PENGARUH PERINGKAS DOKUMEN OTOMATIS DENGAN PENGGABUNGAN METODE FITUR DAN *LATENT SEMANTIC ANALYSIS* (LSA) PADA PROSES *CLUSTERING* DOKUMEN TEKS BERBAHASA INDONESIA

Muhammad Jamhari[1], Edi Noersasongko[2], Hendro Subagyo[3]

[1,2,3] *Pascasarjana Magister, Teknik Informatika, Udinus*

[1]Muhammad_Jamhari@yahoo.co.id
[2]Edi_Noersasongko@yahoo.co.id
[3]Hendro_Subagyo@yahoo.co.id

Abstrak: Penyimpulan adalah proses pengumpulan bagian yang paling penting dari sebuah sumber dokumen yang menghasilkan versi yang lebih singkat. Metode yang dianggap paling layak untuk melakukan penyimpulan adalah metode berbasis fitur dan LSA *(Latent Semantic Analysis)*. Pengklusteran adalah proses pengelompokan dokumen yang mempunyai kesamaan topik. Metode yang paling seringd ilakukan adalah LSA dimana SVD (*Singular Value Decomposition)* digunakan untuk menghubungkan semantik antara istilah dan kalimat begitu juga dengan dokumen. SVD juga mengurangi dimensi yang besar dari matriks dokumen istilah. Yang bersama dengan metode *Feature Selection* melakukan pengurangan fitur. Tesis ini memeriksa pengaruh metode penggabungan fitur dan metode LSA pada penyimpulan pada kumpulan data yang hasilnya akan diklusterkan berdasarkan pada LSA dimana SVD dilakukan bersamaan dengan metode seleksifitur. Uji coba yang dilakukan pada 150 dokumen dari 5 topik dengan beberapa kombinasi metode fitur metode LSA dan kedua metode digabungkan, pada tingkatan penyimpulan yang diintegrasikan tingkatan klusterisasi berdasarkan pada LSA dengan nilai k 12 dan metode kontribusi tema pemilih tema terbimbing memperlihatkan pengaruh yang besar pada metode yang digabungkan pada tahapan penyimpulan yang mendapatkan hasil akurasi 93.33% dan waktu komputasi yang relatif cepat berkisar 57 detik dengan proporsi penggabungan seperti berikut : Kesimpulan LSA + 50% kesimpulan Fitur+20% seleksifitur+ Klusterisasi LSA.
Kata Kunci: Penyimpulan, Pengklusteran, LSA Berbasis Fitur, LSA (*Latent Semantic Analysis)*, Seleksi Fitur, SVD.

Abstract: Summarization is a process of gathering out of the most essential parts of a document source resulting a shorter version of it. Methods considered as most appropriate in summarization are feature based and LSA (latent semantic analysis). Clusterization is a process to grouping doucuments having similar topic. Method that mostly performed is LSA where the SVD (singular value decomposition) is used to link out the semantic connection between term and sentences as well as document. SVD also reduce the high dimensionality of term-document matrix which together with feature selection performed feature reduction. This thesis examine the influence of joined feature based and LSA method in summarization on a data set which the result would be clusterized based on LSA where the SVD is performed together with feature selection method. Experiment upon 150 documents comprised of 5 topics on several combination on parameters offeature method , LSA method and both joined in summarization level integrated with clusterization level based on LSA with k rank of 12 and term contribution method as unsupervised term selection showed a significant influence of the joined method in summartzation level which resulting an accuracy of 93.33 % and relatively low computational time of 57 second in proportion combination as follows : LSA summary + Feature Summary 50 % + feature selection 20 % + LSA clusterization. Keywords : summarization, clusterization, feature based, LSA (Latent Semantic Analysis), feature selection, SVD (Singular Value Decomposition)

I. LATAR BELAKANG

Algoritma pengelompokan memainkan peran semakin penting dalam per-tumbuhan *volume* data teks di perpustakaan internet seperti *static page, dynamic page*, file dokumen, email, forum *online* dan *blog* [1]. Dalam bidang *Information Retrieval*, *clustering* dokumen adalah proses pengelompokan dokumen yang memiliki kesamaan topik [2]. *Clustering* otomatis adalah metode otomatis oleh

mesin untuk mengatur koleksi data yang besar dengan partisi data set, sehingga objek dalam *cluster* yang sama lebih mirip satu sama lain daripada objek dalam *cluster* lain [3]. Ringkasan adalah proses dari pembuatan intisari informasi terpenting dari sumber untuk meng-hasilkan versi yang lebih ringkas. Terdapat dua tipe peringkasan yaitu abstrak dan ekstrak. Abstrak menghasilkan sebuah interprestasi terhadap teks aslinya. Sebuah kalimat akan ditransformasikan menjadi kalimat yang lebih singkat, sedangkan ekstraksi merupakan ringkasan teks yang diperoleh dengan menyajikan kembali bagian tulisan yang dianggap topik utama tulisan dengan bentuk yang lebih disederhanakan [4].

