tabel 4.13.

Penjabaran Hasil Pengujian.

Tabel 4.13. Penjabaran Hasil Pengujian

Ringkasan Sistem	Ringkasan Manual	Rouge-1			Rouge-2			Rouge-L		
		F-Measure	Precision	Recall	F-Measure	Precision	Recall	F-Measure	Precision	Recall
D1	M1	0.6765	0.685	0.6682	0.5558	0.5628	0.549	0.6818	0.6907	0.673
	M2	0.7042	0.75	0.6637	0.6603	0.7035	0.6222	0.7398	0.7763	0.7065
	M3	0.788	0.79	0.786	0.7318	0.7336	0.73	0.7999	0.7894	0.8108
	M4	0.6907	0.765	0.6296	0.6122	0.6783	0.5578	0.7012	0.7565	0.6534
	M5	0.6754	0.64	0.715	0.583	0.5527	0.6179	0.6619	0.6118	0.7209
D2	M1	0.7303	0.8552	0.6372	0.6931	0.8133	0.6039	0.7575	0.8474	0.6849
	M2	0.6358	0.8157	0.521	0.5492	0.7066	0.4491	0.6447	0.8305	0.5268
	M3	0.7301	0.9078	0.6106	0.6844	0.8533	0.5714	0.7591	0.8813	0.6666
	M4	0.6572	0.921	0.5109	0.5971	0.84	0.4632	0.6967	0.9152	0.5625
	M5	0.7484	0.8026	0.7011	0.708	0.76	0.6627	0.7666	0.7796	0.754
D3	M1	0.5622	0.8641	0.4166	0.421	0.65	0.3113	0.6242	0.8709	0.4864
	M2	0.7931	0.8518	0.7419	0.7093	0.7625	0.663	0.8387	0.8387	0.8387
	M3	0.8156	0.9012	0.7448	0.7796	0.8625	0.7113	0.8749	0.9032	0.8484
	M4	0.8287	0.9259	0.75	0.7597	0.85	0.6868	0.9007	0.9516	0.855
	M5	0.8144	0.9753	0.6991	0.7812	0.9375	0.6696	0.8714	0.9838	0.782
D4	M1	0.6666	0.6618	0.6715	0.562	0.5579	0.5661	0.6929	0.6869	0.6991
	M2	0.6594	0.6546	0.6642	0.5912	0.5869	0.5955	0.6666	0.6521	0.6818
	M3	0.702	0.6187	0.8113	0.6419	0.565	0.7428	0.6965	0.6086	0.8139
	M4	0.6801	0.7266	0.6392	0.583	0.6231	0.5477	0.6887	0.7217	0.6587
	M5	0.6666	0.5827	0.7788	0.6473	0.5652	0.7572	0.7211	0.6521	0.8064
D5	M1	0.655	0.6097	0.7075	0.5462	0.5081	0.5904	0.6666	0.6063	0.7402
	M2	0.6666	0.7642	0.5911	0.6071	0.6967	0.5379	0.6868	0.7234	0.6538
	M3	0.7111	0.6504	0.7843	0.6188	0.5655	0.6831	0.7052	0.6489	0.7721
	M4	0.655	0.6097	0.7075	0.5462	0.5081	0.5904	0.6627	0.6063	0.7307
	M5	0.7364	0.7154	0.7586	0.6835	0.6639	0.7043	0.7624	0.734	0.7931
D6	M1	0.5592	0.7662	0.4402	0.4976	0.6842	0.3909	0.5764	0.7656	0.4622

	M2	0.4897	0.6233	0.4033	0.4123	0.5263	0.3389	0.4968	0.625	0.4123
	M3	0.5816	0.7402	0.4789	0.5154	0.6578	0.4237	0.5925	0.75	0.4897
	M4	0.6178	0.7662	0.5175	0.5185	0.6447	0.4336	0.6086	0.7656	0.5051
	M5	0.7204	0.7532	0.6904	0.6415	0.671	0.6144	0.7092	0.7812	0.6493
D7	M1	0.5675	0.497	0.6614	0.4761	0.4166	0.5555	0.5843	0.5144	0.6761
	M2	0.5391	0.5088	0.5733	0.4227	0.3988	0.4496	0.5254	0.4855	0.5726
	M3	0.622	0.5502	0.7153	0.5387	0.4761	0.6201	0.6172	0.5434	0.7142
	M4	0.7134	0.6923	0.7358	0.638	0.619	0.6582	0.7218	0.6956	0.75
	M5	0.6517	0.6035	0.7083	0.553	0.5119	0.6013	0.6536	0.6086	0.7058
D8	M1	0.479	0.4565	0.504	0.3448	0.3284	0.3629	0.4729	0.4137	0.5517
	M2	0.7341	0.6304	0.8787	0.6893	0.5912	0.8265	0.7462	0.6465	0.8823
	M3	0.8207	0.7463	0.9115	0.7469	0.6788	0.8303	0.8151	0.7413	0.9052
	M4	0.7086	0.7753	0.6524	0.6399	0.7007	0.5889	0.7744	0.7844	0.7647
	M5	0.7285	0.7391	0.7183	0.6402	0.6496	0.6312	0.7312	0.7155	0.7477
D9	M1	0.6104	0.5588	0.6725	0.4939	0.4518	0.5446	0.6818	0.6122	0.7692
	M2	0.7443	0.7279	0.7615	0.6666	0.6518	0.6821	0.7567	0.7142	0.8045
	M3	0.6008	0.5367	0.6822	0.4896	0.437	0.5566	0.6555	0.602	0.7195
	M4	0.7188	0.7426	0.6965	0.6236	0.6444	0.6041	0.7403	0.7857	0.7
	M5	0.7172	0.625	0.8415	0.6127	0.5333	0.72	0.7441	0.653	0.8648
D10	M1	0.5857	0.5815	0.5899	0.41	0.4071	0.413	0.599	0.5963	0.6018
	M2	0.5963	0.4609	0.8441	0.537	0.4142	0.7631	0.6071	0.4678	0.8644
	M3	0.7029	0.5957	0.8571	0.6497	0.55	0.7938	0.7292	0.6055	0.9166
	M4	0.6343	0.695	0.5833	0.4429	0.4857	0.4071	0.6434	0.6788	0.6115
	M5	0.6153	0.5673	0.6722	0.5503	0.5071	0.6016	0.6428	0.5779	0.7241
Rata-Rata		0.674174	0.699686	0.673996	0.592082	0.61489	0.59193	0.693882	0.703938	0.7057

