PENENTUAN TINGKAT PLAGIARISME DOKUMEN PENELITIAN MENGGUNAKAN *CENTROID LINKAGE HIERARCHICAL METHOD* (CLHM)

SKRIPSI

Konsentrasi Komputasi Cerdas dan Visualisasi

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh:

WINDA AYU IRIANTO

NIM. 0910683097

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

PROGRAM STUDI TEKNIK INFORMATIKA

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2013

LEMBAR PERSETUJUAN

PENENTUAN TINGKAT PLAGIARISME DOKUMEN PENELITIAN MENGGUNAKAN *CENTROID LINKAGE HIERARCHICAL METHOD* (CLHM)

SKRIPSI

Konsentrasi Komputasi Cerdas dan Visualisasi

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh:

WINDA AYU IRIANTO

NIM. 0910683097

Telah diperiksa dan disetujui oleh :

Dosen Pembimbing I,	Dosen Pembimbing II,
Drs. Achmad Ridok, M.Kom.	**Lailil Muflikhah, S.Kom, M.Sc.**
NIP. 19680825 199403 1 002	**NIP. 19741113 200501 2 001**

LEMBAR PENGESAHAN

PENENTUAN TINGKAT PLAGIARISME DOKUMEN PENELITIAN MENGGUNAKAN *CENTROID LINKAGE HIERARCHICAL METHOD* (CLHM)

SKRIPSI

Konsentrasi Komputasi Cerdas dan Visualisasi

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :

Winda Ayu Irianto

NIM. 0910683097

Skripsi ini telah diuji dan dinyatakan lulus pada

tanggal 14 Juni 2013

Penguji I,	Penguji II,
Edy Santoso, S.Si., M.Kom.	**Budi Darma S., S.Kom, M.Sc.**
NIP. 19740414 200312 1 004	**NIP. 841015 06 1 1 0090**

Penguji III,

Indriati, S.T., M.Kom.

NIK. 831013 06 1 2 0035

Mengetahui,

Ketua Program Studi Teknik Informatika,

Drs. Marji, M.T.

NIP. 19670801 199203 1 001

PERNYATAAN
ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah SKRIPSI ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata didalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku. (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 14 Juni 2013
Mahasiswa,

Winda Ayu Irianto
NIM 0910683097

KATA PENGANTAR

Alhamdulillahi rabbil 'alamin. Puji syukur penulis panjatkan kehadirat Allah SWT, karena atas segala rahmat dan limpahan hidayah-Nya, penulis dapat menyelesaikan tugas akhir yang berjudul "PENENTUAN TINGKAT PLAGIARISME DOKUMEN PENELITIAN MENGGUNAKAN CENTROID LINKAGE HIERARCHICAL METHOD (CLHM)". Dalam pelaksanaan dan penulisan tugas akhir ini penulis mendapatkan banyak bantuan dari berbagai pihak baik secara moril maupun materiil. Dalam kesempatan ini penulis ingin mengucapkan terima kasih yang sebesar – besarnya kepada :

1. Drs. Achmad Ridok, M.Kom. dan Lailil Muflikhah, S.Kom., M.Sc., selaku dosen pembimbing selama pelaksanaan skripsi.
2. Ir. Sutrisno, M.T., Ir. Heru Nurwasito, M.Kom., Himawat Aryadita, S.T., M.Sc., dan Edy Santoso, S.Kom., selaku Ketua, Wakil Ketua 1, Wakil Ketua 2 dan Wakil Ketua 3 Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
3. Drs. Marji, M.T. dan Issa Arwani, S.Kom., M.Sc., selaku Ketua dan Sekretaris Program Studi Teknik Informatika Universitas Brawijaya.
4. Ayahanda Eko Irianto, Ibunda Winarsih Irianto, Adik Savitri Oktavia Irianto, serta seluruh keluarga besar atas segala nasehat, kasih sayang, perhatian dan kesabarannya di dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti-hentinya memberikan doa dan semangat demi terselesaikannya tugas akhir ini.
5. Seluruh Dosen Teknik Informatika Universitas Brawijaya atas kesediaan membagi ilmunya kepada penulis.
6. Seluruh Civitas Akademika Teknik Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Teknik Informatika Universitas Brawijaya dan selama penyelesaian tugas akhir ini.
7. Wahyu Dian Ripnoyo, Putri Setya Adhita W., Fauziah Mayasari, Milani Winangga, Arianty Anggraini, Novelia Kharisma S., Ari Agustina, Adestiana Rahmawati, Dian Arisandy, Hanifa Vidya Rizanti, Bekti Widyaningsih, Wina

Safitri, Chalimatus Sa'diyah, dan Nurul Lailiyah yang selalu bertukar semangat dengan penulis selama menyelesaikan tugas akhir ini.

8. Teman–teman Konsentrasi Komputasi Cerdas dan Visualisasi serta seluruh Angkatan 2009 Teknik Informatika, terima kasih atas segala bantuannya selama menempuh studi di Teknik Informatika Universitas Brawijaya.
9. Seluruh pihak yang telah membantu kelancaran penulisan tugas akhir yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari bahwa dalam penyusunan tugas akhir ini masih banyak kekurangan baik format penulisan maupun isinya. Oleh karena itu, saran dan kritik membangun dari para pembaca senantiasa kami harapkan guna perbaikan bagi tugas akhir selanjutnya. Semoga tugas akhir ini dapat memberikan manfaat bagi semua pihak, Amin.

Malang, Juni 2013

Penulis

ABSTRAK

Winda Ayu Irianto. 2013 : Penentuan Tingkat Plagiarisme Dokumen Penelitian menggunakan Centroid Linkage Hierarchical Method (CLHM). Skripsi Program Studi Teknik Informatika, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya.
Dosen Pembimbing: Drs. Achmad Ridok, M.Kom. dan Lailil Muflikhah, S.Kom., M.Sc.

Tindakan plagiarisme sering ditemukan pada hasil karya tulis mahasiswa. Tingkat plagiarisme dapat diminimalkan dengan mendeteksi kemiripan dokumen penelitian. Pada penelitian ini telah dikembangkan sistem penentuan tingkat plagiarisme dokumen menggunakan *Centroid Linkage Hierarchical Clustering* (CLHM) berdasarkan kemiripan topik pada abstraksi dokumen penelitian. CLHM digunakan untuk mengelompokkan dokumen berdasarkan kemiripannya menggunakan *hill climbing*. Perhitungan prosentase kemiripan pada sistem ini menggunakan *cosine similarity*. Evaluasi sistem dilakukan terhadap 100 data yang dihilangkan sebesar 25%, 50%, dan 75%. Masing-masing perlakuan data diuji pengaruhnya terhadap pemotongan kata sebelum *stemming* dan setelah *stemming*. Hasil uji coba menghasilkan prosentase *error* masing-masing 4,893%, 5,399%, 4,196%, dan 2,501%, 5,256%, 9,271% serta hasil akurasi sistem menghasilkan rata-rata f-measure sebesar 0,984.

Kata Kunci: plagiarisme, *centroid linkage hierarchical method*, *hill climbing*, *cosine similarity.*

ABSTRACT

Winda Ayu Irianto. 2013: ***Determination Levels of Plagiarism Research Documents using Centroid Linkage Hierarchical Method (CLHM). Minor Thesis Informatics Engineering Program, Program on Information Technology and Computer Science, University of Brawijaya.***
Supervisors: **Drs. Achmad Ridok, M.Kom. and Lailil Muflikhah, S.Kom., M.Sc.**

Plagiarism are often found in the papers of students. Level of plagiarism can be minimized by detecting the similarity of research papers. In this study, the system has been developed to define the level of plagiarism document using Centroid Linkage Hierarchical Method (CLHM) based on similarity of research topics on document abstractions. The method used to classify documents based on similarity using hill climbing. The percentage calculation of similarity using cosine similarity. Evaluation system is conducted on one hundred documents reduce by 25%, 50%, and 75%. Each of them tested their effects on the data before stemming and after stemming. The experiment results show the error percentage respectively 4.893%, 5.399%, 4.196%, and 2.501%, 5.256%, 9.271%. The accuracy result of the system produces an average f-measure of 0.984.

Keywords: *plagiarism, centroid linkage hierarchical method, hill climbing, cosine similarity.*

DAFTAR ISI

LEMBAR PERSETUJUAN ii
LEMBAR PENGESAHAN iii
PERNYATAAN ORISINILITAS SKRIPSI iv
KATA PENGANTAR v
ABSTRAK viii
ABSTRACT ix
DAFTAR ISI x
DAFTAR TABEL xii
DAFTAR GAMBAR xiii
DAFTAR *SOURCE CODE* xiv
BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 3
1.4 Tujuan 3
1.5 Manfaat 3
1.6 Sistematika Penyusunan Laporan 4
BAB II KAJIAN PUSTAKA DAN DASAR TEORI 6
2.1 Kajian Pustaka 6
2.2 *Text Mining* 7
2.2.1 *Tokenizing* 8
2.2.2 *Filtering* 9
2.2.3 *Stemming* 9
2.2.4 *Analizing* 13
2.3 *Clustering* 13
2.3.1 Karakteristik *Clustering* 13
2.3.2 Manfaat Metode *Clustering* 14
2.3.3 Prinsip Dasar Metode *Clustering* 14
2.3.4 *Centroid Linkage Herarchical Method* (CLHM) 15
2.4 Varian Suatu *Cluster* 15
2.4.1 Analisa *Cluster* 16
2.4.2 Identifikasi Pola Pergerakan Varian 17
2.5 *Hill Climbing* 17
2.6 *Cosine Similarity* 18
2.6.1 Pembobotan Kata (*Term*) 19
2.6.2 *Term Frequency* (TF) 19
2.6.3 *Inverse Document Frequency* (IDF) 20
2.7 Evaluasi Kinerja 21
BAB III METODE PENELITIAN DAN PERANCANGAN 23

3.1 Studi Literatur 24
3.2 Metode Pengumpulan Data 24
3.3 Analisa Kebutuhan Sistem 24
3.4 Perancangan Sistem 25
3.3.1 Arsitektur Sistem 25
3.3.2 Blok Diagram Sistem 26
3.3.3 Diagram Alir Sistem Penentuan Tingkat Plagiarisme Dokumen 26
3.3.4 Perancangan Struktur Data 36
3.3.5 Perancangan *Database* 38
3.3.6 Manualisasi Sistem Penentuan Tingkat Plagiarisme Dokumen 38
3.3.7 Desain *User Interface* Sistem Penentuan Tingkat Plagiarisme Dokumen 54
3.5 Implementasi 55
3.6 Pengujian dan Analisa Sistem 55
3.6.1 Perancangan Uji Coba 56
3.7 Penarikan Kesimpulan 61
BAB IV IMPLEMENTASI 62
4.1 Spesifikasi Sistem 62
4.1.1 Spesifikasi Perangkat Keras 62
4.1.2 Spesifikasi Perangkat Lunak 62
4.2 Batasan-Batasan Implementasi 63
4.3 Implementasi Algoritma 64
4.3.1 Implementasi Algoritma Proses *Input* Dokumen 64
4.3.2 Implementasi Algoritma Proses Pengecekan Kemiripan Dokumen 75
4.4 Implementasi Antar Muka 77
4.4.1 Tampilan Halaman Cek Kemiripan Dokumen 78
4.4.2 Tampilan Halaman Tambah Dokumen 78
4.4.3 Tampilan Halaman Daftar Dokumen 79
BAB V PENGUJIAN DAN ANALISIS 80
5.1 Pengujian Ketepatan *Cluster* 80
5.2 Pengujian *F-measure* 81
5.3 Pengujian *Similarity* Dokumen 82
5.4 Pengujian Prosentase *Error* 86
5.5 Analisa Hasil 91
BAB VI PENUTUP 96
6.1 Kesimpulan 96
6.2 Saran 97
DAFTAR PUSTAKA 98
LAMPIRAN 100

DAFTAR TABEL

Tabel 2. 1 Evaluasi hasil kinerja 21
Tabel 3. 1 Contoh dokumen 39
Tabel 3. 2 Contoh dokumen hasil *tokenizing* 41
Tabel 3. 3 Contoh dokumen hasil *filtering* 42
Tabel 3. 4 Contoh dokumen hasil *stemming* 43
Tabel 3. 5 Informasi dokumen 45
Tabel 3. 6 Hasil TF-IDF pada dokumen 47
Tabel 3. 7 Hasil normalisasi *terms* pada dokumen 49
Tabel 3. 8 Proses *clustering* CLHM 51
Tabel 3. 9 Hasil analisa varian pada *cluster* 52
Tabel 3. 10 Analisa *hill climbing* 53
Tabel 3. 11 Tabel hasil *cosine similarity* pada dokumen 54
Tabel 3. 12 Prosentase kemiripan dokumen 54
Tabel 3. 13 Contoh tabel pengujian proses pembentukan *cluster* 57
Tabel 3. 14 Contoh tabel pengujian *f-measure* 57
Tabel 3. 15 Contoh tabel pengujian *similarity* plagiarisme 58
Tabel 3. 16 Contoh tabel pengujian prosentase *error* 58
Tabel 4. 1 Spesifikasi Perangkat Keras Komputer 62
Tabel 4. 2 Spesifikasi Perangkat Lunak Komputer 63
Tabel 4. 3 Daftar fungsi pada sistem 64
Tabel 5. 1 Hasil pengujian proses pembentukan *cluster* 80
Tabel 5. 2 Hasil pengujian *f-measure* 81
Tabel 5. 3 Hasil pengujian prosentase plagiarisme 83
Tabel 5. 4 Hasil pengujian *prosentase error* tanpa pemotongan kata 86
Tabel 5. 5 Hasil pengujian prosentase *error* dengan pemotongan kata 25% 87
Tabel 5. 6 Hasil pengujian prosentase *error* dengan pemotongan kata 50% 88
Tabel 5. 7 Hasil pengujian prosentase *error* dengan pemotongan kata 75% 90
Tabel 5. 8 Pengaruh pemotongan kata sebesar 25% secara random 94

DAFTAR GAMBAR

Gambar 2. 1 Tahapan *text mining* 8
Gambar 2. 2 Contoh tahap *tokenizing* 9
Gambar 2. 3 Contoh tahapan *filtering* 9
Gambar 2. 4 Contoh tahapan *stemming* 10
Gambar 2. 5 Ilustrasi *centroid linkage* 15
Gambar 2. 6 Pergerakan varian pada tiap tahap pembentukan *cluster* 17
Gambar 2. 7 Pola nilai beda *hill-climbing* 18
Gambar 3. 1 Desain penelitian sistem penentuan tingkat plagiarisme dokumen .. 23
Gambar 3. 2 Arsitektur sistem penentuan tingkat plagiarisme dokumen 25
Gambar 3. 3 Diagram blok sistem penentuan tingkat plagiarisme 26
Gambar 3. 4 Diagram alir sistem 28
Gambar 3. 5 *Flowchart prepocessing* 28
Gambar 3. 6 *Flowchart tokenizing* 29
Gambar 3. 7 *Flowchart filtering* 30
Gambar 3. 8 *Flowchart stemming* Arifin-Setiono 31
Gambar 3. 9 *Flowchart stemming* Arifin-Setiono (lanjutan) 32
Gambar 3. 10 *Flowchart* pembobotan TF-IDF 33
Gambar 3. 11 *Flowchart* CLHM 34
Gambar 3. 12 *Flowchart* varian 35
Gambar 3. 13 *Flowchart hill climbing* 36
Gambar 3. 14 Perancangan struktur data dari dokumen 36
Gambar 3. 15 Ilustrasi dokumen dengan *array* tiga dimensi 37
Gambar 3. 16 Tabel Database sistem penentuan kemiripan 38
Gambar 3. 17 Contoh dokumen uji 39
Gambar 3. 18 Perancangan desain *user interface* sistem 54
Gambar 4. 1 *Form* cek kemiripan dokumen 78
Gambar 4. 2 *Form* tambah dokumen 79
Gambar 4. 3 *Form* lihat daftar dokumen 79
Gambar 5. 1 Grafik prosentase *error* dengan pemotongan kata 25% 88
Gambar 5. 2 Grafik prosentase *error* dengan pemotongan kata 50% 89
Gambar 5. 3 Grafik prosentase *error* dengan pemotongan kata 75% 91
Gambar 5. 4 Grafik nilai *f-measure* sistem 92
Gambar 5. 5 Grafik rata-rata *similarity* 93
Gambar 5. 6 Grafik rata-rata prosentase *error* 94

DAFTAR *SOURCE CODE*

Sourcecode 4. 1 Implementasi algoritma *text mining* pada dokumen..... 65
Sourcecode 4. 2 Implementasi algoritma *tokenizing*..... 66
Sourcecode 4. 3 Implementasi algoritma *stopword removal*..... 67
Sourcecode 4. 4 Implementasi algoritma *update* nilai tf dan df pada dokumen.. 68
Sourcecode 4. 5 Implementasi algoritma pembobotan TF-IDF pada dokumen .. 68
Sourcecode 4. 6 Implementasi algoritma normalisasi dokumen..... 69
Sourcecode 4. 7 Implementasi algoritma struktur data *pre-clustering* 70
Sourcecode 4. 8 Implementasi algoritma *clustering* menggunakan CLHM..... 72
Sourcecode 4. 9 Implementasi algoritma proses analisa *cluster*..... 74
Sourcecode 4. 10 Implementasi algoritma proses *hill climbing* 75
Sourcecode 4. 11 Implementasi algoritma proses pengecekan kemiripan dokumen 77

BAB I
PENDAHULUAN

1.1 Latar Belakang

Seiring dengan meluasnya penggunaan internet untuk media pembelajaran dan penelitian, kesempatan untuk melakukan plagiarisme meningkat. Begitu juga dalam pembuatan penelitian khususnya tugas akhir atau skripsi, mahasiswa sering melakukan *copy and paste*. Mengambil atau mengacu tulisan atau produk pihak lain diperbolehkan asalkan menganut kaidah yang berlaku. Tindakan pengambilan tulisan ide pihak lain yang melanggar kaidah disebut plagiarisme. Tindakan tersebut perlu diperkecil. Tindakan plagiarisme diduga sering ditemukan pada hasil karya tulis mahasiswa. Tindakan plagiarisme dapat diatasi dengan mendeteksi kemiripan dokumen penelitian yang efektif dan efisien untuk mengecek kesamaan topik pada dokumen.