Berbagai penelitian sebelumnya telah memberikan sumbangan yang signifikan terhadap perkembangan teknik peringkasan dokumen dan pengelompokannya. Catur menawarkan perbaikan atas kelemahan waktu komputasi yang relatif lama pada penggunaan *Latent Semantic Indexing* (LSI) melalui *Singular Value Decomposition* (SVD) dengan peng-gunaan *chi-square* sebagai seleksi fitur dalam *clustering* dokumen [2]. Sebelumnya, Muflikhah mengusulkan SVD sebagai metode untuk mengurangi ukuran matrik term dokumen [3]. Gupta menawarkan *Nonnegative Matrix Factorization* (NMF) untuk memperbaiki kelemahan SVD yang akibat pengurangan dimensi matriks dokumennya memunculkan berbagai komponen negatif [5]. Suanmali menawarkan konsep serupa SVD yaitu *Principal Component Analysis* (PCA) sebagai metode pengurangan dimensi matrik term dokumen [6]. Gulcin Ozsoy menawarkan metode *clustering* terdistribusi (IB) untuk melaksanakan representasi dokumen secara efisien [7]. Selanjutnya, LIU menawarkan empat metode seleksi fitur *unupervised*, DF, TC, TVQ dan TV yang bisa meningkatkan efisiensi dalam proses komputasi pada *clustering* [1]. Suanmali menawarkan metode berbasis algoritma *fuzzy logic* dalam ekstraksi kalimat untuk keperluan peringkasan [4], Gulcin Ozsoy menggunakan metode *Latent Semantic Analysis* (LSA) untuk keperluan peringkasan dokumen teks. Dia juga mengaplikasikan metode yang sama untuk peringkasan dalam dokumenteks berbahasa Turki. [8]. Penelitian ini menggabungkan metode seleksi fitur dan LSA dalam peringkasan dokumen teks sebagai model peringkas dokumen otomatis, untuk kemudian diintegrasikan pada proses *clustering* dokumen. [9].

II. LANDASAN TEORI

2.1. Preprocessing

Preprocessing adalah tahapan mengubah suatu dokumen ke dalam format yang sesuai agar dapat diproses oleh algoritma *clustering* [2]. Terdapat tiga tahapan dalam proses *Preprocessing* dalam penelitian ini, yaitu : *Tokenization*, merupakan tahapan penguraian string teks menjadi *term* atau kata. *Stopword removal,* merupakan tahapan penghapusan kata-kata yang tidak relevan dalam penentuan topik sebuah dokumen dan yang sering muncul pada dokumen, misalnya “and”, “or”, “the”, “a”, “an” pada dokumen berbahasa inggris. *Stemming,* merupakan tahapan pengubahan suatu kata menjadi akar kata nya dengan menghilangkan imbuhan awal atau akhir pada kata tersebut, misal *eating = eat, extraction = extract.* Penelitian ini menggunakan algoritma *porter stemmer*.

2.2. Metode Ekstraksi Peringkasan Teks Dokumen Otomatis *(Automatic Text Summarization)*

Metode ekstraksi adalah metode yang disusun dengan memilih kalimat-kalimat atau paragraph penting dari dokumen asli dan menggabungkannya ke dalam *form* yang lebih singkat. Pentingnya

kalimat-kalimat tersebut dibagi berdasarkan fitur *statistic* dan fitur bahasa dari kalimat [5]. Metode ekstraksi dibentuk dengan mengekstrak kunci (kalimat atau bagian) dari teks berdasar pada analisa statistik dari satu atau beberapa fitur seperti frekuensi munculnya kata atau frase, lokasi, atau kata untuk menjadikan kalimat yang diekstrak. Kata-kata yang penting diasumsikan sebagai kata-kata yang sering muncul atau kata-kata pada lokasi yang dianggap tepat [5].

2.2.1. Metode Berbasis Fitur

Untuk menggunakan metode statistik, kalimat diwakili sebagai vektor fitur. Fitur-fitur ini adalah atribut yang digunakan untuk mewakili data yang digunakan untuk tugasnya. Setiap fitur diberi nilai '0' dan '1'. Kita dapat mengekstrak sejumlah kalimat sesuai dengan tingkat kompresi. Delapan fitur yang digunakan dalam metode berbasis fitur [4] adalah :

a. fitur Judul

Dihitung dari jumlah dari kata judul pada suatu kalimat, kata pada kalimat yang terdapat pada judul memberikan skor tinggi. Hal ini ditentukan dengan menghitung jumlah kata yang sama antara kata pada suatu kalimat dengan kata pada judul. Skornya dihitung dengan rumus sebagai berikut :

$$\text{Skor}(S_i) = \text{QUOTE} \frac{Jumla\ h katapadajudul}{Jumla\ h katayangsamadenganjudul} \quad (1)$$

b. Panjang Kalimat

Dihitung dari jumlah kata pada kalimat, fitur ini berguna untuk menyaring kalimat-kalimat pendek sebagai batas akhir dan nama penulis biasanya ditemukan di artikel berita. Kalimat pendek tidak diharapkan termasuk ke dalam ringkasan. Skornya dihitung dengan rumus sebagai berikut :

$$\text{Skor}(S_i) = \frac{Jumla\ h\ kata\ \ yang\ \ ter\ dapat\ \ pada\ \ kalimat}{Jumla\ h\ kata\ \ yang\ \ terdapat\ \ pada\ \ kalimat\ \ terpanjang} \quad (2)$$

c. Bobot Kata

Dihitung dari jumlah pembagian dari TF-ISF (*Term Frequency, InvereseSentences Frequency*). Frekuensi sering munculnya kata pada suatu dokumen selalu digunakan untuk menghitung pentingnya dari suatu kalimat. Skornya dihitung dengan rumus sebagai berikut:

$$\text{Skor}(S_i) = \frac{Jumla\ h\ TF-ISF\ \ dalam\ \ kalimat}{Maksimal\ \ \ jumla\ h\ TF-ISF} \quad (3)$$

$$\text{TF-ISF} = \textit{term frequency} * \text{idf}$$

$$= \textit{term frequency} * \log\left(\frac{df}{N}\right) \quad (4)$$

Keterangan :

df = jumlah kalimat yang mengandung kata x

term frequency = jumlah kata pada dokumen (dalam bentuk matrik)