## BAB V

### KESIMPULAN DAN SARAN

#### A. Kesimpulan

Setelah melakukan analisis, perancangan, implementasi dan pengujian pada peringkasan teks berita otomatis menggunakan metode *Cross Latent Semantic Analysis*, maka dapat ditarik kesimpulan dari penelitian ini adalah sebagai berikut :

1. Sistem dapat mempermudah user dalam mencari intisari dari sebuah artikel berita tanpa harus membaca keseluruhan isi berita.
2. Sistem dapat melakukan peringkasan teks berita dengan hasil ringkasan menggunakan algoritma LSA dan CLSA.
3. Sistem memperoleh hasil pengujian dengan menggunakan 10 artikel berita yang dibandingkan dengan ringkasan sistem. Hasil ringkasan memperoleh nilai rata-rata Rouge-1 (*f-measure* 0,5811, *precision* 0,5757, *recall* 0,6194), Rouge-2 (*f-measure* 0,47022, *precision* 0,4673, *recall* 0,5005), Rouge-L (*f-measure* 0,5936, *precision* 0,5809, *recall* 0,6332).

#### B. Saran

Penelitian ini masih memiliki beberapa kekurangan, maka di harapkan adanya pengembangan untuk penelitian ini. Saran yang dapat diberikan untuk pengembangan penelitian ini sebagai berikut :

1. Pengembangan dapat dilakukan menambahkan kesamaan similarity antara judul berita dan dokumen kalimat yang dipilih oleh CLSA.



2. Tahapan pemecahan kalimat berpengaruh terhadap hasil ringkasan, untuk itu pengembangan dapat dilakukan dengan melakukan tahapan preprocessing yang lebih kompleks.
3. Perlu adanya data pakar sebagai pembanding hasil ringkasan sistem untuk memperoleh skor pegujian yang lebih akurat.

## DAFTAR PUSTAKA

- Feldman & Sanger. (2007). 済無No Title No Title. In *Journal of Chemical Information and Modeling* (Vol. 53).  
<https://doi.org/10.1017/CBO9781107415324.004>
- Indrawati, N. (2010). *NATURAL LANGUAGE PROCESSING ( NLP ) BAHASA INDONESIA adalah : (1)*.
- Irawan, S. (2017). Studi Awal Peringkasan Dokumen Bahasa Indonesia Menggunakan Metode Latent Semantik Analysis dan Maximum Marginal Relevance. *Annual Research Seminar (ARS)*, 2(1), 235–239.
- Klein, G., Hirschman, L., Firmin, T., & Diego, S. (1998). Inderjeet Mani David House Text Summarization Evaluation Code D44208 53140 Gatchell Rd . USA. *Methods*, 77–85.
- Mandar, G., & Gunawan, G. (2017). Peringkasan dokumen berita Bahasa Indonesia menggunakan metode Cross Latent Semantic Analysis. *Register: Jurnal Ilmiah Teknologi Sistem Informasi*, 3(2), 94.  
<https://doi.org/10.26594/register.v3i2.1161>
- Pustejovsky, J., & Stubbs, A. (n.d.). 1449306667 {59185621} *Natural Language Annotation for Machine Learning\_ A Guide to Corpus-... [Pustejovsky \_ Stubbs 2012-11-04].pdf*.
- Saputra, Jerry. Fachrurrozi, M. Y. (2017). Peringkasan Teks Berita Berbahasa Indonesia Menggunakan Metode Latent Semantic Analysis (LSA) dan Teknik Steinberger & Jezek. *Computer Science and ICT*, 3(1), 215–219.
- Savanti, N., Gotami, W., & Dewi, R. K. (2018). Peringkasan Teks Otomatis Secara Ekstraktif Pada Artikel Berita Kesehatan Berbahasa Indonesia Dengan Menggunakan Metode Latent Semantic Analysis. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya*, 2(9), 2821–2828.
- Steinberger, J. (2004). *Using Latent Semantic Analysis in Text Summarization and Summary Evaluation*. 1–8.
- Widyasanti, N. K., Gede, I. K., Putra, D., Kadek, N., & Rusjyanthi, D. (2018). Seleksi Fitur Bobot Kata dengan Metode TFIDF untuk Ringkasan Bahasa Indonesia. *Merpati*, 6(2), 119–126.
- Winata, F., Rainarli, E., Informatika, T., & Indonesia, U. K. (2016). Implementasi

cross method latent semantic analysis untuk meringkas dokumen berita berbahasa indonesia 1,2. *Techno.COM*, 15(4), 266–277.

Yuliska, Y., & Syaliman, K. U. (2020). Literatur Review Terhadap Metode, Aplikasi dan Dataset Peringkasan Dokumen Teks Otomatis untuk Teks Berbahasa Indonesia. *IT Journal Research and Development*, 5(1), 19–31.  
[https://doi.org/10.25299/itjrd.2020.vol5\(1\).4688](https://doi.org/10.25299/itjrd.2020.vol5(1).4688)