Sistem pendeteksian plagiarisme dokumen penelitian merupakan salah satu solusi yang dapat dilakukan sehingga tindakan curang dapat diminimalisasi. Sistem pendeteksi plagiarisme dokumen dapat digunakan untuk mengukur kemiripan tugas akhir atau skripsi di lingkungan akademik. Saat ini sudah banyak aplikasi dan layanan yang tersedia untuk mendeteksi kemiripan dokumen. Aplikasi deteksi plagiarisme yang dibangun terdiri dari berbagai metode diantaranya metode *Running Kap Robin Matching and Greedy String Tiling*. (RKR-GST), dan algoritma *winnowing* [KUR-08]. Selain itu, metode *single linkage hierarchical method* (SLHM) yang digunakan untuk menentuan kemiripan topik proyek akhir berdasarkan abstrak [VID-10]. Berdasarkan paparan tersebut, penulis melakukan penelitian yang serupa yaitu "penentuan plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM)". Metode *Centroid Linkage Hierarchical* (CLHM) pada penelitian sebelumnya digunakan untuk temu kembali informasi berdasarkan lokasi pada dokumen yang dikelompokkan [DAM-11].

Penelitian ini membahas tentang penentuan tingkat plagiarisme dokumen penelitian dengan membandingkan judul, abstrak, dan kata kunci yang ada pada

abstrak. Metode yang digunakan dalam penentuan tingkat plagiarisme dokumen adalah *Centroid Linkage Hierarchical Method* (CLHM). CLHM merupakan metode *hierarchical clustering*. *Clustering* dokumen merupakan salah satu metode untuk pengelompokan dokumen dengan menemukan keterkaitan antar dokumen. Keterkaitan antar dokumen dapat diketahui berdasarkan jarak *centroid* masing-masing data yang direpresentasikan dengan titik-titik pada bidang koordinat [ERN-09]. Sehingga, kemiripan antar dokumen dapat diketahui dengan *cluster* yang telah terbentuk berdasarkan *term* yang dimiliki tiap dokumen. *Cluster* yang terbentuk dapat mengecek kemiripan satu dokumen yang dibandingkan dengan dokumen yang terdapat pada *cluster* tersebut.

Sistem penentuan tingkat plagiarisme ini terdiri dari beberapa proses, yaitu tahap *text mining*, dan *Centroid Linkage Hierarchical Method* (CLHM), *hill climbing*, dan *cosine similarity*. Tahap *text mining* berfungsi untuk menghilangkan kata yang tidak berguna dan mendapatkan kata dasar. Tahap selanjutnya adalah pembobotan TF-IDF, dan klusterisasi menggunakan *Centroid Linkage Hierarchical Method* (CLHM). Setelah didapatkan *cluster* dari CLHM, dilanjutkan dengan penerapan metode *hill climbing*. Metode *hill climbing* menentukan jumlah *cluster* yang optimal. Proses terakhir dalam sistem ini adalah proses penentuan tingkat plagiarisme menggunakan *cosine similarity*. Hasil yang ditampilkan pada penelitian ini berupa hasil *cluster* dan prosentase masing – masing dokumen. Metode ini diharapkan dapat membantu dan mempermudah pengguna untuk mengetahui tingkat plagiarisme dokumen penelitian, khususnya skripsi atau tugas akhir.

1.2 Rumusan Masalah

Dari uraian yang telah disampaikan pada latar belakang, maka dapat diambil beberapa rumusan masalah diantaranya :

a. Bagaimana penerapan sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM) dan *cosine similarity*.
b. Bagaimana tingkat akurasi penentuan plagiarisme dokumen penelitian metode *Centroid Linkage Hierarchical Method* (CLHM) dan *cosine similarity*.

1.3 Batasan Masalah

Berdasarkan latar belakang dan rumusan masalah yang telah dikemukakan, penelitian ini mempunyai batasan-batasan masalah sebagai berikut:

1. Dokumen yang digunakan adalah skripsi atau tugas akhir.
2. Bagian dokumen yang digunakan adalah judul, abstrak dan *keyword*.
3. Dokumen yang digunakan merupakan dokumen berbahasa Indonesia.
4. File yang digunakan dalam aplikasi ini bertipe data *Text File* (.txt).
5. Hasil penentuan tingkat plagiarisme dokumen terhadap sekelompok dokumen yang telah terbentuk.
6. Sistem penentuan tingkat plagiarisme ini berdasarkan pada perbandingan kemunculan kata dokumen, bukan berdasarkan pada perbandingan analisis *semantic*.
7. Pembentukan *cluster* otomatis menggunakan analisa varian dan *hill climbing*.

1.4 Tujuan

Tujuan yang ingin dicapai dalam pembuatan tugas akhir ini adalah menerapkan dan menguji sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM). Sistem ini membandingkan kemiripan topik penelitian (tugas akhir atau skripsi) berdasarkan judul, abstrak, dan *keyword*. Sehingga dengan membandingkan kemiripan antar dokumen dapat diketahui tingkat plagiarisme suatu dokumen.

1.5 Manfaat

Penelitian ini diharapkan dapat bermanfaat untuk semua pihak khususnya di ruang lingkup akademik. Adapun manfaat yang diharapkan adalah sebagai berikut:

1. Dapat membantu pengguna sebagai bahan pertimbangan dalam menentukan kemiripan pada dokumen penelitian, misalnya skripsi atau tugas akhir.
2. Mengetahui prosentase kemiripan dokumen penelitian sehingga dapat digunakan sebagai bahan pertimbangan untuk mendeteksi adanya plagiarisme.

1.6 Sistematika Penyusunan Laporan

Sistematika penulisan penelitian ditunjukkan untuk memberikan gambaran dan uraian dari penyusunan tugas akhir secara garis besar yang meliputi beberapa bab, sebagai berikut.

BAB I : Pendahuluan

Menguraikan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat dan sistematika penulisan.

BAB II : Tinjauan Pustaka

Menguraikan tentang kajian pustaka dan dasar teori yang mendasari pembuatan sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM).

BAB III : Metodologi dan Perancangan

Metodologi menguraikan tentang metode dan langkah kerja yang dilakukan dalam penulisan tugas akhir yang terdiri dari studi literatur, metode pengambilan data, analisis kebutuhan, perancangan sistem, implementasi, pengujian dan analisis serta pengambilan kesimpulan. Sedangkan perancangan berisi tentang perencanaan aplikasi yang dibuat, meliputi deskripsi aplikasi, spesifikasi kebutuhan, dan perancangan aplikasi sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM).

BAB IV : Implementasi

Membahas implementasi dari sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM) yang sesuai dengan perancangan sistem yang telah dibuat.

BAB V : Pengujian dan analisis

Memuat hasil pengujian dan analisis terhadap penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM) yang telah direalisasikan.

BAB VI : Penutup

Pada bab ini berisi kesimpulan yang diambil berdasarkan analisa sistem setelah pengujian, kelebihan atau kekurangan, serta saran–saran untuk penyempurnaan aplikasi yang dibuat.

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

Bab ini membahas teori yang diperlukan untuk menyusun penelitian. Bab ini terdiri dari kajian pustaka dan dasar teori. Kajian pustaka membahas penelitian yang telah ada. Penyusunan dasar teori berdasarkan latar belakang dan rumusan masalah adalah *text mining*, *clustering*, *hill climbing*, *cosine similarity*, dan evaluasi kinerja sistem. *Text mining* atau biasa disebut *preprocessing* akan dibahas tentang *tokenizing, filtering, stemming, analizing*. Teori yang selanjutnya akan menjabarkan tentang karakteristik *clustering*, manfaat *clustering*, prinsip dasar *clustering*, *clustering* menggunakan *centroid linkage hierarchical* (CLHM), dan analisa *cluster*.

2.1 Kajian Pustaka

Kajian pustaka pada penelitian ini membahas penelitian sebelumnya yang berjudul 'Penentuan Kemiripan Topik Proyek Akhir Berdasarkan Abstrak Pada Jurusan Teknik Informatika Menggunakan Metode *Single Linkage Hierarchical*'. Penelitian ini membahas tentang penentuan kemiripan topik proyek akhir berdasarkan abstrak pada dokumen berbahasa Indonesia. Metode yang digunakan adalah *Single Linkage Hierarchical Method* (SLHM). SLHM digunakan untuk mengelompokkan dokumen menjadi tujuh *cluster*. Penentuan kemiripannya menggunakan *inner product*.

Perbedaan yang dibuat pada penelitian ini adalah pada penggunaan metode untuk mengelompokkan dokumen-dokumen dan proses pengecekan kemiripannya. Untuk mengelompokkan dokumen digunakan *Centroid Linkage Hierarchical Method* (CLHM) dan untuk menghitung kemiripan antar dokumen digunakan *cosine similarity*. Metode CLHM sebelumnya diterapkan pada '*Temu Kembali Informasi Berdasarkan Lokasi pada Dokumen yang Dikelompokkan Menggunakan Metode Centroid Linkage Hierarchical'* [DAM-11]. Penelitian ini menggabungkan CLHM dengan analisa *hill climbing* agar terbentuk *cluster* secara otomatis, sehingga jumlah *cluster* yang dihasilkan pada penelitian ini dinamis. Hal

ini dikarenakan penambahan analisa varian *hill climbing* setelah proses *clustering*. Hal ini yang membedakan dengan penelitian sebelumnya.

Proses pengecekan kemiripan dokumen pada penelitian ini menggunakan *cosine similarity*. *Cosine similarity* dapat menghitung sudut antara vektor dokumen dengan vektor *query*. *Inner product* memiliki kelemahan yaitu dokumen yang panjang cenderung akan memiliki koefisien kesamaan yang tinggi karena peluang *term* sesuai *query* dan dokumen cukup tinggi [SAL-12]. Sedangkan kelebihan utama dari metode *cosine similarity* adalah tidak terpengaruh pada panjang pendeknya suatu dokumen karena yang diperhitungkan hanya nilai *term* dari masing-masing dokumen.

2.2 *Text Mining*

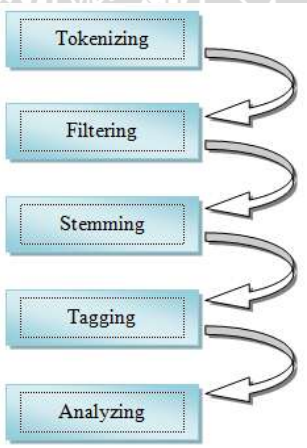
Banyaknya data yang ada terkadang membuat kita kesulitan untuk menemukan substansi informasi dari data itu sendiri. Banyak hal telah dilakukan untuk menangani data sedemikian sehingga *user* lebih mudah untuk menemukan informasi yang mereka butuhkan dengan tepat dan dalam waktu yang efisien [ERN-09]. Salah satu teknik untuk menemukan informasi pada konsep data dalam *database* dikenal dengan istilah *data mining.*

Salah satu bagian dari *data mining* yang cukup menarik adalah *text mining*. Metode ini digunakan untuk menggali informasi dari data-data dalam bentuk teks seperti buku, makalah, paper, dan lain sebagainya. Yang membedakan *data mining* dengan *text mining* adalah proses analisis terhadap suatu data. *Data Mining* atau KDD (*Knowledge Discovery in Databases*) adalah proses untuk menemukan pengetahuan dari sejumlah besar data yang disimpan baik di dalam *databases, data warehouses* atau tempat penyimpanan informasi lainnya. Sedangkan untuk *text mining* sering disebut dengan *Keyword-Based Association Analysis. Keyword-Based Association Analysis* merupakan sebuah analisa yang mengumpulkan *keywords* atau *terms* (istilah) yang sering muncul secara bersamaan dan kemudian menemukan hubungan asosiasi dan korelasi di antara *keywords* atau *terms* itu, lihat Tan (1999) [KUR-09].

Text mining adalah proses penemuan akan informasi yang sebelumnya tidak terungkap dengan memproses dan menganalisa data dalam jumlah besar.

Dalam menganalisa sebagian atau keseluruhan *unstructured text*, *text mining* mencoba untuk mengasosiasikan satu bagian teks dengan yang lainnya berdasarkan aturan-aturan tertentu. Hasil yang diharapkan adalah informasi baru atau *insight* yang tidak terungkap jelas sebelumnya [ERN-09].

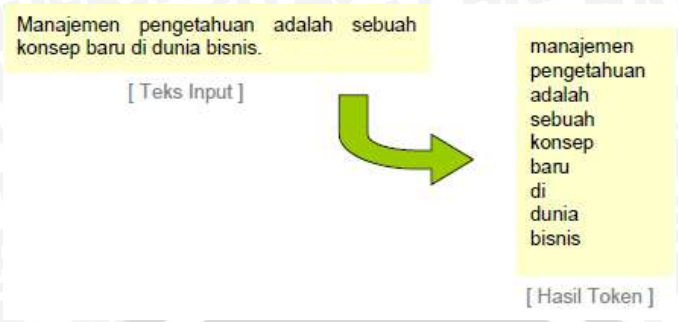
Tujuan utama *text mining* adalah mendukung proses *knowledge discovery* pada koleksi dokumen yang besar. Pada prinsipnya, *text mining* adalah bidang ilmu *multidiscipline*, melibatkan *information retrieval* (IR), *text analysis*, *information extraction* (IE), *clustering*, *categorization*, *visualization*, *database technology*, *natural language processing* (NLP), *machine learning*, dan *data mining*. Dapat pula dikatakan bahwa *text mining* merupakan salah satu bentuk aplikasi kecerdasan buatan (*artificial intelligence*) [SAL-12]. Proses *text mining* dilakukan beberapa tahapan secara umum yang ditunjukkan pada gambar 2.1.



Gambar 2. 1 Tahapan *text mining*
Sumber : [SAL-12]

2.2.1 *Tokenizing*

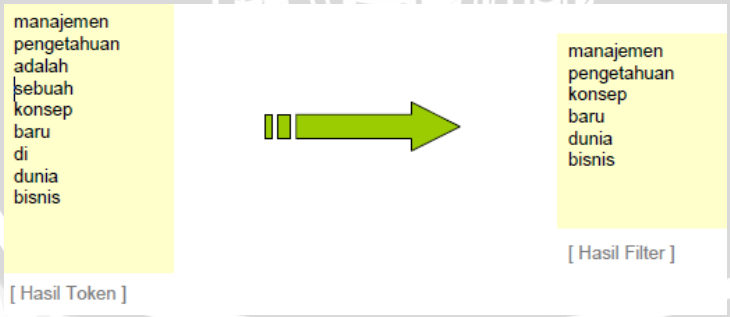
Tahap *tokenizing* adalah tahap pemotongan *string* berdasarkan tiap kata yang menyusunnya. *Tokenizing* merupakan proses penghilangan tanda baca pada kalimat yang ada dalam dokumen sehingga menghasilkan kata-kata yang berdiri sendiri [SUS-12].



Gambar 2. 2 Contoh tahap *tokenizing*
Sumber: [SAL-12]

2.2.2 *Filtering*

Tahap *filtering* adalah tahap pengambilan kata – kata yang penting dari hasil *tokenizing*. Tahap *filtering* ini dapat mengunakan algoritma *stoplist* atau *wordlist*. *Stoplist* / *stopword* yaitu *filtering* terhadap kata – kata yang tidak layak untuk dijadikan sebagai *keyword* sehingga kata – kata tersebut dapat dihilangkan. Sedangkan *wordlist* adalah daftar kata – kata yang mungkin digunakan sebagai *keyword*, dengan demikian maka tertentu jumlah kata termasuk dalam *wordlist* akan lebih banyak daripada *stoplist*. Contoh *stopword* adalah "yang", "dan", "di", "dari" dan seterusnya. Pada proses *filtering* ini jika isi teks berisi kata sambung, kata depan, nama hari, nama bulan, nama tempat, serta tanda titik, koma, kurung buka, kurung tutup, tanda tanya, tanda seru, dan spasi maka dihilangkan, sehingga proses filtering ini akan menghasilkan kata – kata yang penting saja [SUS-12].

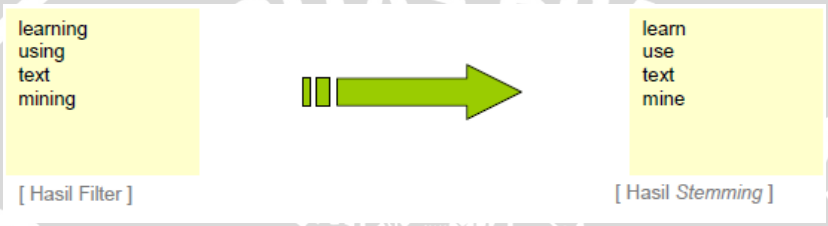


Gambar 2. 3 Contoh tahapan *filtering*
Sumber : [SAL-12]

2.2.3 *Stemming*

Tahap *stemming* adalah tahap mencari *root* kata dari tiap kata hasil *filtering*. *Stem* (akar kata) adalah bagian dari kata yang tersisa setelah dihilangkan

imbuhannya (awalan dan akhiran). Metode *stemming* memerlukan *input* berupa *term* yang terdapat dalam dokumen. Sedangkan *output* berupa *stem*. Setiap kata yang memiliki imbuhan seperti imbuhan awalan dan akhiran maka akan diambil kata dasarnya. Pada tahap ini dilakukan proses pengembalian berbagai bentukan kata ke dalam representasi yang sama. Tahap ini kebanyakan dipakai untuk teks bahasa Inggris dan lebih sulit diterapkan pada teks berbahasa Indonesia. Hal ini dikarenakan bahasa Indonesia tidak memiliki rumus bentuk baku yang permanen [SUS-12].