N = jumlah kalimat dalam pada dokumen

d. Posisi Kalimat

Jika kalimat pertama dan kalimat terakhir pada paragraf, maka posisi kalimat memberi pentingnya dari kalimat tersebut. Fitur ini melibatkan sejumlah *item,* seperti posisi dari kalimat, bagian, paragraf, dan lain-lain. Kalimat pertama dan kalimat terakhir memberi rangking tertinggi. Skor untuk fitur ini adalah 1 untuk kalimat pertama dan kalimat terakhir, 0 untuk kalimat lainnya. Skornya dihitung dengan rumus sebagai berikut :

$\text{Skor}(S_i)$ = 1 untuk kalimat pertama dan kalimat terakhir. 0 untuk kalimat lainnya (5)

e. Kesamaan Antar Kalimat

Dihitung dari kesamaan antarkalimat, untuk tiap kalimat s, kesamaan antara s dan kalimat lainnya dihitung dengan pengukuran *cosine similarity.*Skor fitur ini untuk tiap kalimat a dihitung dari rasio ringkasan kesamaan kalimat dari kalimat s dan kalimat lainnya. Berikut ini adalah rumus menghitung *cosine similarity :*

$$\text{sim}_{\cos}(d_i,d_j) = \frac{\sum_{k=1}^{n} w_{ik} X\ w_{jk}}{\sqrt{\sum_{k=1}^{n} w_{ik}^2}\ X\ \sqrt{\sum_{k=1}^{n} w_{jk}^2}} \quad (6)$$

w_{ik} = Bobot kata pada dokumen

w_{jk} = Bobot kata pada query

sedangkan untuk menghitung skor dari fitur ini adalah[4] :

$$\text{Skor}(S_i) = \frac{jumlah\ cosine\ similarity}{jumlah\ maksimal\ similarity} \quad (7)$$

f. Kata Tematik

Dihitung dari jumlah kata tematik pada suatu kalimat, fitur ini penting karena kata yang sering muncul pada dokumen akan lebih sering dikaitkan dengan topik. Yang dimaksud kata tematik disini adalah kata-kata yang ada dalam daftar *library*. Skornya dihitung dengan rumus sebagai berikut :

$$\text{Skor}(S_i) = \frac{jumlah\ kata\ tematik\ dalam\ kalimat}{panjang\ kalimat(jumlah\ kata\ pada\ kalimat)} \quad (8)$$

g. Data Numerik

Dihitung dari jumlah data numerik pada kalimat. Kalimat yang berisi data numerik itu penting dan banyak kemungkinan termasuk ke dalam hasil ringkasan dokumen.

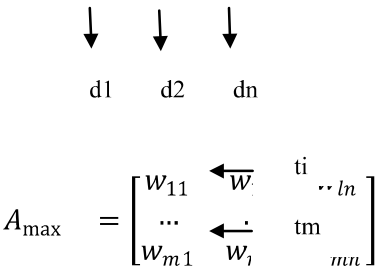
$$\text{Skor}(S_i) = \frac{jumla\ h\ data\ numerik}{panjang\ \ kalimat\ \ (jumla\ h\ kata\ pada\ kalimat\ )} \quad (9)$$

2.2.2. Metode Berbasis LSA (Latent Semantic Analysis)

LSA *(Latent Semantic Analysis)* adalah metode statistik aljabar yang mengekstrak struktur semantik yang tersembunyi dari kata dan kalimat. LSA ini menggunakan konteks yaitu memasukkan dokumen dan mengekstrak informasi dari kata yang digunakan bersama dan kata-kata umum yang sering dilihat pada kalimat yang berbeda. Jika jumlah dari kata-kata umum pada kalimat dalam jumlah banyak, itu berarti kalimat tersebut lebih banyak bersifat semantik [7]. Untuk mencari interelasi diantara kalimat dan kata, metode aljabar yang dinamakan *Singular Value Decomposition* (SVD) digunakan. SVD juga mempunyai kapasitas reduksi *noise* yang membantu untuk meningkatkan akurasi [8]. Algoritma peringkas dokumen teks yang berbasis pada LSA ini biasanya terdiri dari tiga tahap [8], yaitu: pembentukan matrik input dengan dokumen yang diinput ditunjukkan dengan matrik untuk menampilkan kalkulasi, *Singular Value Decomposition* (SVD), dan penyeleksian kalimat

2.2.2.1. *Vector Space Model*

Vector Space Model (VSM) mengubah koleksi dokumen ke dalam matrik *term-document* [9]. Matrik *term-document* (Gambar 1) tersebut memiliki dimensi *mxn* dimana *m* adalah jumlah *term* dan *n* adalah jumlah dokumen.



Gambar 1. Matrik Term Dokumen

Keterangan :

t1 : *term* ke 1

tm : *term* ke m

d1 : dokumen ke 1

dn : dokumen ke n

w : adalah nilai atau bobot setiap *term* dalam dokumen

2.2.2.2. *Term Weighting*

Terdapat tiga (3) metode pembobotan atau *term weighting* dalam VSM yaitu *Term Frequency* (TF), *Invers Document Frequency* (IDF) dan *Term Frequency Invers Document Frequency* (TFIDF). TF adalah banyaknya kemunculan suatu *term* dalam suatu dokumen, IDF adalah perhitungan logaritma antara pembagian jumlah total dokumen dengan cacah dokumen yang mengandung suatu *term*, dan TFIDF adalah perkalian antara TF dengan *IDF*. Semakin besar bobot TFIDF pada

suatu *term*, semakin penting *term* tersebut untuk digunakan pada tahapan.

$$IDF = log\frac{D}{DF} \quad (10)$$

$$TFIDF(t) = TF * \log\frac{D}{DF} \quad (11)$$

Keterangan :

IDF : Perhitungan logaritma antara pembagian jumlah total dokumen dengan cacah dokumen yangmengandung suatu *term*

D : Jumlah total dokumen

DF : Banyaknya dokumen yang mengandung t*erm*

TF : Banyaknya kemunculan suatu *term* dalam suatu dokumen

TFIDF : Perkalian antara TF dengan *IDF*

2.2.2.3. *Similiarity Measure*

Cosines Similarity akan mengukur jarak antara dua dokumen *di* dan *dj*, besarnya nilai *cosines* mengindikasikan bahwa dua dokumen tersebut memiliki kemiripan yang tinggi [11].

$$similarity(d_{i,}\ d_j) = \frac{di.dj}{\|di\|.\|dj\|} \quad (12)$$

2.2.2.4. Teknik *Dimension Reduction*

Ada dua teknik dalam *feature reduction* yaitu *feature selection* dan *feature transformation*. Pada *feature selection* dapat dibedakan menjadi 2 jenis berdasarkan ada tidaknya informasi label atau keberadaan informasi awal tentang kategori dari dokumen yaitu *supervised feature selection* dan *unsupervised feature selection*. Pada teknik *feature selection* metode *supervised feature selection* diantaranya adalah *Information Gain* (IG) dan x^2 *statistic*(CHI) dan untuk *supervised feature selection* metode yang digunakan antara lain *document Frequency (DF), Term Contribution* (TC), *Term Variance* (TV), dan *Term Variance Quality* (TVQ), dalam penelitian ini *unsupervised feature reduction* berupa *Term Contribution* (TC), sedangkan untuk *feature transformation* digunakan *Latent Semantic Indexing (*LSI) dengan *Singular Value Decomposition* (SVD). Dalam penelitian ini digunakan *Term Contribution (TC)*. TC diperkenalkan pertama kali oleh Tao Liu dan kawan-kawannya pada tahun 2003 [11]. Disebutkan bahwa hasil dari *clustering* teks mempunyai ketergantungan dengan kesamaan dokumen.sehingga, kontribusi dari sebuah term dapat diartikan sebagai kontribusi terhadap kesamaan dokumen, kesamaan antar dokumen ***di*** dan ***dj***dapat dihitung menggunakan *dot product:*

$$sim(d_{i,}\ d_j) = \frac{d_i\ .\ d_j}{||d_i||.\ ||d_j||} \quad (15)$$

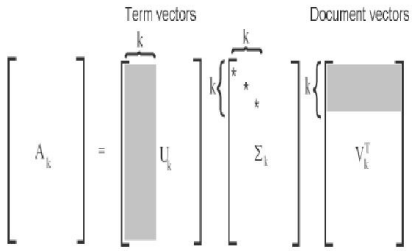
$$TC(t) = \sum_t f(t,di).f(t,dj) \quad (16)$$

dimana, *f*(*t*,*d*) merupakan bobot *tf***idf* dari term *t* di dokumen *d.*

Latent Semantic Indexing (LSI) sebagai ekstensi VSM untuk mengatasi permasalahan munculnya dimensi tinggi pada VSM, dimana pendekatan dilakukan pada *term-document* dengan menggunakan teknik pengurangan dimensi [10]. *Latent Semantic Indexing* (LSI) melalui metode *Singular Value Decomposition* (SVD) mengurai matrik term-document menjadi 3 matrik *U*, *S* dan *V* yang memiliki dimensi lebih kecil.

$$A = USV^T \quad (19)$$

Dimana *U* merupakan matrik term yang berdimensi *m* x *k*, *S* adalah matrik diagonal yang berisi *eigen value* berdimensi *k x k* dan V^T adalah matrik dokumen yang memiliki dimensi *k* x *n*.



Gambar 2. Dekomposisi *truncated* SVD.

Truncated SVD menggunakan pendekatan *rank-k* untuk mengurangi SVD, terdapat tingkat kemiripan dengan matrik term-document dengan matrik yang dihasilkan dengan *truncated* SVD. SVD sangat cocok diterapkan untuk varian matrik yang banyak mengandung nilai 0, sedangkan hal yang perlu diperhatikan dari SVD adalah SVD rumit dalam proses perhitungan, dalam satu kali proses perhitungan itu hanya mencerminkan dekomposisi dari matrik asli [10].

2.3 K-means

K-means adalah algoritma *clustering* yang cukup sederhana dan mampu diimplementasikan untuk koleksi data yang besar untuk dikelompokkan kedalam beberapa *cluster* [2]. *K-means* memilih beberapa dokumen secara acak untuk dijadikan *centroid* atau pusat *cluster*. Banyaknya *centroid* menentukan jumlah *cluster* yang akan dihasilkan. Berikut adalah *pseudocode* dari algoritma K-*Means* [11].