Gambar 2. 4 Contoh tahapan *stemming*
Sumber : [SAL-12]

2.1.3.1 *Stemming* Arifin dan Setiono

Algoritma ini didahului dengan pembacaan tiap kata dari file sampel. Sehingga *input* dari algoritma ini adalah sebuah kata yang kemudian dilakukan :

1. Pemeriksaan semua kemungkinan bentuk kata. Setiap kata diasumsikan memiliki 2 Awalan (prefiks) dan 3 Akhiran (sufiks). Sehingga bentuknya menjadi :

   Prefiks 1 + Prefiks 2 + Kata dasar + Sufiks 3 + Sufiks 2 + Sufiks 1

   Seandainya kata tersebut tidak memiliki imbuhan sebanyak imbuhan di atas, maka imbuhan yang kosong diberi tanda x untuk prefiks dan diberi tanda xx untuk sufiks.
2. Pemotongan dilakukan secara berurutan sebagai berikut :

   AW : AW (Awalan)

   AK : AK (Akhiran)

   KD : KD (Kata Dasar)

   a. AW I, hasilnya disimpan pada p1 (prefiks 1)
   b. AW II, hasilnya disimpan pada p2 (prefiks 2)
   c. AK I, hasilnya disimpan pada s1 (sufiks 1)

d. AK II, hasilnya disimpan pada s2 (sufiks 2)

e. AK III, hasilnya disimpan pada s3 (sufiks 3)

Pada setiap tahap pemotongan di atas diikuti dengan pemeriksaan di kamus apakah hasil pemotongan itu sudah berada dalam bentuk dasar. Kalau pemeriksaan ini berhasil maka proses dinyatakan selesai dan tidak perlu melanjutkan proses pemotongan imbuhan lainnya. Contoh pemenggalan kata "mempermainkannya"

a. Langkah 1 :

Cek apakah kata ada dalam kamus

Ya : Success

Tidak : lakukan pemotongan AW I

Kata = permainkannya

b. Langkah 2 :

Cek apakah kata ada dalam kamus

Ya : Success

Tidak : lakukan pemotongan AW II

Kata = mainkannya

c. Langkah 3 :

Cek apakah kata ada dalam kamus

Ya : Success

Tidak : lakukan pemotongan AK I

Kata = mainkan

d. Langkah 4 :

Cek apakah kata ada dalam kamus

Ya : Success

Tidak : lakukan pemotongan AK II

Kata = main

e. Langkah 5 :

Cek apakah kata ada dalam kamus

Ya : Success

Tidak : lakukan pemotongan AK III. Dalam hal ini AK III tidak ada, sehingga kata tidak diubah.

Kata = main

f. Langkah 6

Cek apakah kata ada dalam kamus

Ya : Success

Tidak : "Kata tidak ditemukan"

3. Namun jika sampai pada pemotongan AK III, belum juga ditemukan di kamus, maka dilakukan proses kombinasi. KD yang dihasilkan dikombinasikan dengan imbuhan-imbuhannya dalam 12 konfigurasi berikut:
   a. KD
   b. KD + AK III
   c. KD + AK III + AK II
   d. KD + AK III + AK II + AK I
   e. AW I + AW II + KD
   f. AW I + AW II + KD + AK III
   g. AW I + AW II + KD + AK III + AK II
   h. AW I + AW II + KD + AK III + AK II + AK I
   i. AW II + KD
   j. AW II + KD + AK III
   k. AW II + KD + AK III + AK II
   l. AW II + KD + AK III + AK II + AK I

Sebenarnya kombinasi a, b, c, d, h, dan l sudah diperiksa pada tahap sebelumnya, karena kombinasi ini adalah hasil pemotongan bertahap tersebut. Dengan demikian, kombinasi yang masih perlu dilakukan tinggal 6 yakni pada kombinasi-kombinasi yang belum dilakukan (e, f, g, i, j, dan k). Tentunya bila hasil pemeriksaan suatu kombinasi adalah 'ada', maka pemeriksaan pada kombinasi lainnya sudah tidak diperlukan lagi.

Pemeriksaan 12 kombinasi ini diperlukan, karena adanya fenomena *overstemming* pada algoritma pemotongan imbuhan. Kelemahan ini berakibat pada pemotongan bagian kata yang sebenarnya adalah milik kata dasar itu sendiri yang kebetulan mirip dengan salah satu jenis imbuhan yang ada. Dengan 12 kombinasi itu, pemotongan yang sudah terlanjur tersebut dapat dikembalikan sesuai posisinya [ARI-01].

2.2.4 *Analizing*

Tahap *analizing* merupakan tahap penentuan seberapa jauh keterhubungan antar kata – kata antar dokumen yang ada. Tahap *analizing* merupakan tahap terakhir dari *teks mining*.

2.3 *Clustering*

Clustering didefinisikan sebagai upaya pengelompokan data ke dalam *cluster* sehingga data-data didalam *cluster* yang sama memiliki lebih kesamaan dibandingkan dengan data-data pada *cluster* yang berbeda. Dikenal dua pendekatan, yaitu *hierarchical* dan *partisional* dengan masing-masing memiliki banyak variasi [HAM-07].

Clustering biasa digunakan pada banyak bidang, seperti : *data mining*, *pattern recognition* (pengenalan pola), *image classification* (pengklasifikasian gambar), ilmu biologi, pemasaran, perencanaan kota, pencarian dokumen, dan lain sebagainya [NOO-09].

2.3.1 Karakteristik *Clustering*

Karakteristik *clustering* dibagi menjadi empat, yaitu :

a. Partitioning clustering

Partitioning clustering disebut juga *exclusive clustering*, dimana setiap data harus termasuk ke *cluster* tertentu. Karakteristik tipe ini juga memungkinkan bagi setiap data yang termasuk *cluster* tertentu pada suatu tahapan proses, pada tahapan berikutnya berpindah ke *cluster* yang lain.

Contoh : *K-Means, residual analysis*.

b. Hierarchical clustering

Pada *hierarchical clustering*, setiap data harus termasuk ke *cluster* tertentu. Dan suatu data yang termasuk ke *cluster* tertentu pada suatu tahapan proses, tidak dapat berpindah ke *cluster* lain pada tahapan berikutnya.

Contoh: *Single Linkage, Centroid Linkage, Complete Linkage, Average Linkage.*

c. Overlapping clustering

Dalam *overlapping clustering*, setiap data memungkinkan termasuk ke beberapa *cluster*. Data mempunyai nilai keanggotaan (*membership*) pada beberapa *cluster*.

Contoh: *Fuzzy C-means, Gaussian Mixture*.

d. Hybrid

Karakteristik *hybrid* adalah menyatukan karakteristik dari *partitioning, overlapping* dan *hierarchical*.

2.3.2 Manfaat Metode *Clustering*

Manfaat dari metode *clustering* antara lain yaitu [SAL-12]:

a. Identifikasi objek (*Recognition*)

Identifikasi objek mempelajari bagaimana komputer dapat mengenali objek yang diamati dan diobservasi. Proses ini bertujuan untuk mengekstrak informasi atau pesan yang disampaikan oleh objek. Ciri – ciri yang dimiliki oleh suatu pola ditentukan oleh distribusi statiknya, dimana objek yang berbeda memiliki objek yang berbeda pula. Ciri – ciri inilah yang akan digunakan untuk membedakan suatu objek dengan objek lainnya.

b. Decission Support System (DSS) dan *data mining*

Decission support system dan data mining merupakan suatu sistem yang berfungsi sebagai penunjang keputusan, dengan adanya *decission support system* dan *data mining*, pekerjaan dari para pengambil keputusan akan lebih terbantu secara signifikan.

2.3.3 Prinsip Dasar Metode *Clustering*

Pada proses *clustering* terdapat beberapa prinsip dasar yaitu:

a. *Similarity Measures* (ukuran kedekatan).
b. *Distances* dan *Similarity Coeficients* untuk beberapa sepasang dari item.
c. *Ecluidean Distance*.

Tujuan utama teknik ini adalah melakukan pengelompokan berdasarkan kriteria tertentu sehingga objek – objek tersebut mempunyai variasi di dalam *cluster* relatif kecil dibandingkan variasi antar *cluster* [SAL-12].

2.3.4 *Centroid Linkage Herarchical Method* (CLHM)

Centroid Linkage Herarchical Method (CLHM) adalah proses pembentukan *cluster* yang didasarkan pada jarak antar centroidnya. Metode ini bagus untuk memperkecil *variance within cluster* karena melibatkan *centroid* pada saat penggabungan antar *cluster*. Metode ini juga baik untuk data yang mengandung *outlier* [NOO-09].

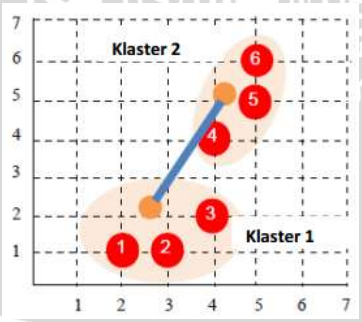
Algoritma *Centroid Linkage Herarchical Method* (CLHM) sebagai berikut [NOO-09]:

1. Diasumsikan setiap data dianggap sebagai *cluster*.
   Jika n = jumlah data dan c = jumlah *cluster*, berarti ada c = n.
2. Menghitung jarak antar *cluster* dengan *euclidian distance*.
3. Mencari 2 *cluster* yang mempunyai jarak *centroid* antar *cluster* yang paling minimal dan digabungkan (*merge*) kedalam *cluster* baru (sehingga C= c-1).
4. Kembali ke langkah 3, dan diulangi sampai dicapai *cluster* yang diinginkan.

Perhitungan jarak antar objek, maupun antar klasternya dilakukan dengan *euclidian distance*, khususnya untuk data *numeric*. Untuk data 2 dimensi digunakan persamaan 2.1

$$d\,(x,y) = \sqrt{\sum_{i=n}^{n} |\, x_i - y_i \,|^{\,2}} \qquad (2.1)$$

Jarak *cluster* 1 ke *cluster* 2 sama dengan jarak centroid *cluster* 1 ke centroid *cluster* 2. Ilustrasi *Centroid linkage hierarchical* ditunjukkan oleh gambar 2.5.



Gambar 2. 5 Ilustrasi *centroid linkage*
Sumber : [NOO-09]

2.4 Varian Suatu *Cluster*

Varian suatu *cluster* adalah seperangkat teknik yang dipergunakan untuk mendapatkan jumlah *cluster* yang tepat secara otomatis [ELD-10].

2.4.1 Analisa *Cluster*

Analisa *cluster* bisa diperoleh dari kepadatan *cluster* yang dibentuk (*cluster dencity*). Kepadatan suatu *cluster* dapat ditentukan dengan *variance within cluster* (V_w) dan *variance between cluster* (V_b) [MAR-10]. Varian tiap tahap pembentukan *cluster* dihitung menggunakan persamaan 2.2.

$$Vc^2 = \frac{1}{n_c - 1} \sum_{i=1}^{c} (y_i - \bar{y_c})^2 \quad (2.2)$$

Keterangan :

V_c^2 = varian pada *cluster* c

c = 1 .. k , dimana k = jumlah *cluster*

n_c = jumlah data pada *cluster* c

y_i = data ke-i pada suatu *cluster*

y_c = rata – rata dari data pada suatu *cluster*

Selanjutnya dari nilai varian tersebut dihitung nilai *variance within cluster* (V_w) seperti persamaan 2.3 , sedangkan nilai *variance between cluster* (V_b) dihitung menggunakan persamaan 2.4.

$$Vw = \frac{1}{N-k} \sum_{i=1}^{k} (n_i - 1).V_i^2 \quad (2.3)$$

$$Vb = \frac{1}{k-1} \sum_{i=1}^{k} n_i.(\bar{y_i} - \bar{y})^2 \quad (2.4)$$

Keterangan :

N = jumlah semua data

k = jumlah *cluster*

n_i = jumlah data *cluster* ke-i

V_i^2 = varian pada *cluster* ke-i

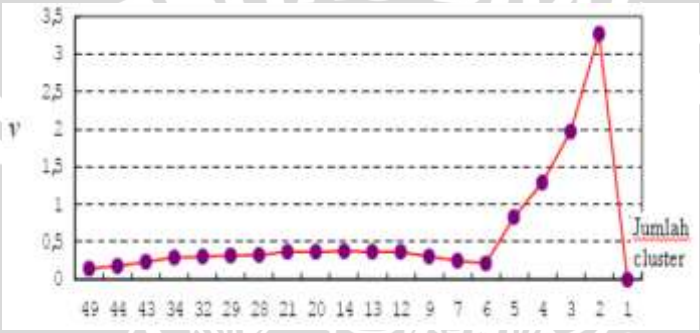
$\bar{y}$ = rata-rata dari y_i

Salah satu metode yang digunakan untuk menentukan *cluster* yang ideal adalah batasan varian, yaitu dengan menghitung kepadatan *cluster* berupa *variance within cluster* (V_w) dan *variance between cluster* (V_b). *Cluster* yang ideal mempunyai V_w minimum yang merepresentasikan *internal homogenity* dan maksimum V_b yang menyatakan *external homogenity* [MAR-10]. Nilai varian dihitung menggunakan persamaan 2.5.

$$V = \frac{V_w}{V_b} \quad (2.5)$$

2.4.2 Identifikasi Pola Pergerakan Varian

Identifikasi pola pergerakan varian merupakan metode untuk memperoleh *cluster* yang mencapai *global optimum*, yang mampu mengatasi masalah dari minimum V. Gambar 2.6 menunjukkan pergerakan varian pada tiap tahap pembentukan *cluster*, dimana dari gambar tersebut terlihat bahwa global optimum berada pada tahap ke 15, dengan enam total *cluster* [ELD-10].



Gambar 2. 6 Pergerakan varian pada tiap tahap pembentukan *cluster*
Sumber: [ELD-10]

Berikut tahap untuk menemukan *global optimum* dari tahap pembentukan *cluster* [ELD-10]:

a. Mendeskripsikan semua pola dari pergerakan varian, seperti gambar 2.6.
b. Menganalisa kemungkinan *global optimum* yang berada pada tempat yang tepat.
c. Melihat posisi dari *global optimum* yang mungkin.

Posisi yang mungkin untuk menemukan *global optimum* pada pergerakan varian, dikelompokkan menjadi dua, yaitu *Hill Climbing* dan *Valley Tracing*.

2.5 *Hill Climbing*

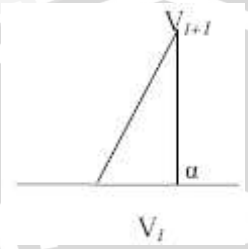
Pada *Hill-climbing* didefinisikan bahwa kemungkinan mencapai *global optimum* terletak pada tahap ke-i, jika memenuhi persamaan 2.5.

$$V_{i+1} > \alpha . V_i \quad (2.5)$$

Keterangan :

α : nilai tinggi

Nilai tinggi digunakan untuk menentukan seberapa mungkin metode ini mencapai *global optimum*. Nilai α yang biasa digunakan adalah 2, 3, dan 4 [ELD-10]. Persamaan diatas, diperoleh berdasarkan analisa pergerakan varian pola *Hill Climbing* yang ditunjukkan pada gambar 2.7.



Gambar 2. 7 Pola nilai beda *hill-climbing*
Sumber : [ELD-10]

Selanjutnya, dengan pendekatan metode *hill climbing* dilakukan identifikasi perbedaan nilai tinggi (∂) pada tiap tahap, yang didefinisikan dengan :

$$\partial = V_{i-1} - (V_i * \alpha) \quad (2.6)$$

Nilai ∂ digunakan untuk menghindari *local optima*, dimana persamaan ini diperoleh dari maksimum ∂ yang dipenuhi pada persamaan 2.6. Untuk mengetahui keakuratan dari suatu metode pembentukan *cluster* pada *hierarchical method*, dengan menggunakan *hill climbing* digunakan persamaan 2.7.

$$\varphi = \frac{\max(\partial)}{\text{nilai terdekat ke } \max(\partial)} \quad (2.7)$$

Dimana nilai terdekat ke max(∂) adalah nilai kandidat max(∂) sebelumnya. Nilai φ yang lebih besar atau sama dengan 2 (φ≥2), menunjukkan *cluster* yang terbentuk merupakan *cluster* yang *well-separated* (terpisah dengan baik) [ELD-10].

2.6 *Cosine Similarity*

Metode *cosine similarity* adalah metode untuk menghitung kesamaan dari dua dokumen. *Cosine similarity* dapat menyamakan frekuensi kata pada kalimat mengunakan persamaan *Term Frequency* (TF). *Term frequency* mengekstrak dokumen menjadi proses yang terdiri dari kumpulan kata perkalimat. Tujuannya adalah menyamakan kedua kalimat pada suatu dokumen yang nantinya akan

dibandingkan, sehingga dapat melangkah ke tahap selanjutnya yaitu tahapan *similarity* [RIZ-12].

Penentuan kesesuaian dokumen dengan *query* dipandang sebagai pengukuran (*similarity measure*) antara vektor dokumen (D) dengan vektor *query* (Q). Semakin sama suatu vektor dokumen dengan vektor *query* maka dokumen dapat dipandang semakin sesuai dengan *query* [SUS-12]. Persamaan 2.8 digunakan untuk menghitung *cosine similarity*.

$$CosSim\left(d_j, q\right) = \frac{d_j . q}{|d_j| . |q|} = \frac{\sum_{i=1}^{t} \left(W_{ij} . W_{iq}\right)}{\sqrt{\sum_{i=1}^{t} {W_{ij}}^2 \cdot \sum_{i=1}^{t} {W_{ig}}^2}} \quad (2.8)$$

Keterangan:

t = kata di *database*

d = dokumen

q = kata kunci

Kedekatan *query* dan dokumen diindikasikan dengan sudut yang dibentuk. Nilai *cosinus* yang cenderung besar mengindikasikan bahwa dokumen cenderung sesuai *query* [SUS-12]. Dalam proses membandingkan dokumen yang sesuai dengan dokumen yang telah ada atau dokumen lainnya, maka digunakan perhitungan dengan rumus pada persamaan 2.8 untuk mengetahui angka similaritas dari dokumen tersebut.

2.6.1 Pembobotan Kata (*Term*)

Term merupakan kata yang memiliki arti yang terdapat pada dokumen dan *query*. Dari semua kata yang ada, kata - kata merupakan kata umum dihilangkan pada proses *filtering* sehingga kata yang tersisa benar - benar berhubungan dengan isi dokumen. Kumpulan kata yang tersisa itulah yang disebut dengan *term*. Salah satu cara untuk memberi bobot terhadap suatu kata adalah memberi nilai jumlah kemunculan suatu kata adalah memberikan nilai kesesuaian yang semakin besar [SAL-12].

2.6.2 *Term Frequency* (TF)

Term Frequency (TF) adalah jumlah kemunculan sebuah *term* pada sebuah dokumen. Jika sebuah *term* t sering muncul pada sebuah dokumen, maka *query*

yang mengandung t harus mendapatkan dokumen tersebut. *Term frequency* ini didasari pada aspek lokal pada TF-IDF. *Local weight*, atau yang biasa disebut dengan TF (*term frequency*) berfungsi untuk menentukan bobot dari *term* t pada dokumen tertentu, yang pada dasarnya menghasilkan estimasi berdasarkan frekuensi atau *relative frequency* dari term t pada dokumen tersebut [KAO-07].

Ada empat cara yang bisa digunakan untuk mendapatkan nilai TF yaitu [SAL-12] :

1. *Raw* TF

   Pada *Raw TF* ini, nilai TF sebuah *term* dihitung berdasarkan kemunculan *term* tersebut dalam dokumen.

2. *Logarithmic* TF

   Dalam memperoleh nilai TF, cara ini menggunakan fungsi logaritmik pada matematika.

   $$tf = 1 + \log(tf) \quad (2.9)$$

   Keterangan : tf adalah kemunculan kata pada dokumen.

3. *Binary* TF

   Cara ini akan menghasilkan nilai boolean berdasarkan kemunculan *term* pada dokumen tersebut. Bernilai 0 apabila *term* tidak ada pada sebuah dokumen, dan bernilai 1 apabila *term* tersebut ada dalam dokumen. Sehingga banyaknya kemunculan *term* pada sebuah dokumen tidak berpengaruh.

4. *Augmented* TF

   $$tf = 0{,}5 + 0{,}5\ x\ \frac{tf}{\max(tf)} \quad (2.10)$$

   Keterangan:

   - Nilai *tf* adalah jumlah kemunculan *term* pada sebuah dokumen.
   - Nilai max (*tf*) adalah jumlah kemunculan terbanyak *term* pada dokumen yang sama.

2.6.3 *Inverse Document Frequency* (IDF)

Inverse Document Frequency (IDF) adalah jumlah dokumen yang mengandung sebuah *term* yang dicari dari kumpulan dokumen yang ada. IDF ini didasari pada aspek global pada TF-IDF [SAL-12]. *Global weight* atau biasa yang

disebut dengan istilah IDF mendefinisikan kontribusi dari term t ke dokumen tertentu dalam sebuah *global sense* [KAO-07].

$$idf = \log\ (\frac{N}{df}) \quad (2.11)$$

Keterangan:

N : jumlah dokumen yang terdapat pada kumpulan dokumen

df : jumlah dokumen yang mengandung *term*

Pembobotan TF-IDF untuk sebuah *term* t dari dokumen D didapat dari perkalian TF dan IDF.

$$W_{d,t} = tf_{d,t}\ x\ idf_{d,t} \quad (2.12)$$

Keterangan :

W : bobot untuk sebuah *term*

tf : kemunculan kata pada tiap dokumen

idf : jumlah dokumen yang mengandung sebuah *term* yang dicari.

2.7 Evaluasi Kinerja

Untuk mengetahui hasil kinerja dari sistem dilakukan suatu proses pengujian atau evaluasi. Pada sistem ini digunakan *precision*, *recall*, dan *f-measure*. *Precision* dan *recall* merupakan dua ukuran luas digunakan untuk mengevaluasi kualitas hasil dalam pencarian informasi. *Precision* dapat dilihat sebagai ukuran ketepatan, sedangkan *recall* adalah ukuran kelengkapan. *Precision* mencerminkan bagaimana kemungkinan dokumen sistem diambil benar-benar baru dan *recall* mencerminkan bagaimana kemungkinan dokumen yang benar-benar baru dapat diambil oleh sistem. *Precision* dan *recall* didefinisikan pada tabel 2.1 [BER-10].

Tabel 2. 1 Evaluasi hasil kinerja

		Actual Class (expectation)	
		+	-
Predicted Class (Observation)	+	TP	FP
	-	FN	TN

$$Precision\ (P) = \frac{\text{TP}}{\text{TP+FP}} \quad (2.14)$$

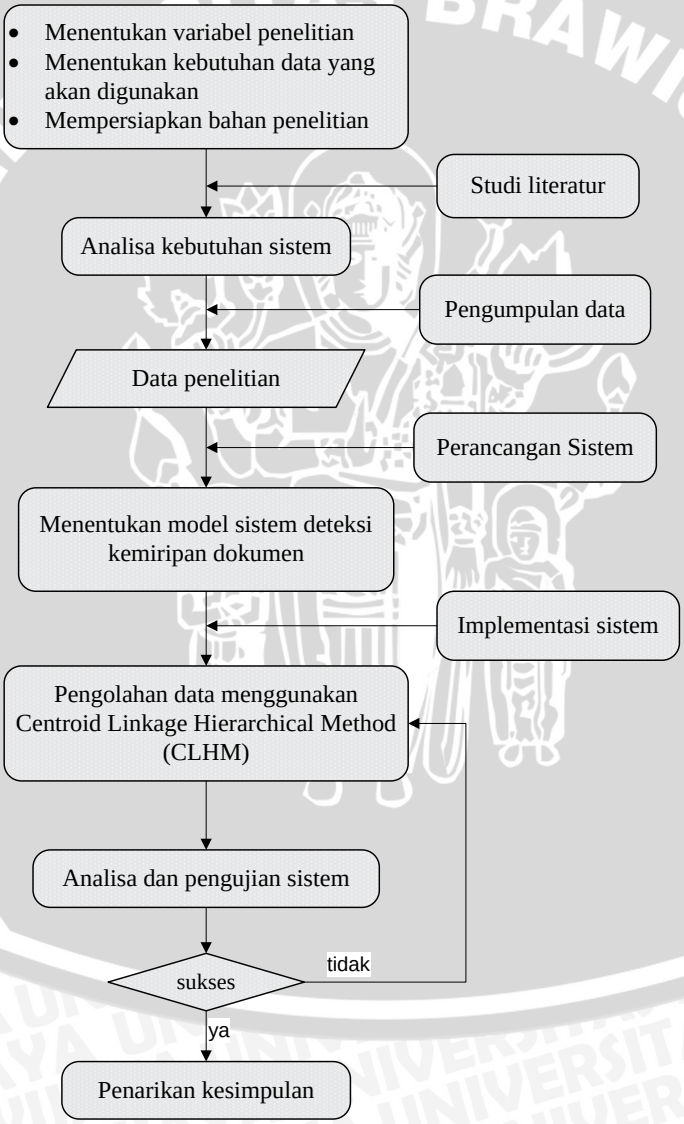
$$Recall\ (R) = \frac{\mathrm{TP}}{\mathrm{TP+FN}} \quad (2.15)$$

Ukuran evaluasi yang paling umum digunakan adalah *F-measure* yang merupakan rata-rata dari *precision* dan *recall*.

$$F - measure = \frac{2\,\mathrm{P}\,\mathrm{R}}{\mathrm{R+P}} \quad (2.16)$$

BAB III
METODE PENELITIAN DAN PERANCANGAN

Pada bab ini akan dibahas metode yang akan digunakan dalam pembuatan sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM). Adapun desain penelitiannya digambarkan pada diagram alir sebagai berikut:



Gambar 3. 1 Desain penelitian sistem penentuan tingkat plagiarisme dokumen
Sumber: Perancangan

3.1 Studi Literatur

Studi literatur digunakan untuk mengumpulkan dan mempelajari berbagai referensi yang bersumber dari buku, artikel *online*, jurnal ilmiah, laporan penelitian yang telah ada. Referensi utama yang digunakan untuk mendukung penulisan skripsi ini meliputi:

1. Pemahaman tentang struktur dokumen penelitian.
2. Pemahaman tentang *text mining*.
3. Metode *cosine similarity*.
4. *Clustering* menggunakan *Centroid Linkage Hierarchical Method* (CLHM).
5. Metode *Hill Climbing*.
6. Bahasa *script* PHP.
7. Database MySQL.

3.2 Metode Pengumpulan Data

Data yang dibutuhkan untuk penelitian ini adalah data tentang dokumen penelitian yang berisi judul, abstrak berbahasa Indonesia, dan *keyword* pada abstrak. Data dokumen penelitian meliputi dokumen tugas akhir, penelitian, dan makalah yang merupakan data sekunder. Data sekunder adalah data yang telah dikumpulkan oleh orang lain dan tidak dipersiapkan untuk kegiatan penelitian, tetapi dapat digunakan untuk tujuan penelitian. Data dokumen penelitian diambil dari internet.

3.3 Analisa Kebutuhan Sistem

Analisa kebutuhan bertujuan untuk menganalisis dan mendapatkan kebutuhan – kebutuhan yang diperlukan dalam perancangan sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM). Metode analisa yang digunakan adalah *procedural analysis* dengan menggunakan bahasa pemodelan prosedural. Pemrograman berbasis prosedur merupakan teknik pemrograman yang dikembangkan berdasarkan algoritma untuk memecahkan suatu masalah. Algoritma merupakan cara-cara yang ditempuh dalam memanipulasi data sehingga masalah yang dihadapi bisa dipecahkan.

Kebutuhan yang digunakan dalam pembuatan sistem penentuan tingkat plagiarisme dokumen ini diantaranya:

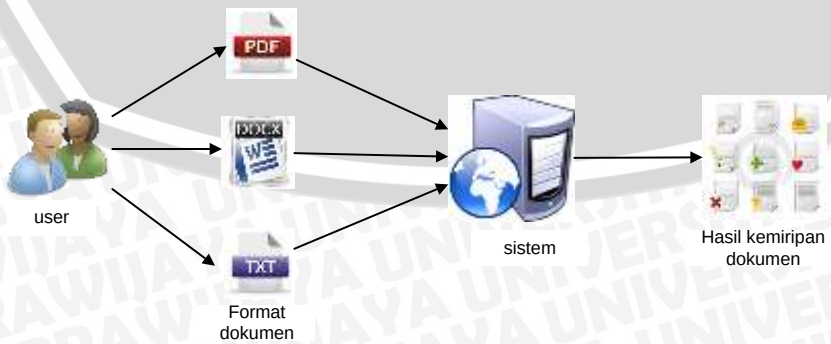
1. Kebutuhan *hardware*, meliputi:
   - Laptop atau PC
2. Kebutuhan *software*, meliputi:
   - Microsoft Windows 7 Ultimate sebagai sistem operasi
   - MySQL sebagai server *Database Management System*
   - Adobe Dreamwaver sebagai aplikasi untuk pembuatan sistem menggunakan bahasa pemrograman PHP.
3. Data yang dibutuhkan yaitu dokumen penelitian meliputi skripsi, tugas akhir, karya ilmiah yang memiliki abstrak berbahasa Indonesia.

3.4 Perancangan Sistem

Perancangan sistem memiliki tujuan sebagai tahap awal pengimplementasian desain sistem dan perancangan *interface* yang digunakan. Perancangan sistem dibangun berdasarkan hasil studi literatur dan analisis kebutuhan sistem. Tahap perancangan sistem meliputi arsitektur berbasis web pada sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM).

3.3.1 Arsitektur Sistem

Perancangan sistem penentuan tingkat plagiarisme dokumen penelitian untuk dokumen berberbahasa Indonesia dapat dilihat pada gambar 3.2.

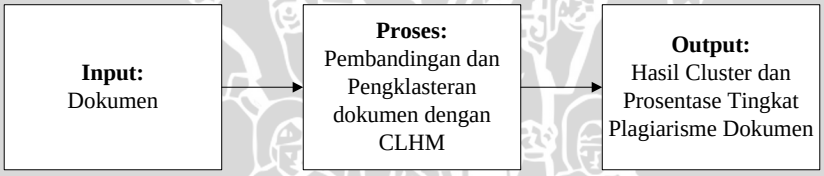


Gambar 3. 2 Arsitektur sistem penentuan tingkat plagiarisme dokumen
Sumber : Perancangan

Pada gambar 3.2 menjelaskan langkah kerja sistem. Pertama, *user* dapat memasukkan dokumen dalam bentuk format .pdf, atau .txt. Sistem kemudian akan memproses dokumen yang telah dimasukkan oleh *user* melalui proses yang terdiri dari *preprocessing*, perhitungan TF-IDF, *centroid linkage hierarchical method* (CLHM), *hill climbing*, dan *cosine similarity*. Hasil perbandingan dokumen dapat dilihat pada hasil *cluster* dan prosentase kemiripan tiap dokumen.

3.3.2 Blok Diagram Sistem

Diagram blok sistem merupakan penguraian logis dari fungsi-fungsi sistem dan memperlihatkan bagaimana bagian-bagian (blok-blok) yang berbeda mempengaruhi satu sama lain. Interaksi ini digambarkan dengan anak panah antar blok-blok. Sebuah sistem yang diberikan biasanya direpresentasikan oleh beberapa model diagram blok yang berbeda tergantung seberapa detail prosesnya. Garis besar perancangan sistem penentuan tingkat plagiarisme dapat dilihat pada gambar 3.3.



Gambar 3. 3 Diagram blok sistem penentuan tingkat plagiarisme
Sumber: Perancangan

3.3.3 Diagram Alir Sistem Penentuan Tingkat Plagiarisme Dokumen

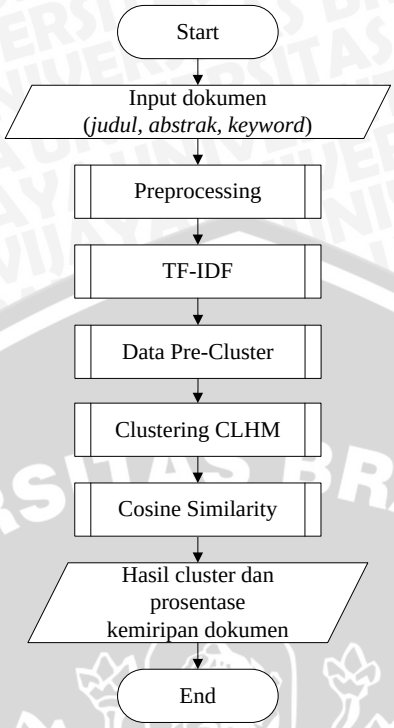
Diagram alir merupakan arus pekerjaan secara keseluruhan dari sistem. Diagram alir ini menunjukkan prosedur – prosedur yang ada pada keseluruhan sistem. Pada sistem penentuan tingkat plagiarisme dokumen ini terdiri dari beberapa komponen utama, yaitu tahap *text mining*, tahap *clustering* dokumen, tahap analisa varian, dan tahap *hill climbing*.

Tahap *text mining* atau biasa disebut *prepocessing* terdiri dari *tokenizing*, *filtering*, *stemming*, dan *analizing*. *Tokenizing* merupakan proses penghilangan tanda baca pada kalimat yang ada dalam dokumen sehingga menghasilkan kata yang berdiri sendiri. Setelah *tokenizing* selesai, proses selanjutnya adalah *filtering.* Proses *filtering* berfungsi untuk menghilangkan kata tidak penting, yang biasa

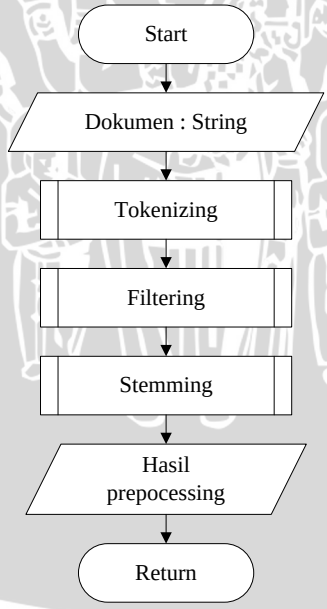
dikenal sebagai proses *stopword*. Contoh *stopword* adalah “ke”, “di”, “yang”, “dan”, “dari”, dan sebagainya. Pada proses *filtering* ini kata yang mengandung tanda baca akan dihilangkan, misalnya tanda baca titik, koma, petik dua, dan sebagainya. Setelah kedua proses ini selesai, dilakukan proses *stemming* yaitu proses merubah kata menjadi bentuk dasarnya. *Stemming* yang digunakan pada sistem ini adalah *stemming arifin-setiono*. Kata yang telah melewati tahap *stemming* kemudian dihitung frekuensi kata pada tiap dokumen menggunakan persamaan *Term Frequency* (TF). Setelah itu, digunakan *Inverse Document Frequency* (IDF) untuk mendapatkan bobot pada masing-masing dokumen. Setelah nilai TF dan IDF diperoleh, proses selanjutnya adalah pembobotan TF-IDF. Setelah masing – masing dokumen memiliki bobot, selanjutnya akan dilakukan proses *clustering* menggunakan metode *Centroid Linkage Hierarchical* (CLHM).