Algoritma K-Means Clustering :

Input : Koleksi Dokumen *D={d1, d2, d3, ... dn}*; Jumlah *cluster* (*k*) yang akan dibentuk*;*

Output: *k cluster;*

Proses :

1. Memilih *k* dokumen untuk dijadikan *centroid* (titik pusat *cluster*) awal secara random*;*
2. Hitung jarak setiap dokumen ke masing-masing *centroid* menggunakan persamaan *cosines similarity* (persamaan 3) kemudian jadikan satu *cluster* untuk tiap-tiap dokumen yang memiliki jarak terdekat dengan *centroid*;
3. Tentukan *centroid* baru dengan cara menghitung nilai rata-rata dari data-data yang ada pada *centroid* yang sama;
4. Kembali ke langkah 2 jika posisi *centroid* baru dan *centroid* lama tidak sama;

2.4 Evaluation Measure

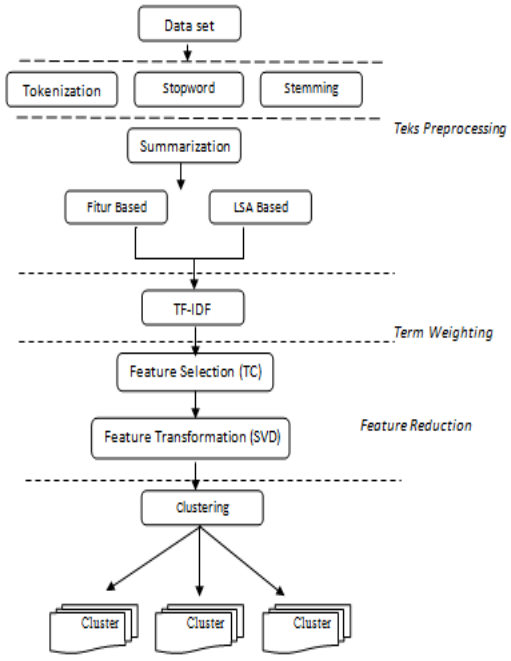
Dalam penelitian ini, digunakan *F-measure* dari pengukuran *precision* dan *recall* untuk mengukur kinerja *clustering*. *Recall* adalah rasio dokumen yang relevan yang terambil dengan jumlah seluruh dokumen dalam koleksi dokumen, sedangkan *precision* adalah rasio jumlah dokumen relevan terambil dengan seluruh jumlah dokumen terambil. Nilai *recall* dan *precision* yang tinggi menunjukan tingkat keakuratan dari sebuah *clustering[3]*.

$$Recall\ (i,j) = \frac{nij}{ni} \quad (14)$$

$$Precision\ (i,j) = \frac{nij}{nj} \quad (15)$$

III. EKSPERIMEN

Guna menentukan solusi atas masalah penelitian yang diungkap di latar belakang, maka disusunlah model berikut ini. Model ini diawali dengan data set yang kemudian dimasukkan ke tahap *text processing, term weighting, feature reduction,* yang kemudian diakhiri dengan proses *clustering.*



Gambar 1. Model yang Diusulkan

3.1. Dataset

Data set terdiri dari 150 dokumen berita yang diambil dari yahoo news yang terdiri atas 5 topik, ekonomi (EC), sport (SP), politik (PL), hukum (HK) dan kriminal (KR) masing-masing 30 dokumen berita.

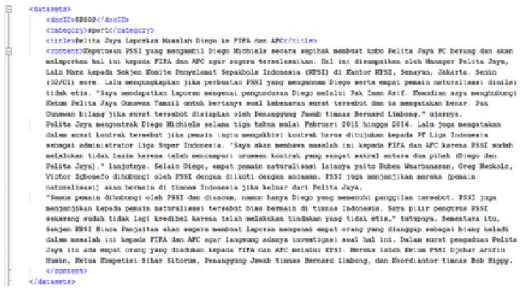
3.2. Eksperimen

Model *clustering* pada percobaan yang dilakukan, yaitu:

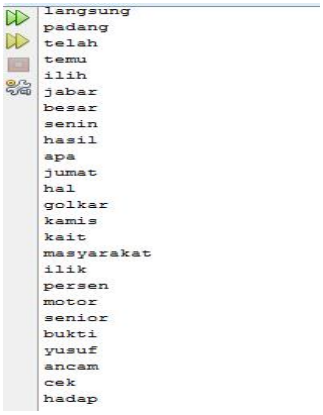
a. Model *clustering* standar
b. Model *clustering* menggunakan *feature selection* standar dan *clustering* dengan menggunakan peringkas dokumen otomatis dengan metode LSA (*Feature Selection* +LSA)
c. Model clustering menggunakan *feature selection standard* dan *clustering* dengan menggunakan peringkas dokumen otomatis dengan menggabungkan metode Fitur dan LSA (*Feature Summary* + *Feature Selection* + LSA)
d. Model *clustering* menggunakan *feature selection* standar dan *clustering* dengan menggunakan peringkas dokumen otomatis dengan metode LSA (LSA *Summary* + *Feature Selection* + LSA)
e. Model *clustering* menggunakan *feature selection* standar dan *clustering* dengan menggunakan peringkas dokumen otomatis dengan penggabungan metode Fitur dan metode LSA (LSA *Summary* + *Feature Summary* + *Feature Selection*+ LSA)

Urutan langkah pada *clustering* standar pada percobaan yang dilakukan adalah: *Tokenization*, penghapusan *Stopword*, *Stemming*, dan proses *K-means* untuk tahap *clustering* dokumen. Berikutnya urutan langkah pada *clustering* menggunakan *feature reduction* standar adalah: *Tokenization*, penghapusan *Stopword, Stemming*, Pembobotan TF, Pembobotan TFIDF, *unsupervised feature selection* TC, LSI *feature transformation* dan proses *K-means* untuk tahap *clustering* dokumen. Sedangkan urutan langkah pada *clustering* dengan menggunakan peringkas dokumen otomatis yang diintegrasikan sebagai *feature reduction* adalah: *Tokenization*, penghapusan *Stopword*, *Stemming*, proses peringkas dokumen otomatis, Pembobotan TF, Pembobotan TFIDF, *unsupervised feature selection* TC, LSI *feature transformation* dan proses *k-means* untuk tahap *clustering* dokumen.

Gambar 4. menunjukan dokumen asli sebelum *preprocessing* dan Gambar 5. menunjukan dokumen setelah tahap *preprocessing*.



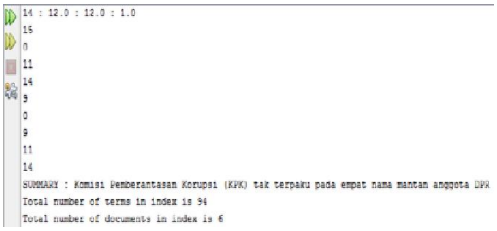
Gambar 4. Dokumen asli sebelum *preprocessing*



Gambar 5. Hasil dokumen setelah proses *preprocessing*

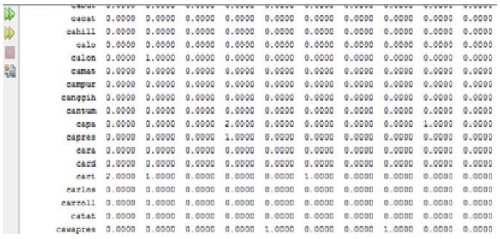
Setelah proses *tokenization*, *stopword* dan *stemming* selesai, selanjutnya dilakukan proses pemenggalan kalimat baru kemudian tahap berikutnya adalah proses peringkasan dokumen otomatis berbasi metode peringkasan Fitur.

Gambar 6. menunjukan sampel hasil peringkas dokumen otomatis pada salah satu dokumen.

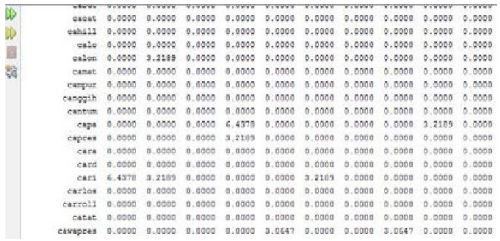


Gambar 6. Proses peringkas dokumen otomatis

Setelah dihasilkan dokumen dalam bentuk *summary* dokumen direpresentasikan ke dalam bentuk matrik *term-dokument mxn* menggunakan *Vector Space Model*. Proses selanjutnya adalah proses pembobotan TF, pembobotan TF menghasilkan matriks term frekuensi. Kolom dari matriks merepresentasikan dokumen dan baris dari matrik merupakan *term* dalam dokumen. TF menunjukkan jumlah setiap istilah dalam setiap dokumen, karena nilai TF matriks adalah nilai *integer* positif. Gambar 7. menunjukkan hasil pembobotan TF. Kemudian, dari hasil TF digunakan untuk menghitung pembobotan TFIDF. TFIDF adalah perkalian antara TF dan *inverse document frequency* (IDF). Gambar 8. menunjukkan hasil pembobotan TFIDF.



Gambar 7. Hasil pembobotan TF

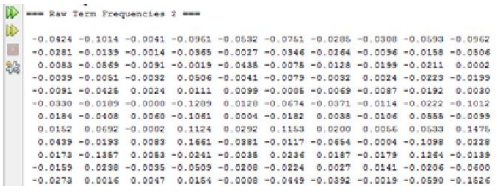


Gambar 8. Hasil Pembobotan TFIDF

Setelah pembobotan di atas menggunakan TFIDF matrik telah terbentuk dan peran dari *unsupervised feature selection* digunakan untuk menghilangkan *term* yang memiliki tingkat kemunculan yang paling sedikit untuk penentuan *clustering* dokumen. *Term* dari hasil pembobotan diurutkan dari *term* yang mempunyai bobot paling besar, urut secara *descending*. Bobot tinggi pada sebuah *term* merupakan definisi untuk *term* yang mempunyai pengaruh tinggi dalam proses *clustering* dokumen. Penelitian ini menggunakan n% jumlah *term* yang mempunyai bobot tertinggi dari keseluruhan jumlah *term* yang ada, dimana *term* yang dibawah n% yang telah ditentukan akan dihilangkan. Pada penelitian ini untuk jumlah % *term* yang diproses yaitu 20%, 30%, 40% dan 60%. Alasan pemilihan angka-angka proporsi tersebut adalah karena angka-angka tersebut adalah *milestone* dari angka-angka moderat antara 0% dan 100%.

Tahap berikutnya adalah *feature transformation* SVD. SVD digunakan untuk mengurangi matrik *term* dokumen ke dalam matrik yang mempunyai dimensi lebih rendah. *Truncated* SVD menggunakan pendekatan peringkat-*k* untuk mengurangi SVD [12]. Percobaan ini menggunakan peringkat-k pembulat-an nilai akar dari jumlah dokumen yang diproses, yaitu pembulatan dari $\sqrt{150} = 12$.

Gambar 9. menunjukkan Hasil Dekomposisi SVD



Gambar 9. Hasil Dekomposisi SVD

Tahapan yang terakhir adalah algoritma *clustering K-means*, untuk 5 titik pusat *cluster* ditentukan secara manual dengan mengambil satu dokumen dari 5 kategori yang ada. Dari percobaan

yang dilakukan 5 dokumen yang digunakan sebagai titik pusat *cluster* adalah dokumen dengan *id* SP001, EC001, HK001, KR001 dan PL001.

Dalam percobaan yang dilakukan diawali dengan mengukur tingkat akurasi *clustering* dari *original K-Means* dimana proses *clustering* tanpa menggunakan metode pengurangan fitur/*feature reduction* baik *feature selection* maupun *feature transformation*, percobaan berikutnya proses *clustering* dengan ditambahkan proses *feature reduction* yang mencakup *feature selection* dan *feature transformation*, parameter untuk *feature selection* yang digunakan adalah 20%, 30%, 40% dan 60% sedangkan *feature transformation* menggunakan SVD dengan peringkat-*k* 12 (pembulatan nilai akar dari jumlah dokumen yang diproses). Dan percobaan yang terakhir adalah *feature reduction* yang diintegrasikan dengan peringkas dokumen otomatis, proses peringkas dokumen otomatis ini dijalankan sebelum proses *feature selection* dan *feature transformation* yang merupakan proses *feature reduction* standar. Gambar 10. menunjukan daftar dokumen untuk masing-masing hasil *cluster* dari proses *clustering* dokumen.