Centroid Linkage Herarchical Method (CLHM) adalah proses pengklasteran yang didasarkan pada jarak antar centroidnya. Metode *centroid linkage* ini berfungsi untuk mengklasterkan dokumen – dokumen berdasarkan nilai similaritasnya. Perhitungan jarak antar dokumen menggunakan *euclidian distance*. Setelah terbentuk *cluster* menggunakan CLHM, dilanjutkan dengan analisis varian tiap *cluster*. Setelah diperoleh nilai varian, proses selanjutnya adalah *hill climbing. Hill climbing* berfungsi untuk mengetahui keakuratan dari suatu metode pembentukan *cluster* pada *hierarchical method*. Proses *hill climbing* menghasilkan jumlah *cluster* yang optimal.

Proses terakhir dalam sistem ini adalah proses penentuan tingkat plagiarisme. Penentuan tingkat plagiarisme ini menggunakan *cosine similarity*. *Cosine similarity* dalam sistem ini berfungsi untuk menghitung similaritas kata antar dokumen dalam satu *cluster* yang sama. *Output* dari sistem akan menghasilkan anggota *cluster* dan menampilkan similaritas dokumen berdasarkan prosentase. Sistem penentuan tingkat plagiarisme dokumen penelitian ini dirancang sesuai diagram alir pada gambar 3.4.



Gambar 3. 4 Diagram alir sistem
Sumber : perancangan

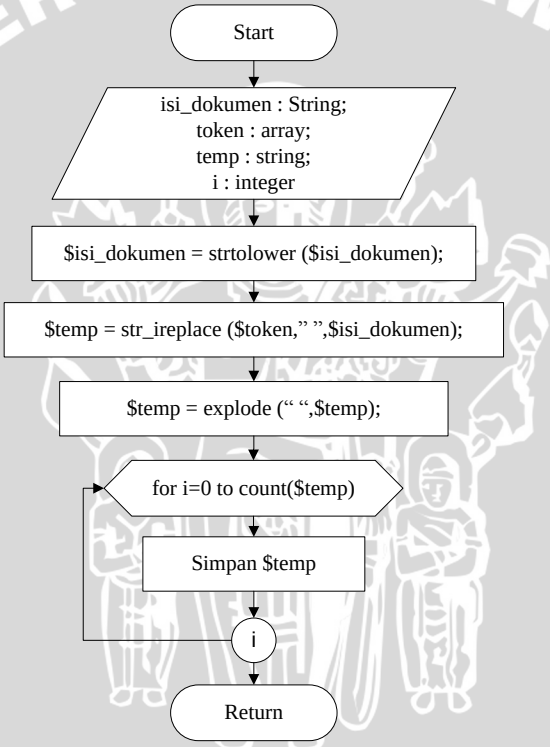


Gambar 3. 5 *Flowchart prepocessing*
Sumber: Perancangan

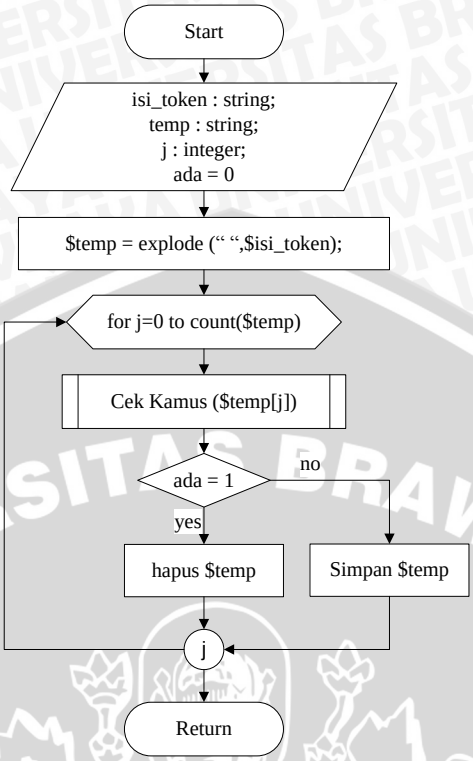
1. *Prepocessing*

Tahap *preprocessing* mempunyai beberapa proses. Proses-proses tersebut adalah *tokenizing, filtering*, dan *stemming*. Gambar 3.5 menunjukkan

proses *preprocessing* pada dokumen. Proses *tokenizing* adalah proses memecah kalimat menjadi potongan kata. Setelah *tokenizing* selesai, dilanjutkan dengan proses *filtering*. Proses *filtering* merupakan tahap pengambilan kata – kata yang penting dari hasil *tokenizing*. Proses *tokenizing* ditunjukkan pada gambar 3.6. Proses *filtering* pada sistem ini menggunakan algoritma *stopword* dimana tiap kata (*term*) akan dicek apakah kata tersebut ada dalam daftar *stopword*. Jika terdapat dalam *stopword*, kata tersebut akan dihilangkan sehingga setelah dilakukan proses *filtering* akan didapatkan *wordlist*. Proses *filtering* ditunjukkan pada gambar 3.7.

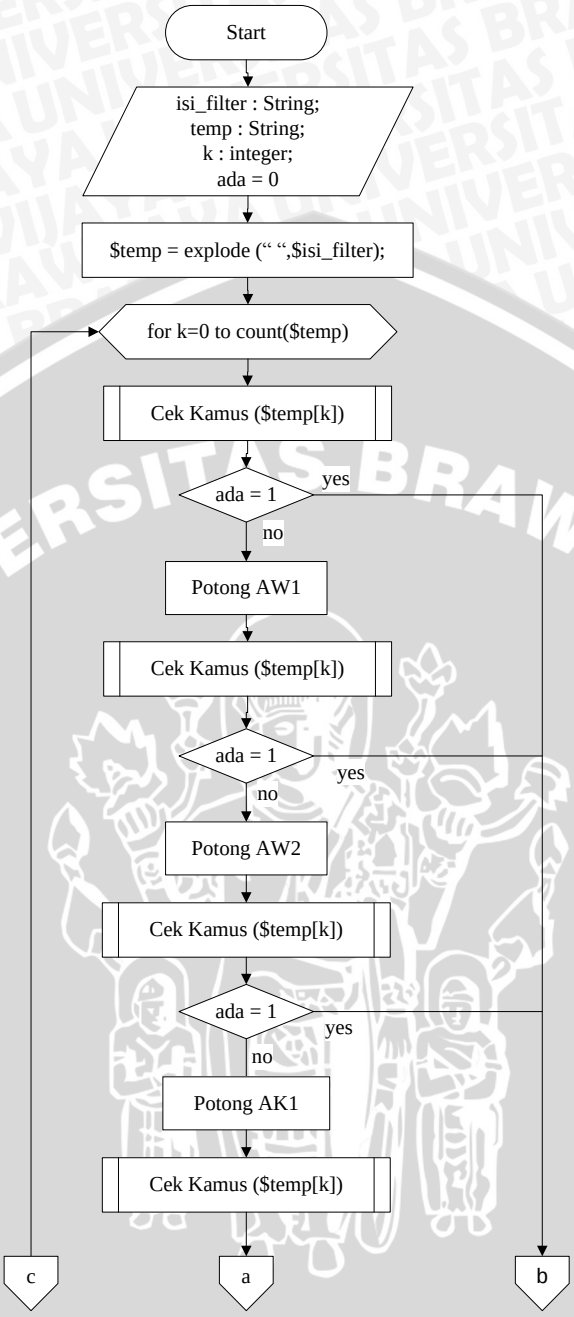


Gambar 3. 6 *Flowchart tokenizing*
Sumber : Perancangan

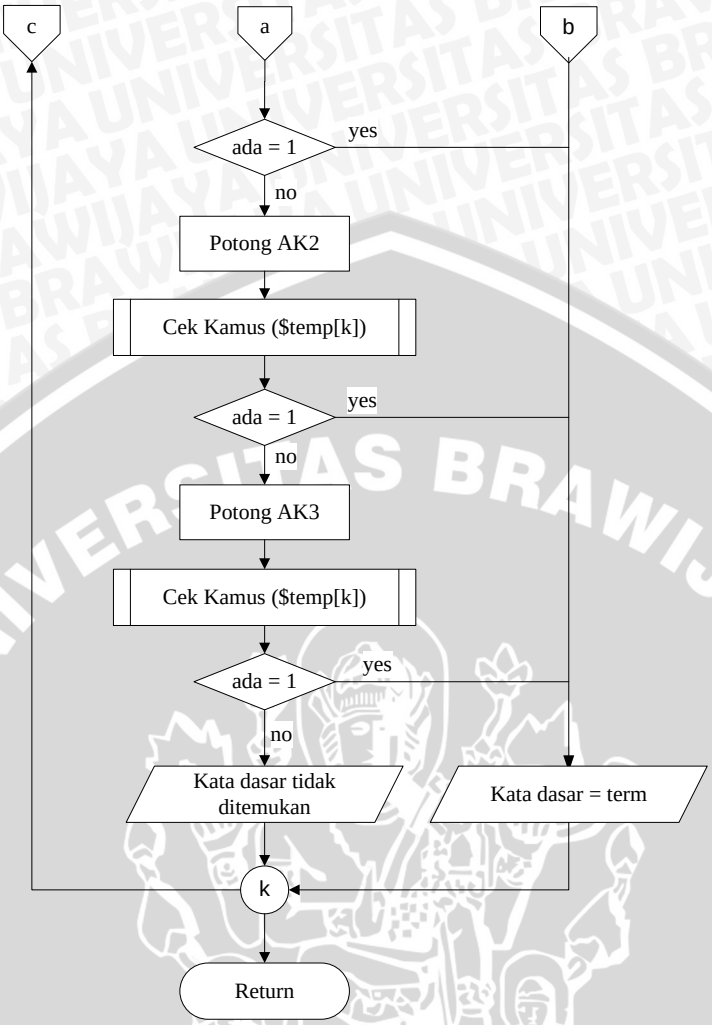


Gambar 3. 7 *Flowchart filtering*
Sumber: Perancangan

Proses *stemming* adalah proses pemotongan partikel-partikel *term* menjadi kata dasar. Algoritma *stemming* yang digunakan adalah *stemming* Arifin-Setiono. Proses *stemming* ini digunakan untuk menangani masalah kata pasif-aktif dan perubahan partikel kata. Proses *stemming* ditunjukkan pada gambar 3.8 sampai gambar 3.9.



Gambar 3. 8 *Flowchart stemming* Arifin-Setiono
Sumber : Perancangan



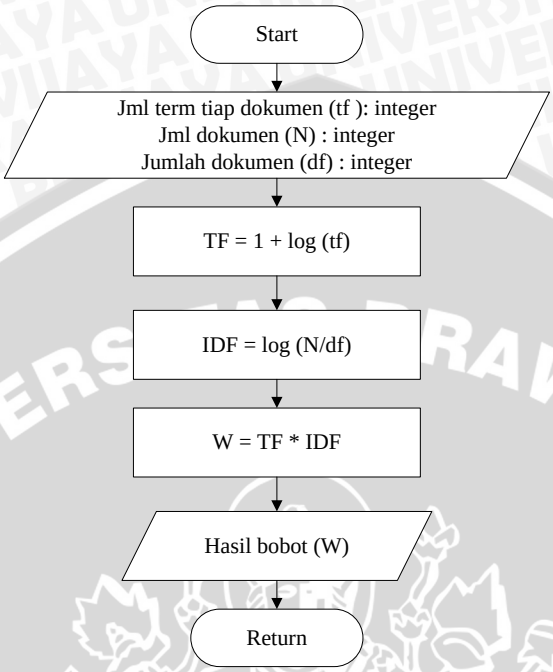
Gambar 3. 9 *Flowchart stemming* Arifin-Setiono (lanjutan)
Sumber : Perancangan

Setelah proses *stemming*, dilanjutkan dengan proses *analizing*. *Analizing* merupakan tahap penentuan seberapa jauh keterhubungan antar kata – kata antar dokumen yang ada. Proses analizing pada penelitian ini menggunakan pembobotan TF-IDF.

2. Pembobotan TF-IDF

Kata (*term*) tiap dokumen dihitung menggunakan pembobotan TF-IDF untuk mendapatkan bobot kata dari dokumen. *Term Frequency* (TF) berfungsi untuk menghitung bobot dari *term* t pada dokumen. Sedangkan *Inverse Document Frequency* (IDF) berfungsi untuk menghitung jumlah dokumen yang mengandung sebuah *term* yang dicari dari kumpulan dokumen yang ada.

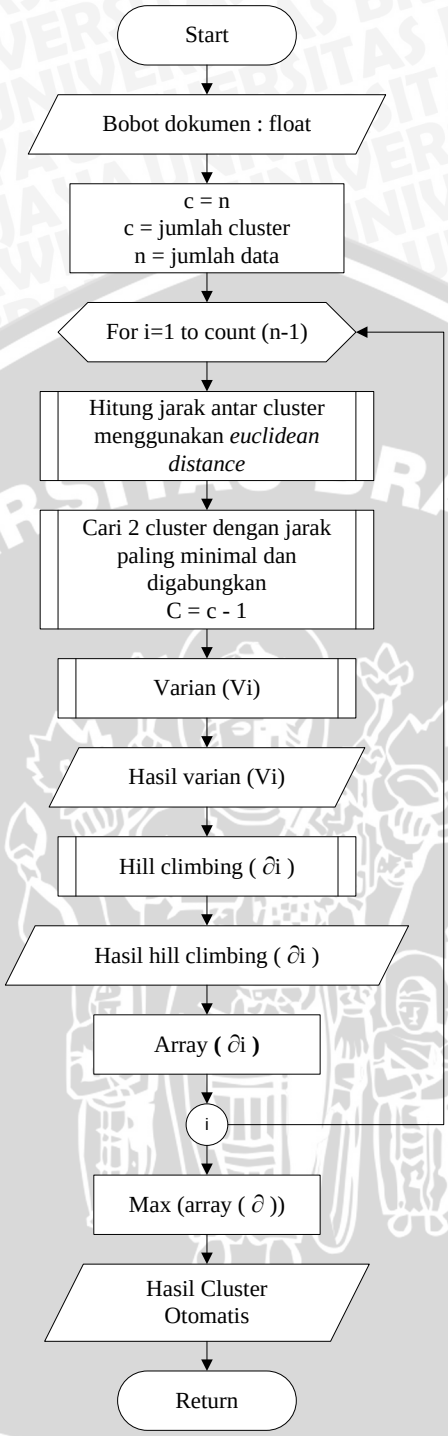
Setelah itu dilakukan pembobotan dengan mengalikan hasil dari TF dan IDF. *Flowchart* pembobotan TF-IDF ditunjukkan pada gambar 3.10.



Gambar 3. 10 *Flowchart* pembobotan TF-IDF
Sumber: Perancangan

3. Pembentukan *Cluster Centroid Linkage Hierarchical Method* (CLHM)

Setelah diketahui bobot tiap dokumen, langkah selanjutnya adalah proses pembentukan *cluster* menggunakan *Centroid Linkage Hierarchical Method* (CLHM). Pembentukan *cluster* menggunakan CLHM ini dilakukan untuk meningkatkan efisiensi dalam menemukan dokumen yang saling berkaitan. Sehingga, tiap dokumen termasuk ke dalam *cluster* tertentu dengan kedekatan dokumen berdasarkan *term*. *Flowchart* CLHM ditunjukkan pada gambar 3.11.

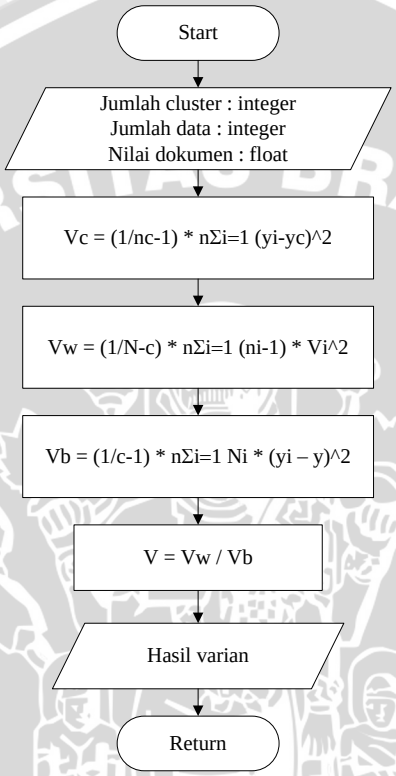


Gambar 3. 11 *Flowchart* CLHM
Sumber: Perancangan

4. Analisa Varian pada *Cluster*

Salah satu metode yang digunakan untuk menentukan *cluster* yang ideal adalah batasan varian. Batasan varian dihasilkan dengan menghitung kepadatan

cluster berupa *variance within cluster* (V_w) dan *variance between cluster* (V_b). Varian dihitung setelah terbentuknya *cluster* menggunakan CLHM. Setiap nilai varian dari pada tahap pembentukan *cluster* dianalisa untuk menentukan jumlah *cluster* yang ideal. Gambar 3.12 menunjukkan *flowchart* penentuan varian.



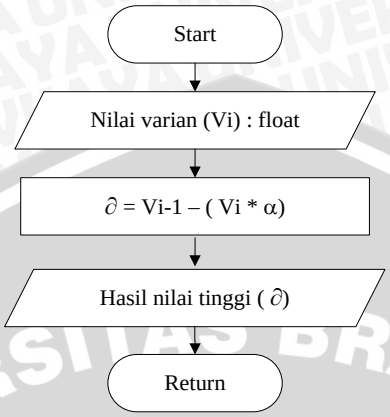
Gambar 3. 12 *Flowchart* varian
Sumber: Perancangan

Nilai varian merupakan salah satu metode untuk memperoleh jumlah *cluster* yang mencapai *global optimum* yang mampu mengatasi masalah dari *minimum V*. Metode *Hill Climbing* digunakan untuk menemukan posisi *global optimum* yang mungkin dari pergerakan varian pada sistem ini.

5. Hill Climbing

Proses *hill climbing* dilakukan setelah didapatkan nilai varian. *Hill climbing* berfungsi untuk mengetahui keakuratan dari suatu metode pembentukan *cluster* pada *hierarchical method*. *Hill climbing* dapat

mengetahui optimasi jumlah *cluster* yang terbentuk pada proses CLHM. *Flowchart hill climbing* ditunjukkan pada gambar 3.13.

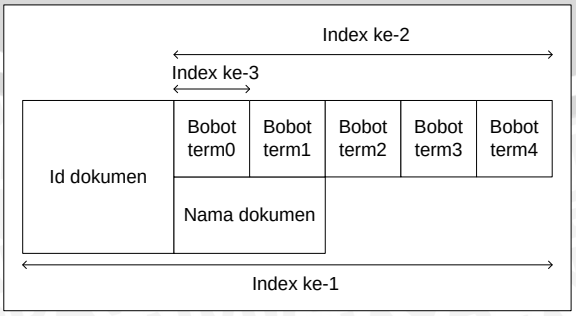


Gambar 3. 13 *Flowchart hill climbing*
Sumber: Perancangan

Pendekatan metode *hill-climbing* dilakukan untuk memperoleh perbedaan nilai tinggi (∂) pada tiap tahap pembentukan *cluster*. *Cluster* secara otomatis terbentuk ketika didapatkan nilai maksimum (∂).

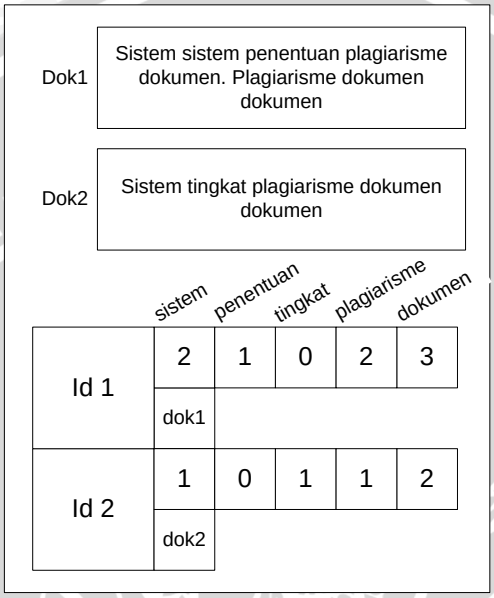
3.3.4 Perancangan Struktur Data

Perancangan struktur data merupakan perancangan data, dalam penelitian ini adalah perancangan dokumen dan *term* yang dimiliki. Perancangan struktur data pada penelitian ini digunakan untuk merancang dokumen yang akan diolah sebelum proses *clustering* menggunakan CLHM. Kumpulan dokumen akan diolah menjadi bentuk *array* tiga dimensi. Gambar 3.14 menunjukkan perancangan struktur data dari dokumen dan gambar 3.15 merepresentasikan bentuk dokumen yang akan diolah.



Gambar 3. 14 Perancangan struktur data dari dokumen
Sumber: Perancangan

Proses pengolahan data sebelum dikluster adalah proses mengubah bentuk dokumen menjadi *array* tiga dimensi, dimana *index* kesatu merupakan dokumen keseluruhan. *Index* kedua merupakan pemecahan data berdasarkan nama dokumen dan kumpulan term dokumen. Bobot masing-masing term dan nama dokumen berada pada *index* ketiga.



Gambar 3. 15 Ilustrasi dokumen dengan *array* tiga dimensi
Sumber: Perancangan

Dari ilustrasi pada gambar 3.15 bentuk dokumen yang akan diproses dalam *clustering* adalah sebagai berikut:

[0] => array
  [0] => array
    [0] => nilai $term_o$
    [1] => nilai $term_1$
    ……..
    [n] => nilai $term_n$
  [1] => array
    [0] => nama dokumen

Hasil dari pembuatan *array* tersebut akan digunakan dalam proses pembentukan *cluster* menggunakan CLHM.

3.3.5 Perancangan *Database*

Perancangan *database* merupakan perancangan manajemen data yang akan digunakan. Manajemen data termasuk *basis data* yang mengandung data relevan untuk berbagai situasi dan diatur oleh *software* yang disebut *Database Management System* (DBMS). Sistem ini menggunakan DBMS yaitu MySQL. Perancangan basis data sistem ini menggunakan enam tabel yaitu tabel daftar_dokumen, tabel dokumen, tabel bobot_dokumen, tabel token, tabel stopword, dan tabel kamus_kata. Daftar dokumen digunakan untuk menyimpan informasi dokumen. Tabel dokumen berfungsi untuk menyimpan tf dan df dokumen. Tabel bobot_dokumen berfungsi untuk menyimpan hasil normalisasi dokumen. Tabel token berfungsi untuk menyimpan kumpulan token. Tabel stopword berfungsi untuk menyimpan kumpulan kata yang termasuk *stopword*. Tabel kamus_kata berfungsi untuk menyimpan kumpulan kata dasar yang digunakan pada proses *stemming*. Tabel pada sistem ini tidak berelasi dengan tabel yang lainnya. Tabel perancangan *database* ditunjukkan pada gambar 3.16.

Gambar 3. 16 Tabel Database sistem penentuan kemiripan
Sumber: Perancangan

3.3.6 Manualisasi Sistem Penentuan Tingkat Plagiarisme Dokumen

Perhitungan manual berfungsi untuk memberikan gambaran umum perancangan sistem yang akan dibangun. Sistem penentuan tingkat plagiarisme dokumen penelitian ini menggunakan *Centroid Linkage Hierarchical Method* (CLHM). Contoh manualisasi CLHM pada kasus penentuan plagiarisme dokumen penelitian adalah sebagai berikut:

1. Pengambilan informasi dokumen.

PENINGKATAN KINERJA CLUSTERING DOKUMEN TEKS MENGGUNAKAN PEMBOBOTAN SAMPEL

Amir Hamzah
Jurusan Teknik Informatika, Fakultas Teknologi Industri,
Institut Sains & Teknologi AKPRIND, Jalan Kalisahak 28, Yogyakarta 55222
Phone: (0274)-563029; Fax: (0274)-563847
e-mail: mramzah@yahoo.co.id

Abstrak
Algoritma clustering berbasis pembobotan sampel (sample weighting) saat ini banyak diteliti. Ada beberapa model pembobotan yang pada prinsipnya bertujuan untuk merubah nilai vektor sampel dan formula similaritas vektor sampel dengan pusat clusternya. Dalam dokumen teks pembobotan dapat berupa konektifitas antar dokumen, misalnya dalam dokumen akademik yang ada koneksi referensi. Namun dalam dokumen berita koneksi referensi mungkin jarang ditemukan. Dalam makalah ini teknik pembobotan baru diajukan, yaitu menggunakan kata-kata yang muncul dalam kata kunci (keyword) dan judul (title) dari suatu dokumen teks. Eksperimen dilakukan terhadap abstrak dokumen akademik sebanyak 500 dokumen dan dokumen berita. Sebanyak 3000 dokumen Algoritma yang diuji kinerjanya adalah algoritma K-Means clustering dan algoritma Fuzzy C-Means clustering. Parameter kinerja algoritma digunakan nilai F-measure dari hasil clustering sebelum dilakukan pembobotan sampel dan setelah dilakukan pembobotan sampel. Hasil eksperimen menunjukkan bahwa pembobotan sampel dapat meningkatkan kinerja clustering sebesar 12,8% untuk pembobotan dengan keyword dan title dan meningkatkan kinerja clustering 9,8% untuk pembobotan dengan title saja.

Kata kunci: clustering dokumen, pembobotan sampel, kinerja clustering, F-measure

Gambar 3. 17 Contoh dokumen uji
Sumber: Perancangan

Tabel 3. 1 Contoh dokumen

Dokumen	Judul, abstrak, dan kata kunci
Dok1	MESIN PENCARI DOKUMEN DENGAN PENGKLASTERAN SECARA OTOMATIS Web mining untuk pencarian berdasarkan kata kunci dengan pengklasteran otomatis adalah suatu metode pencarian dokumen dengan cara mengelompokkan atau mengklaster dokumen dari dokumen-dokumen berdasarkan kata kuncinya. Selanjutnya dilakukan pengklasteran dengan metode centroid linkage hierarchical method (CLHM) terhadap jumlah kata kunci yang diperoleh dari masing-masing dokumen. Dalam pengklasteran, umumnya harus dilakukan inisialisasi jumlah klaster yang ingin dibentuk terlebih dahulu, padahal pada beberapa kasus pengklasteran, user bahkan tidak tahu berapa banyak klaster yang bisa dibangun. Untuk itu, pada makalah ini diaplikasikan metode Valley Tracing sebagai constraint yang akan melakukan identifikasi terhadap pergerakan varian dari tiap tahap pembentukan klaster dan menganalisa polanya untuk membentuk suatu klaster secara otomatis (automatic clustering). Data yang digunakan adalah data hasil dari proses text mining pada dokumen. Dari percobaan yang dilakukan dengan 424 dokumen hasilnya memberikan simpulan bahwa pada umumnya pencarian dokumen menggunakan teknik pengklasteran dengan algoritma CLHM dapat digunakan untuk mengelompokkan dokumen dengan jumlah yang tepat secara otomatis. Automatic clustering, CLHM, text mining, valley tracing
Dok2	PENINGKATAN KINERJA CLUSTERING DOKUMEN TEKS MENGGUNAKAN PEMBOBOTAN SAMPEL Algoritma clustering berbasis pembobotan sampel (sample weighting) saat ini banyak diteliti. Ada beberapa model pembobotan yang pada prinsipnya bertujuan untuk merubah nilai vektor sampel dan formula similaritas vektor sampel dengan pusat clusternya. Dalam dokumen teks pembobotan dapat berupa konektifitas antar dokumen, misalnya dalam dokumen akademik yang ada koneksi referensi. Namun dalam dokumen berita koneksi referensi mungkin jarang ditemukan. Dalam makalah ini teknik pembobotan baru diajukan, yaitu menggunakan kata-kata yang muncul dalam kata kunci (keyword) dan judul (title) dari suatu dokumen teks. Eksperimen dilakukan terhadap abstrak dokumen akademik sebanyak 500 dokumen dan dokumen berita. Sebanyak 3000

	dokumen Algoritma yang diuji kinerjanya adalah algoritma K-Means clustering dan algoritma Fuzzy C-Means clustering. Parameter kinerja algoritma digunakan nilai F-measure dari hasil clustering sebelum dilakukan pembobotan sampel dan setelah dilakukan pembobotan sampel. Hasil eksperimen menunjukkan bahwa pembobotan sampel dapat meningkatkan kinerja clustering sebesar 12,8% untuk pembobotan dengan keyword dan title dan meningkatkan kinerja clustering 9.8% untuk pembobotan dengan title saja. Clustering dokumen, pembobotan sampel, kinerja clustering, F-measure
Dok3	PENGELOMPOKKAN DOKUMEN MENGGUNAKAN K-MEANS DAN SINGULAR VALUE DECOMPOSITION: STUDI KASUS MENGGUNAKAN DATA BLOG Peningkatan jumlah dokumen dalam format teks yang cukup signifikan belakangan ini, seperti blogs dan website, membuat proses pengelompokkan dokumen (document clustering) menjadi semakin penting. Pengelompokkan dokumen bertujuan membagi dokumen dalam beberapa kelompok (cluster) sedemikian hingga dokumen-dokumen dalam cluster yang sama (intra-cluster) memiliki derajat kesamaan yang tinggi, sementara dokumen-dokumen dalam cluster yang berbeda (inter-cluster) memiliki derajat kesamaan yang rendah. Tulisan ini mendiskusikan dan memperlihatkan metode pengelompokkan dokumen yang dimulai dengan membangun matriks terms-documents A dan kemudian memecahnya menjadi tiga matriks TSD menggunakan Singular Value Decomposition (SVD). Selanjutnya, pengelompokkan dokumen dilakukan dengan k-means. T adalah matriks kata (terms) berukuran t x r, S adalah matriks diagonal berisi nilai skalar (eigen values) berdimensi r x r, dan r ditentukan sebelumnya, D adalah matriks dokumen berukuran r x d. Dekomposisi nilai singular dari matriks A dinyatakan sebagai A = TSDT. Eksperimen dilakukan menggunakan data blog dan dekomposisi matrik menggunakan program General Text Parser (GTP). Hasil menunjukkan bahwa dekomposisi matrik terms-documents A dengan Singular Value Decomposion dapat mempercepat proses pengelompokkan dokumen karena dimensi dari setiap vector telah diperkecil tanpa mengurangi arti sebenarnya. Namun, karena metode clustering yang digunakan adalah k-means maka hasil cluster sangat sensitif terhadap dokumen yang diduga sebagai outlier. Document clustering, Singular Value Decomposition, k-means, Latent Semantic Indexing
Dok4	SISTEM VERIFIKASI WAJAH MENGGUNAKAN JARINGAN SARAF TIRUAN LEARNING VECTOR QUANTIZATION Verifikasi wajah merupakan salah satu teknologi biometrika yang menjadi perhatian para peneliti. Banyak sekali sistem aplikasi yang berbasis kepada verifikasi wajah misalnya: akses pintu, akses mesin ATM, sistem presensi kehadiran, dll. Pada makalah ini akan dibahas perancangan dan pembuatan sistem verifikasi wajah manusia menggunakan metode ekstraksi menggunakan metode SPCA (Simple Principle Component Analysis) dan teknik klasifikasi jaringan saraf tiruan Learning Vector Quantization. Data citra wajah yang digunakan berasal dari 5 orang yang terdiri masing-masing sebanyak 10 citra wajah untuk proses pelatihan dan juga masingmasing sebanyak 10 wajah untuk proses pengujian. Hasil pengujian unjuk kerja sistem didapat nilai FRR rata-rata 0% dan FAR rata-rata = 1,55%. Verifikasi wajah, SPCA, learning vector quantization

Sumber: Perancangan

Setelah mendapatkan judul, abstrak dan kata kunci dari masing-masing dokumen, langkah berikutnya adalah *preprocessing*.

2. *Prepocessing* dokumen

Tokenizing merupakan proses penghilangan karakter dan *parsing*. *Parsing* adalah tahap memecah kalimat menjadi kata yang berdiri sendiri. Hasil *tokenizing* ditunjukkan oleh tabel 3.2. Setelah proses *tokenizing* dilanjutkan dengan proses *filtering*. *Filtering* merupakan proses penghilangan kata yang tidak penting dan mengambil kata yang penting atau biasa disebut dengan istilah *stopword removal*. Hasil *tokenizing* ditunjukkan oleh tabel 3.3.