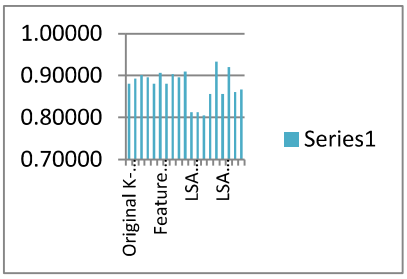
```
=== Clusters from K-Means Algorithm ===
sport:[SP001, SP002, SP004, SP006, SP007, SP008, SP009, SP010, SP011, SP012, SP
economy:[EC001, EC002, EC003, EC004, EC005, EC006, EC007, EC008, EC009, EC010,
hukum:[EC024, HK001, HK002, HK003, HK004, HK005, HK006, HK007, HK008, HK009, HK
kriminal:[SP003, HK025, HK029, KR001, KR003, KR004, KR005, KR006, KR007, KR008,
politik:[SP005, HK014, HK023, KR011, KR016, KR019, KR020, KR023, KR024, KR025,
F-measure : 0.9178333333333333
BUILD SUCCESSFUL (total time: 39 seconds)
```

Gambar 10. Hasil Proses *clustering* dokumen *k-means*

IV. HASIL DAN PEMBAHASAN

4.1. Akurasi

Dari hasil penelitian yang dilakukan dapat dibuktikan bahwa integrasi peringkas dokumen otomatis dengan menggabungkan metode fitur dan *Latent Semantic Analysis* (LSA) dapat meningkatkan akurasi hasil *clustering* pada dokumen teks Berbahasa Indonesia.



Gambar 11. Hasil kinerja proses *clustering* dokumen

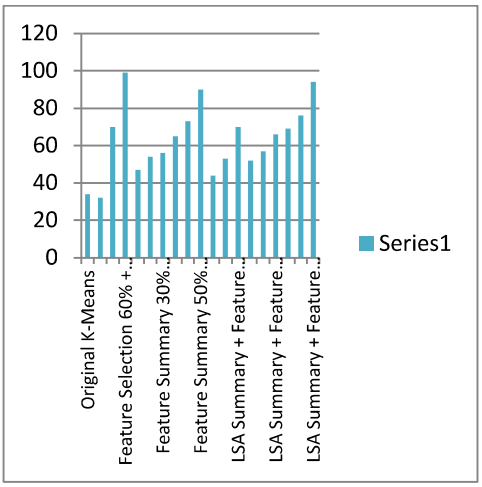
Tingkat akurasi menggunakan peringkas dokumen otomatis yang diintegrasikan sebagai *feature reduction* dengan menggabungkan metode fitur dan metode LSA pada percobaan di atas mencapai 93,33 % yang diperoleh pada tingkat peringkas dokumen otomatis LSA *Summary*+ *Feature Summary* 50% + *Feature Selection* 20% + LSA dibandingkan dengan *feature selection* 20 % tanpa menggunakan peringkas dokumen otomatis yang hanya mencapai tingkat akurasi 89,33 %. Dari Gambar 11. juga dapat dilihat penurunan tingkat akurasi untuk % *feature selection* yang lain, akan tetapi pada proporsi 60 % *feature selection* integrasi peringkas dokumen otomatis dengan metode LSA mengalami penurunan tingkat akurasi.

Tingkat akurasi pada proporsi *feature selection* 20% dengan integrasi mesin peringkas menggunakan metode fitur mengalami kenaikan tingkat akurasi pada prosporsi *feature selection* 40% dan pada proporsi *feature selection* 60% tingkat akurasi mengalami kenaikan dari proporsi *feature selection* 40%. Kemudian pada percobaan dengan kombinasi proporsi *feature selection* 20% mengalami penurunan tingkat akurasi dari proporsi *feature selection* 60% dengan integrasi peringkas otomatis dengan metode LSA. Kemudian pada percobaan terakhir dari proporsi *feature selection* 20% mengalami kenaikan pada proporsi *feature selection* 40%. Kemudian mengalami penurunan lagi proporsi *feature selection* 60% pada pada percobaan dengan kombinasi proporsi *feature selection* dengan integrasi peringkas otomatis

dengan penggabungan metode fitur dan metode LSA. Dari hasil percobaan tersebut bahwa semakin kecil proporsi dari % *feature selection* pada proses *clustering* dokumen tidak dapat dipastikan menghasilkan tingkat akurasi *clustering* yang semakin tinggi dan proporsi % *feature selection* yang semakin besar juga tidak dipastikan dapat menghasilkan tingkat akurasi *clustering* yang semakin rendah. Dari percobaan yang dilakukan tingkat proporsi % *feature selection* yang proporsional dan menghasilkan tingkat akurasi tertinggi dengan dataset yang diolah adalah % *feature selection* 20%.

4.2. Waktu

Waktu rata-rata yang diambil diukur mulai dari proses *preprocessing* sampai dengan hasil *clustering* diperoleh.



Gambar 12. Waktu Proses *Clustering* Dokumen

Tabel 1. Tabel Waktu Proses *Clustering* Dokumen

Metode	Second	Menit	Detik	Time
Original K-Means	34		34	34 seconds
Feature Selection 20% + LSA	32		32	32 seconds
Feature Selection 40% + LSA	70	60	10	1 minute 10 seconds
Feature Selection 60% + LSA	99	60	39	1 minute 39 seconds
Feature Summary 30% + Feature Selection 20% + LSA	47		47	47 seconds
Feature Summary 50% + Feature Selection 20% + LSA	54		54	54 seconds
Feature Summary 30% + Feature Selection 40% + LSA	56		56	56 seconds
Feature Summary 50% + Feature Selection 40% + LSA	65	60	5	1 minute 5 seconds
Feature Summary 30% + Feature Selection 60% + LSA	73	60	13	1 minute 13 seconds
Feature Summary 50% + Feature Selection 60% + LSA	90	60	30	1 minute 30 seconds
LSA Summary + Feature Selection 20% + LSA	44		44	44 seconds
LSA Summary + Feature Selection 40% + LSA	53		53	53 seconds
LSA Summary + Feature Selection 60% + LSA	70	60	10	1 minute 10 seconds
LSA Summary + Feature Summary 30% + Feature Selection 20% + LSA	52		52	52 seconds
LSA Summary + Feature Summary 50% + Feature Selection 20% + LSA	57		57	57 seconds
LSA Summary + Feature Summary 40% + Feature Selection 40% + LSA	66	60	6	1 minute 6 seconds
LSA Summary + Feature Summary 50% + Feature Selection 40% + LSA	69	60	9	1 minute 9 seconds
LSA Summary + Feature Summary 30% + Feature Selection 60% + LSA	76	60	16	1 minute 16 seconds
LSA Summary + Feature Summary 50% + Feature Selection 60% + LSA	94	60	34	1 minute 34 seconds

Tabel 1. menunjukan bahwa pada % *feature selection* yang semakin kecil *feature reduction* yang diintegrasikan dengan peringkas dokumen otomatis membutuhkan tambahan waktu komputasi

tersendiri, dari percobaan yang dilakukan untuk 20% *feature selection* terdapat peningkatan waktu komputasi dari percobaan *clustering* tanpa peringkas dokumen otomatis, menggunakan peringkas dokumen otomatis dengan proporsi 30%, 40% dan 60%. Akan tetapi pada proporsi *feature selection* yang semakin besar, % peringkas dokumen otomatis dapat menurunkan waktu komputasi yang ada, pada percobaan 40% dan 60% *feature selection* dapat dilihat bahwa integrasi peringkas dokumen otomatis sebagai *feature reduction* dapat mengurangi rata-rata waktu komputasi yang dibutuhkan.

V. PENUTUP

Eksperimen atas 150 dokumen yang terdiri atas 5 topik pada berbagai kombinasi parameter dari metode fitur dan LSA pada peringkasan serta seleksi fitur dan LSA pada *clustering* membuktikan pencapaian tingkat akurasi yang lebih tinggi (93.33%) dengan waktu komputasi yang relatif singkat (57 detik) pada kombinasi sebagai berikut : *LSA summary + Feature Summary 50 % + feature selection 20 % + LSA clusterization.*

DAFTAR PUSTAKA

[1] Luying LIU, KANG Jianchu, Jing YU, and Zhongliang WANG, "A Comparative Study on Unsupervised Feature Selection Methods for Text Clustering," IEEE, pp. 597-601, 2005.

[2] Supriyanto Catur and Affandy, "Kombinasi Teknik Chi Square dan Singular Value Decomposition untuk Reduksi Fitur pada pengelompokan Dokumen," Semantik Udinus, pp. 1-8, 2011.

[3] Lailil Muflikhah and Baharum Baharudin, "Document Clustering using Concept Space and Cosine Similarity Measurement," International Conference on Computer Technology and Development, pp. 58-62, 2009.

[4] Ladda Suanmali, Naomie Salim, and Mohammed Salem Binwahlan, "Automatic Text Summarization Using Feature Based Fuzzy Extraction," Iurnal Teknologi Maklumat, pp. 105-115, Desember 2008.

[5] Vishal Gupta and Gurpreet Singh Lehal, "A Survey of Text Summarization Extractive Techniques," Journal Of Emerging Technologies In Web Intellince, vol. 2, pp. 258-268, AUGUST 2010.

[6] Ladda Suanmali, Naomie Salim, and Binwahlan Salem Mohammed, "Automatic Text Summarization Using Feature Based Fuzzy Extraction," Iurnal Teknologi Maklumat, pp. 105-115, 2008.

[7] Makbule Gulcin Ozsoy, Cicekli Ilyas, and Ferda Nur Alpaslan, "Text Summarization of Turkish Texts using Latent Semantic Analysis," Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pp. 869-876, Agustus 2010.

[8] Özsoy Makbule Gülçin, Dr. Ferda Nur Alpaslan, and İlyas Çiçekli, "Text Summarization Using Latend Semantic Analysis," pp. 1-69, 2011.

[9] M. Thangamani and P. Thangaraj , "Integrated Clustering and Feature Selection Scheme for Text Documents," Journal of Computer Science, pp. 536-541, 2010.

[10] P.Prabhu and N.Anbazhagan , "Improving the Performance of K-Means Clustering For High Dimensional Data Set," International Journal on Computer Science and Engineering (IJCSE), vol. 3, pp. 2317-2322, Juni 2011.

[11] Tao Liu , Shengping Liu , Zheng Chen , and Wei-Ying Ma , "An Evaluation on Feature Selection for Text Clustering," Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), 2003.

[12] QING YANG and FANG-MIN LI1 , "SUPPORT VECTOR MACHINE FOR CUSTOMIZED EMAIL FILTERING BASED ON IMPROVING LATENT SEMANTIC INDEXING," Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, pp. 3787-3791, Agustus 2005.