Tabel 3. 2 Contoh dokumen hasil *tokenizing*

Dokumen	**Judul, abstrak, dan kata kunci**
Dok1	mesin pencari dokumen pengklasteran secara otomatis web mining pencarian kata kunci pengklasteran otomatis metode pencarian dokumen cara mengelompokkan mengklaster dokumen dokumendokumen kata kuncinya selanjutnya dilakukan pengklasteran metode centroid linkage hierarchical method clhm jumlah kata kunci diperoleh dokumen pengklasteran umumnya dilakukan inisialisasi jumlah klaster dibentuk kasus pengklasteran user tahu klaster dibangun makalah diaplikasikan metode valley tracing constraint melakukan identifikasi pergerakan varian tahap pembentukan klaster menganalisa polanya membentuk klaster secara otomatis automatic clustering data digunakan data hasil proses text mining dokumen percobaan dilakukan 424 dokumen hasilnya memberikan simpulan umumnya pencarian dokumen menggunakan teknik pengklasteran algoritma clhm digunakan mengelompokkan dokumen jumlah tepat secara otomatis automatic clustering clhm text mining valley tracing
Dok2	peningkatan kinerja clustering dokumen teks menggunakan pembobotan sampel algoritma clustering pembobotan sampel sample weighting diteliti model pembobotan prinsipnya bertujuan merubah nilai vektor sampel formula similaritas vektor sampel pusat clusternya dokumen teks pembobotan berupa konektifitas dokumen misalnya dokumen akademik koneksi referensi dokumen berita koneksi referensi jarang ditemukan makalah teknik pembobotan baru diajukan menggunakan katakata muncul kata kunci keyword judul title dokumen teks eksperimen dilakukan abstrak dokumen akademik 500 dokumen dokumen berita 3000 dokumen algoritma diuji kinerjanya algoritma kmeans clustering algoritma fuzzy cmeans clustering parameter kinerja algoritma digunakan nilai fmeasure hasil clustering dilakukan pembobotan sampel dilakukan pembobotan sampel hasil eksperimen menunjukkan pembobotan sampel meningkatkan kinerja clustering sebesar 128% pembobotan keyword title meningkatkan kinerja clustering 9 8% pembobotan title clustering dokumen pembobotan sampel kinerja clustering fmeasure
Dok3	pengelompokkan dokumen menggunakan kmeans singular value decomposition studi kasus menggunakan data blog peningkatan jumlah dokumen format teks cukup signifikan belakangan blogs website membuat proses pengelompokkan dokumen document clustering menjadi penting

	pengelompokkan dokumen bertujuan membagi dokumen kelompok cluster dokumendokumen cluster intracluster memiliki derajat kesamaan tinggi dokumendokumen cluster berbeda intercluster memiliki derajat kesamaan rendah tulisan mendiskusikan memperlihatkan metode pengelompokkan dokumen dimulai membangun matriks termsdocuments a memecahnya menjadi tiga matriks tsd menggunakan singular value decomposition svd selanjutnya pengelompokkan dokumen dilakukan kmeans t matriks kata terms berukuran t x r s matriks diagonal berisi nilai skalar eigen values berdimensi r x r r ditentukan d matriks dokumen berukuran r x d dekomposisi nilai singular matriks a dinyatakan a tsdt eksperimen dilakukan menggunakan data blog dekomposisi matrik menggunakan program general text parser gtp hasil menunjukkan dekomposisi matrik termsdocuments a singular value decomposion mempercepat proses pengelompokkan dokumen dimensi vector diperkecil mengurangi arti metode clustering digunakan kmeans hasil cluster sensitif dokumen diduga outlier document clustering singular value decomposition kmeans latent semantic indexing
Dok4	sistem verifikasi wajah menggunakan jaringan saraf tiruan learning vector quantization verifikasi wajah salah satu teknologi biometrika menjadi perhatian peneliti sistem aplikasi verifikasi wajah misalnya akses pintu akses mesin atm sistem presensi kehadiran dll makalah dibahas perancangan pembuatan sistem verifikasi wajah manusia menggunakan metode ekstraksi menggunakan metode spca simple principle component analysis teknik klasifikasi jaringan saraf tiruan learning vector quantization data citra wajah digunakan berasal 5 orang 10 citra wajah proses pelatihan 10 wajah proses pengujian hasil pengujian unjuk kerja sistem didapat nilai frr ratarata 0% far ratarata 155% verifikasi wajah spca learning vector quantization

Sumber: Perancangan

Tabel 3. 3 Contoh dokumen hasil *filtering*

Dokumen	**Judul, abstrak, dan kata kunci**
Dok1	mesin pencari dokumen pengklasteran otomatis web mining pencarian kata kunci pengklasteran otomatis metode pencarian dokumen cara mengelompokkan mengklaster dokumen dokumendokumen kata kuncinya pengklasteran metode centroid linkage hierarchical method clhm jumlah kata kunci dokumen pengklasteran umumnya inisialisasi jumlah klaster dibentuk kasus pengklasteran user tahu klaster dibangun makalah diaplikasikan metode valley tracing constraint identifikasi pergerakan varian tahap pembentukan klaster menganalisa polanya membentuk klaster otomatis automatic clustering data digunakan data hasil proses text mining dokumen percobaan dokumen hasilnya memberikan simpulan umumnya pencarian dokumen menggunakan teknik pengklasteran algoritma clhm digunakan mengelompokkan dokumen jumlah tepat otomatis automatic clustering clhm text mining valley tracing
Dok2	peningkatan kinerja clustering dokumen teks menggunakan pembobotan sampel algoritma clustering berbasis pembobotan sampel sample weighting diteliti model pembobotan prinsipnya bertujuan merubah nilai vektor sampel formula similaritas vektor sampel pusat clusternya dokumen teks pembobotan berupa konektifitas dokumen misalnya dokumen akademik koneksi referensi dokumen berita koneksi referensi jarang ditemukan makalah teknik pembobotan baru diajukan menggunakan katakata muncul kata kunci keyword judul title dokumen teks eksperimen abstrak dokumen akademik dokumen dokumen berita dokumen algoritma diuji kinerjanya

	algoritma kmeans clustering algoritma fuzzy cmeans clustering parameter kinerja algoritma digunakan nilai fmeasure hasil clustering pembobotan sampel pembobotan sampel hasil eksperimen menunjukkan pembobotan sampel meningkatkan kinerja clustering sebesar pembobotan keyword title meningkatkan kinerja clustering pembobotan title clustering dokumen pembobotan sampel kinerja clustering fmeasure
Dok3	pengelompokkan dokumen menggunakan kmeans dan singular value decomposition studi kasus menggunakan data blog peningkatan jumlah dokumen format teks cukup signifikan belakangan blogs website membuat proses pengelompokkan dokumen document clustering menjadi penting pengelompokkan dokumen bertujuan membagi dokumen kelompok cluster dokumendokumen cluster intracluster memiliki derajat kesamaan tinggi dokumendokumen cluster berbeda intercluster memiliki derajat kesamaan rendah tulisan mendiskusikan memperlihatkan metode pengelompokkan dokumen dimulai membangun matriks termsdocuments a memecahnya menjadi tiga matriks tsd menggunakan singular value decomposition svd selanjutnya pengelompokkan dokumen kmeans t matriks kata terms berukuran t x r s matriks diagonal berisi nilai skalar eigen values berdimensi r x r r ditentukan d matriks dokumen berukuran r x d dekomposisi nilai singular matriks a dinyatakan a tsdt eksperimen menggunakan data blog dekomposisi matrik menggunakan program general text parser gtp hasil menunjukkan dekomposisi matrik termsdocuments a dengan singular value decomposion mempercepat proses pengelompokkan dokumen dimensi vector diperkecil mengurangi arti metode clustering digunakan kmeans hasil cluster sensitif dokumen diduga outlier document clustering singular value decomposition kmeans latent semantic indexing
Dok4	sistem verifikasi wajah menggunakan jaringan saraf tiruan learning vector quantization verifikasi wajah salah satu teknologi biometrika perhatian peneliti sistem aplikasi berbasis verifikasi wajah misalnya akses pintu akses mesin atm sistem presensi kehadiran dll makalah dibahas perancangan pembuatan sistem verifikasi wajah manusia menggunakan metode ekstraksi menggunakan metode spca simple principle component analysis teknik klasifikasi jaringan saraf tiruan learning vector quantization data citra wajah digunakan berasal orang citra wajah proses pelatihan masingmasing sebanyak wajah proses pengujian hasil pengujian unjuk kerja sistem didapat nilai frr ratarata far ratarata verifikasi wajah spca learning vector quantization

Sumber: Perancangan

Selanjutnya adalah proses *stemming* arifin-setiono. Tabel 3.4 menunjukkan hasil *stemming* dokumen.

Tabel 3. 4 Contoh dokumen hasil *stemming*

Dokumen	**Judul, abstrak, dan kata kunci**
Dok1	mesin cari dokumen klaster secara otomatis web mining cari kata kunci klaster otomatis metode cari dokumen cara kelompok klaster dokumen dokumen kata kunci lanjut laku klaster metode centroid linkage hierarchical method clhm jumlah kata kunci dokumen klaster umum laku inisialisasi jumlah klaster bentuk kasus klaster user tahu klaster bangun makalah aplikasi metode valley trace constraint laku identifikasi gerak varian tahap bentuk klaster analisa pola bentuk klaster secara otomatis automatic clustering data guna data hasil proses text mining dokumen coba laku dokumen hasil beri simpul umum cari dokumen guna teknik klaster algoritma

	clhm guna kelompok dokumen jumlah tepat secara otomatis automatic clustering clhm text mining valley tracing
Dok2	tingkat kinerja clustering dokumen teks guna bobot sampel algoritma clustering bobot sampel sample weight teliti model bobot prinsip tujuan ubah nilai vektor sampel formula similaritas vektor sampel pusat kluster dokumen teks bobot rupa konektifitas dokumen misal dokumen akademik koneksi referensi dokumen berita koneksi referensi jarang temu makalah teknik bobot baru aju guna katakata muncul kata kunci keyword judul title dokumen teks eksperimen laku abstrak dokumen akademik dokumen dokumen berita dokumen algoritma uji kinerja algoritma kmeans clustering algoritma fuzzy cmeans clustering parameter kinerja algoritma guna nilai fmeasure hasil clustering laku bobot sampel laku bobot sampel hasil eksperimen tunjuk bobot sampel tingkat kinerja clustering besar bobot keyword title tingkat kinerja clustering bobot title clustering dokumen bobot sampel kinerja clustering fmeasure
Dok3	kelompok dokumen guna kmeans singular value decomposition studi kasus guna data jumlah dokumen format teks cukup signifikan belakang blog website buat proses kelompok dokumen dokumen clustering jadi penting kelompok dokumen tujuan bagi dokumen kelompok kluster dokumen kluster kluster milik derajat sama tinggi dokumen kluster beda intercluster milik derajat sama rendah tulis diskusi lihat metode kelompok dokumen mulai bangun matriks terms documents pecah jadi tiga matriks guna singular value decomposition lanjut kelompok dokumen laku kmeans matriks kata term ukur matriks diagonal isi nilai skalar eigen value dimensi tentu matriks dokumen ukur dekomposisi nilai singular matriks nyata eksperimen laku guna data blog dekomposisi matrik guna program general text parser hasil tunjuk dekomposisi matrik terms documents singular value decomposion cepat proses kelompok dokumen dimensi vector kecil kurang arti metode clustering guna kmeans hasil kluster sensitif dokumen duga outlier dokumen clustering singular value decomposition kmeans latent semantic
Dok4	sistem verifikasi wajah guna jaringan saraf tiru learn vector quantization verifikasi wajah salah satu teknologi biometrika jadi hati peneliti sistem aplikasi verifikasi wajah misal akses pintu akses mesin atm sistem presensi hadir dll makalah bahas rancang buat sistem verifikasi wajah manusia guna metode ekstraksi guna metode simple principle component analysis teknik klasifikasi jaringan saraf tiru learn vector quantization data citra wajah guna asal orang citra wajah proses latih wajah proses uji hasil uji unjuk kerja sistem dapat nilai rata far rata verifikasi wajah learn vector

Sumber: Perancangan

3. Setelah itu dilakukan pembobotan menggunakan TF-IDF. Pada tabel 3.5 menampilkan perhitungan term yang ada pada tiap dokumen. Misalnya pada dokumen yang bernama dok1 dan dok4 memiliki *term* ‘mesin’ sebanyak satu, dok2 dan dok3 tidak memiliki *term* ‘mesin’, dan seterusnya. Kolom DF untuk mengetahui jumlah dokumen yang mengandung *term.*

Tabel 3. 5 Informasi dokumen

Kata	Dok1	Dok2	Dok3	Dok4	DF
mesin	1	0	0	1	2
cari	4	0	0	0	1
dokumen	8	11	10	0	3
pengklasteran	6	0	0	0	1
otomatis	4	0	0	0	1
web	1	0	0	0	1
mining	3	0	0	0	1
kata	3	1	1	0	3
kunci	3	1	0	0	2
metode	3	0	2	2	3
cara	1	0	0	0	1
kelompok	2	0	7	0	2
mengklaster	1	0	0	0	1
dokumendokumen	1	0	2	0	2
centroid	1	0	0	0	1
linkage	1	0	0	0	1
hierarchical	1	0	0	0	1
method	1	0	0	0	1
clhm	3	0	0	0	1
jumlah	3	0	1	0	2
umum	2	0	0	0	1
inisialisasi	1	0	0	0	1
klaster	4	0	0	0	1
bentuk	2	0	0	0	1
kasus	1	0	1	0	2
user	1	0	0	0	1
tahu	1	0	0	0	1
bangun	1	0	1	0	2
makalah	1	1	0	1	3
aplikasi	1	0	0	1	2
valley	2	0	0	0	1
tracing	2	0	0	0	1
clustering	2	9	3	0	3
data	2	0	2	1	3
guna	3	3	6	4	4
hasil	2	2	2	1	4
proses	1	0	2	2	3
text	2	0	1	0	2
coba	1	0	0	0	1
pi	1	0	0	0	1

simpul	1	0	0	0	1
teknik	1	1	0	1	3
algoritma	1	5	0	0	2
tepat	1	0	0	0	1
tingkat	0	3	1	0	2
kinerja	0	6	0	0	1
teks	0	3	1	0	2
bobot	0	11	0	0	1
sampel	0	8	0	0	1
nilai	0	2	2	1	3
vektor	0	2	0	0	1
formula	0	1	0	0	1
similaritas	0	1	0	0	1
pusat	0	1	0	0	1
…	…	…	…	…	…
misal	0	1	0	1	2

Sumber: Perancangan

Tabel 3.6 menunjukkan hasil pembobotan TF-IDF. Berikut adalah contoh perhitungan bobot (W) mengunakan TF-IDF. Pembobotan merupakan hasil perkalian dari TF dan IDF.

$$W^{KEYWORD}{}_{J} = (1 + \log_{10} tf_{JK}) * (\log_{10}(N/df_{JK}))$$

1. $W_{mesin,dok1}$ = $(1 + \log_{10} 1) * (\log_{10}(4/2))$
   = $(1 + 0) * (0{,}301)$
   = 0,301
2. $W_{mesin,dok2}$ = $(1 + \log_{10} 0) * (\log_{10}(4/0))$
   = $(1 + 0) * (0)$
   = 0
3. $W_{mesin,dok3}$ = $(1 + \log_{10} 0) * (\log_{10}(4/0))$
   = $(1 + 0) * (0)$
   = 0
4. $W_{mesin,dok4}$ = $(1 + \log_{10} 1) * (\log_{10}(4/2))$
   = $(1 + 0) * (0{,}301)$
   = 0,301

Tabel 3. 6 Hasil TF-IDF pada dokumen

Kata	Dok1	Dok2	Dok3	Dok4
mesin	0,301	0,000	0,000	0,301
cari	0,965	0,000	0,000	0,000
dokumen	0,238	0,255	0,250	0,000
pengklasteran	1,071	0,000	0,000	0,000
otomatis	0,965	0,000	0,000	0,000
web	0,602	0,000	0,000	0,000
mining	0,889	0,000	0,000	0,000
kata	0,185	0,125	0,125	0,000
kunci	0,445	0,301	0,000	0,000
metode	0,185	0,000	0,163	0,163
cara	0,602	0,000	0,000	0,000
kelompok	0,392	0,000	0,555	0,000
mengklaster	0,602	0,000	0,000	0,000
dokumendokumen	0,301	0,000	0,392	0,000
centroid	0,602	0,000	0,000	0,000
linkage	0,602	0,000	0,000	0,000
hierarchical	0,602	0,000	0,000	0,000
method	0,602	0,000	0,000	0,000
clhm	0,889	0,000	0,000	0,000
jumlah	0,445	0,000	0,301	0,000
umum	0,783	0,000	0,000	0,000
inisialisasi	0,602	0,000	0,000	0,000
klaster	0,965	0,000	0,000	0,000
bentuk	0,783	0,000	0,000	0,000
kasus	0,301	0,000	0,301	0,000
user	0,602	0,000	0,000	0,000
tahu	0,602	0,000	0,000	0,000
bangun	0,301	0,000	0,301	0,000
makalah	0,125	0,125	0,000	0,125
aplikasi	0,301	0,000	0,000	0,301
valley	0,783	0,000	0,000	0,000
tracing	0,783	0,000	0,000	0,000
clustering	0,163	0,244	0,185	0,000
data	0,163	0,000	0,163	0,125
guna	0,000	0,000	0,000	0,000
hasil	0,000	0,000	0,000	0,000
proses	0,125	0,000	0,163	0,163
text	0,392	0,000	0,301	0,000
coba	0,602	0,000	0,000	0,000
pi	0,602	0,000	0,000	0,000

simpul	0,602	0,000	0,000	0,000
teknik	0,125	0,125	0,000	0,125
algoritma	0,301	0,511	0,000	0,000
tepat	0,602	0,000	0,000	0,000
tingkat	0,000	0,445	0,301	0,000
kinerja	0,000	1,071	0,000	0,000
teks	0,000	0,445	0,301	0,000
bobot	0,000	1,229	0,000	0,000
sampel	0,000	1,146	0,000	0,000
nilai	0,000	0,163	0,163	0,125
vektor	0,000	0,783	0,000	0,000
formula	0,000	0,602	0,000	0,000
similaritas	0,000	0,602	0,000	0,000
pusat	0,000	0,602	0,000	0,000
…	…	…	…	…
misal	0,000	0,301	0,000	0,301

Sumber: Perancangan

Hasil bobot setelah dinormalisasi ditunjukkan pada tabel 3.7.

Bobot dinormalisasikan dengan persamaan berikut :

$$w_{t,d} = \frac{w_{t,d}}{\sqrt{\sum_{t=1}^{n} {w_{t,d}}^2}}$$

1. $W_{mesin,dok1}$ = 0,301 / √((0,301^2)+(0,965^2)+(0,238^2)+…+(0^2))
   = 0,301/ √(0,091+0,931+0,057+…+0)
   = 0,301/4,246
   = 0,071
2. $W_{mesin,dok2}$ = 0 / √((0,0^2)+(0,0^2)+(0,255^2)+…+(0,301^2))
   = 0/ √(0+0+0,065+…+0,91)
   = 0/4,394
   = 0
3. $W_{mesin,dok3}$ = 0/ √((0 ^2)+(0^2)+(0,250^2)+…+(0^2))
   = 0/ √(0+0+0,063+…+0)
   = 0/5,430
   = 0
4. $W_{mesin,dok4}$ = 0,301 / √((0,301^2)+(0^2)+(0^2)…+(0^2))
   = 0,301/ √(0,091+0+0+…+0)

= 0,301/4,562

= 0,066

Tabel 3. 7 Hasil normalisasi *terms* pada dokumen

Kata	Dok1	Dok2	Dok3	Dok4
mesin	0,071	0,000	0,000	0,066
cari	0,227	0,000	0,000	0,000
dokumen	0,056	0,058	0,046	0,000
pengklasteran	0,252	0,000	0,000	0,000
otomatis	0,227	0,000	0,000	0,000
web	0,142	0,000	0,000	0,000
mining	0,209	0,000	0,000	0,000
kata	0,043	0,028	0,023	0,000
kunci	0,105	0,069	0,000	0,000
metode	0,043	0,000	0,030	0,036
cara	0,142	0,000	0,000	0,000
kelompok	0,092	0,000	0,102	0,000
mengklaster	0,142	0,000	0,000	0,000
dokumendokumen	0,071	0,000	0,072	0,000
centroid	0,142	0,000	0,000	0,000
linkage	0,142	0,000	0,000	0,000
hierarchical	0,142	0,000	0,000	0,000
method	0,142	0,000	0,000	0,000
clhm	0,209	0,000	0,000	0,000
jumlah	0,105	0,000	0,055	0,000
umum	0,184	0,000	0,000	0,000
inisialisasi	0,142	0,000	0,000	0,000
klaster	0,227	0,000	0,000	0,000
bentuk	0,184	0,000	0,000	0,000
kasus	0,071	0,000	0,055	0,000
user	0,142	0,000	0,000	0,000
tahu	0,142	0,000	0,000	0,000
bangun	0,071	0,000	0,055	0,000
makalah	0,029	0,028	0,000	0,027
aplikasi	0,071	0,000	0,000	0,066
valley	0,184	0,000	0,000	0,000
tracing	0,184	0,000	0,000	0,000
clustering	0,038	0,000	0,030	0,027
data	0,000	0,000	0,000	0,000
guna	0,000	0,000	0,000	0,000
hasil	0,029	0,000	0,030	0,036
proses	0,092	0,000	0,055	0,000

text	0,142	0,000	0,000	0,000
coba	0,142	0,000	0,000	0,000
pi	0,142	0,000	0,000	0,000
simpul	0,029	0,028	0,000	0,027
teknik	0,071	0,116	0,000	0,000
algoritma	0,142	0,000	0,000	0,000
tepat	0,000	0,101	0,055	0,000
tingkat	0,000	0,244	0,000	0,000
kinerja	0,000	0,101	0,055	0,000
teks	0,000	0,280	0,000	0,000
bobot	0,000	0,261	0,000	0,000
sampel	0,000	0,037	0,030	0,027
nilai	0,000	0,178	0,000	0,000
vektor	0,000	0,137	0,000	0,000
formula	0,000	0,137	0,000	0,000
similaritas	0,000	0,137	0,000	0,000
pusat	0,000	0,137	0,000	0,000
….	…	…	…	…
misal	0,000	0,069	0,000	0,066
Bobot	**6,521**	**6,578**	**8,529**	**6,767**

Sumber: Perancangan

4. Setelah didapatkan bobot proses selanjutnya adalah *clustering* CLHM.
   a. Diasumsikan setiap data dianggap sebagai *cluster*. Jika n = jumlah data dan c = jumlah *cluster*, berarti ada c = n.
   b. Menghitung jarak antar *cluster* menggunakan *euclidian distance*.
      Contoh hasil *euclidian distance:*

$$d_{(dok1,dok2)} = \sqrt{(0,071-0)^2+(0,227-0)^2+ (0,056-0,058)^2+\ldots +(0-0)^2}$$
$$= 1,397$$

$$d_{(dok1,dok3)} = \sqrt{(0,071-0)^2+(0,227-0)^2+ (0,056-0,046)^2+\ldots +(0-0)^2}$$
$$= 1,385$$

   c. Mencari 2 *cluster* yang mempunyai jarak centroid antar *cluster* yang paling minimal dan digabungkan (*merge*) ke dalam *cluster* baru (sehingga C = c-1).
   d. Kembali ke langkah 3, dan diulangi sampai dicapai *cluster* yang diinginkan.

Tabel 3. 8 Proses *clustering* CLHM

Tahap	Proses
Tahap1	dok1, dok3 = digabung ; dok2 ; dok4
Tahap2	dok1, dok3, dok2 = digabung ; dok4
Tahap3	dok1, dok3, dok2, dok4 = digabung

Sumber: Perancangan

Penjelasan tabel 3.8:

- Tahap menunjukkan proses iterasi pada setiap pembentukan *cluster*.
- Pada tahap1: dok1,dok3 digabung menjadi satu *cluster* baru karena memiliki jarak *euclidian* terkecil. Kemudian dihitung bobot *cluster* baru.
- Pada tahap2: dok2 digabung dengan dok1,dok3 karena memilki jarak terdekat.
- Pada tahap3: semua dokumen digabung menjadi satu *cluster*.

5. *Cluster* telah terbentuk, kemudian dilanjutkan dengan analisa *cluster*. Tabel hasil analisa *cluster* ditunjukkan pada tabel 3.9.
   a. Tahap pertama, pada analisis *cluster* adalah menentukan varian pada pembentukan masing – masing *cluster* (V_i). Berikut contoh perhitungan varian tiap tahap pembentukan *cluster*.

   $Vc^2 = (1 / (n_c - 1))*(\sum_{i=1}^{c}(y_i - \bar{y_c})^2)$

   Tahap1 : cluster1=dok1, dok3; cluster2=dok2; cluster3=dok4

   V_{c1}^2 = (1/(2-1))*((6,521-((6,521+8,529)/2))^2 + (8,529-(6,521+8,529)/2))^2)
   = (1/1)*((6,521-7,525)^2 + (8,529-7,525)^2)
   = 1* (1,008+ 1,008)
   = 2,016

   V_{c2}^2 = (1/(1-1))*((6,578-((6,578+6,578)/2))^2)
   = 0*(6,578-6,578)^2
   = 0

   V_{c3}^2 = (1/(1-1))*((6,767-((6,767+6,767)/2))^2)
   = 0*(6,767-6,767)^2
   = 0

b. Menghitung nilai *variance within cluster* (V_w) dan *variance between cluster* (V_b). Berikut contoh perhitungannya V_w dan V_b pada pembentukan *cluster* tahap 1:

V_w = $(1/(N-k))* \sum_{i=1}^{k}(n_i - 1).V_i^2$

= (1/4-3)*(((2-1)*2,016)+((1-1)*0)+((1-1)*0))

= (1/1)* (2,016+0+0)

= 2,016

V_b = $(1/(k-1))* \sum_{i=1}^{k} n_i * (\bar{y_i} - \bar{y})^2$

= (1/(3-1))*((2*(7,525-6,957)^2)+(1*(6,578-6,957)^2)+(1*(6,767-6,957)^2))

= (1/2)*(0,827)

= 0,413

c. Menghitung nilai varian keseluruhan (V) dengan membagi nilai V_w dan V_b. *Cluster* yang ideal adalah *cluster* yang memiliki nilai varian yang paling kecil, dan dihitung menggunakan Vw dibagi Vb. Berikut perhitungan variannya.

Tahap 1 :

$V_{iterasi1}$ = Vw/Vb

= 2,016 / 0,413 = 4,876

Tabel 3. 9 Hasil analisa varian pada *cluster*

Tahap	Vi			Vw	Vb	V
	c1, c3	c2	c4			
tahap1	2,016	0,000	0,000	2,016	0,413	4,876
tahap2	1,307		0,000	1,307	0,196	6,662
tahap3	0,921			0,921	0,000	0,000

Sumber: Perancangan

6. Setelah proses perhitungan varian selesai, dilanjutkan dengan proses *hill climbing* untuk mengetahui jumlah *cluster* yang telah terbentuk secara otomatis. Tabel 3.10 menunjukkan hasil analisa *hill climbing*. Berikut adalah perhitungan beda nilai tinggi (∂) menggunakan *hill climbing*.

∂ = $V_{i-1} - (V_i * \alpha)$

Tahap1: $\partial_{,\alpha=2}$ = 0 – (4,876*2)

= 9,751

Tahap2: $\partial_{,\alpha=2}$ = 4,876 – (6,662*2)

= 8,448

Tahap3: $\partial_{,\alpha=2}$ = 6,662 – (0*2)

= 6,662

Tabel 3. 10 Analisa *hill climbing*

Tahap	Global Optima			Beda Nilai Tinggi (∂) max		
	α=2	α=3	α=4	α=2	α=3	α=4
tahap1	x	x	x	9,751	14,627	19,503
tahap2	x	x	v	8,448	15,110	21,772
tahap3	x	x	x	6,662	6,662	6,662

Sumber: Perancangan

Dari tabel 3.10, diambil nilai (∂) yang paling optimal. *Cluster* otomatis terbentuk ketika max(∂). Tanda (v) menunjukkan *cluster* otomatis terbentuk sebanyak dua *cluster* pada tahap2 dan memiliki α = 4.

7. Proses penentuan plagiarisme dapat berjalan, jika satu dokumen dipilih dari dokumen yang telah ada pada sistem. Misalnya dokumen yang akan dibandingkan adalah dokumen01. Dokumen tersebut akan dibandingkan dengan dokumen pada satu *cluster*. Maka prosentase kemiripan dapat dilihat menggunakan *cosine similarity*. Hasil *cosine similarity* digunakan untuk menentukan prosentase kemiripan dokumen. Nilai *cosine similarity* dihitung menggunakan perhitungan bobot TF-IDF pada dokumen satu *cluster*. Hasil perhitungan *cosine similarity* ditunjukkan oleh tabel 3.11. Berikut adalah contoh perhitungan *cosine similarity*.

$$CosSim(d_j, q) = \vec{d_j} \cdot \vec{q} = \sum_{i=1}^{t} (w_{ij} \cdot w_{iq})$$

CosSim = (0,138*0,138)+(0,220*0,220)+ (0,054*0.054)+…+(0*0)

(dok1,dok1) = 1

CosSim = (0,071*0)+(0,220*0)+ (0,054*0,057)+…+(0*0,134)

(dok1,dok2) = 0,026

Tabel 3. 11 Tabel hasil *cosine similarity* pada dokumen

	Dok1	**Dok2**	**Dok3**
Dok1	1,000	0,018	0,038

Sumber: Perancangan

Diperoleh prosentase kemiripan dari hasil perhitungan *cosine similarity* dengan cara mengalikan nilai *cosine similarity*. Hasil prosentase kemiripan ditunjukkan pada tabel 3.12.

Tabel 3. 12 Prosentase kemiripan dokumen

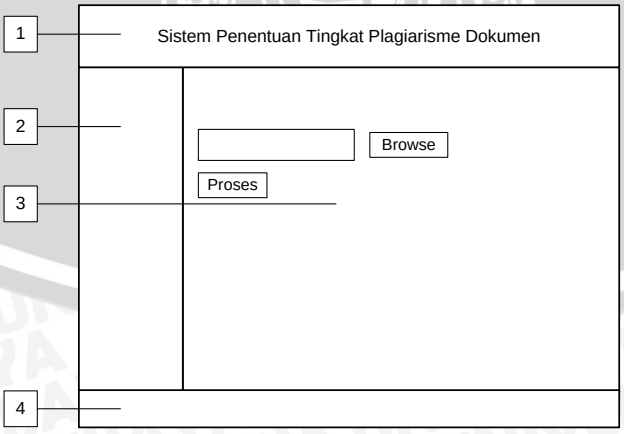
	Dok1	**Prosentase (%)**
Dok2	0,018	18,765 %
Dok3	0,038	38,907 %

Sumber: Perancangan

Hasil prosentase kemiripan dokumen01 dengan dokumen lainnya adalah dokumen02 sebesar 18,765% dan dokumen03 sebesar 38,907%. Berdasarkan hasil diatas prosentase dokumen01 lebih mirip dengan dokumen03.

3.3.7 Desain *User Interface* Sistem Penentuan Tingkat Plagiarisme Dokumen

Perancangan desain *user interface* sistem penentuan tingkat plagiarisme dokumen ini dapat dilihat pada gambar 3.18. Tujuan perancangan desain ini bertujuan agar *user* dapat nyaman dalam menggunakan aplikasi yang akan dirancang.



Gambar 3. 18 Perancangan desain *user interface* sistem

Sumber : Perancangan

Keterangan gambar 3.18:

1. *Header*
2. Menu
3. Konten
4. *Footer*

Desain *user interface* sistem terdiri dari empat komponen utama. Konten pada *user interface* terdapat menu *input file* yang bertipe data .pdf, atau .txt. Pada halaman perbandingan dokumen *user* dapat memasukkan dokumen yang akan dibandingkan dengan dokumen yang telah tersimpan pada sistem. Setelah tombol proses dijalankan, sistem akan menampilkan hasil perbandingan dokumen berupa judul dokumen dan prosentase dokumen dalam satu *cluster*.

3.5 Implementasi

Pada tahap ini akan dilakukan penerapan desain sistem dengan *coding*. Sistem ini dibangun dengan bahasa script PHP dan dikoneksikan dengan database MySQL untuk menyimpan *teks* yang telah diolah, dan tools pendukung pembuatan sistem penentuan plagiarisme dokumen penelitian. Implementasi sistem penentuan tingkat plagiarisme dokumen penelitian ini meliputi:

1. Penerapan komponen – komponen *text mining*.
2. Penerapan metode *stemming* arifin-setiono pada *text mining*
3. Penerapan metode *centroid linkage hierarchical method* (CLHM) dalam *clustering* dokumen.
4. Penerapan *hill climbing* untuk otomatisasi jumlah *cluster*.
5. Pembuatan *user interface* sistem penentuan tingkat plagiarisme dokumen penelitian untuk *end-user*.

3.6 Pengujian dan Analisa Sistem

Pengujian sistem dilakukan untuk menguji kelayakan sistem penentuan tingkat plagiarisme dokumen penelitian yang telah dibuat dan menunjukkan bahwa sistem yang telah dibuat bekerja dengan baik sesuai spesifikasi sistem yang telah ditentukan.

3.6.1 Perancangan Uji Coba

Perancangan uji coba mencakup bahan pengujian, tujuan pengujian, perancangan tabel uji coba, perancangan dokumen uji dan dokumen latih yang digunakan.

3.6.1.1 Bahan Pengujian

Data yang diuji berupa dokumen teks yang memiliki tipe data .txt dan .pdf. Data diambil dari situs di internet. Dokumen yang diuji akan diproses menggunakan *text mining*, kemudian pembobotan TF-IDF, mengelompokkan dokumen tersebut ke dalam *cluster* menggunakan *Centroid Linkage Hierarchical Method* (CLHM), dan menghitung prosentase kemiripan dokumen menggunakan *cosine similarity*.

3.6.1.2 Tujuan Pengujian

Tujuan dari pengujian sistem penentuan tingkat plagiarisme dokumen penelitian ini adalah sebagai berikut:

1. Menganalisa nilai tinggi (α) yang digunakan dalam proses *hill climbing* untuk menentukan jumlah *cluster* yang optimal.
2. Menganalisa ketepatan *clustering* menggunakan presisi, *recall*, dan *f-measure*.
3. Menganalisa hasil *similarity* dokumen penelitian menggunakan prosentase.
4. Menganalisa hasil *prosentase error* untuk mengetahui galat pada sistem.

3.6.1.3 Perancangan Tabel Uji Coba

Perancangan tabel hasil percobaan diharapkan dapat menguji ketepatan pembentukan *cluster* secara otomatis, ketepatan pengelompokan dokumen yang memiliki kesamaan, dan pengecekan kemiripan dokumen uji terhadap dokumen yang akan dibandingkan.

1. Pengujian ketepatan pembentukan *cluster* secara otomatis. Pengujian ketepatan pembentukan *cluster* pada sistem ini menggunakan *hill climbing*. Jumlah *cluster* secara otomatis terbentuk ketika nilai beda tinggi (∂) adalah nilai maksimal. Pengujian proses pembentukan *cluster* yang optimal menggunakan nilai tinggi $\alpha = 2$, $\alpha = 3$, dan $\alpha = 4$ [ELD-10].

Tabel 3. 13 Contoh tabel pengujian proses pembentukan *cluster*

No.	Jumlah Dokumen	Jumlah Cluster / Nilai beda Tinggi Max		
		α = 2	α = 3	α = 4
1.	4			
2.	8			
3.	12			
4.	16			

Sumber : Perancangan

Penjelasan tabel 3.13:

- Kolom jumlah dokumen menunjukkan jumlah dokumen yang akan dikluster.
- Kolom jumlah *cluster* dan hasil nilai beda tinggi menunjukkan jumlah *cluster* otomatis yang dibentuk dan *maximum* dari nilai beda tinggi (∂).

2. Pengujian tingkat akurasi sistem menggunakan *presisi*, *recall*, *f-measure*. Akurasi sistem penentuan tingkat plagiarisme ini dihitung menggunakan *f-measure*. *F-measure* menghitung ketepatan *output* pada sistem dengan ketepatan *output* yang diharapkan.

Tabel 3. 14 Contoh tabel pengujian *f-measure*

No.	Jumlah Dokumen	Nilai α	Max (δ)	Presisi	Recall	F-Measure
1.	4					
2.	8					
3.	12					
4.	16					

Sumber : Perancangan

Penjelasan tabel 3.14:

- Kolom jumlah dokumen menunjukkan jumlah dokumen yang diproses.
- Kolom nilai α menunjukkan nilai tinggi yang akan digunakan.
- Kolom nilai Max (δ) menunjukkan nilai tinggi maksimum yang digunakan.
- Kolom presisi menunjukkan jumlah dokumen yang relevan pada sistem dalam proses pengklusteran dokumen.
- Kolom *recall* menunjukkan jumlah dokumen yang muncul pada sistem dalam proses pengklusteran dokumen.
- Kolom *f-measure* menunjukkan hasil perhitungan presisi dan *recall*.

3. Pengujian *similarity* dokumen. Prosentase *similarity* adalah penentuan kemiripan dokumen uji dengan dokumen pada *database* yang diukur dengan angka.

Tabel 3. 15 Contoh tabel pengujian *similarity* plagiarisme

No.	Dokumen Uji	Dokumen Latih	Prosentase kemiripan (Pemotongan kata sebelum *stemming*) (%)	Prosentase kemiripan (Pemotongan kata setelah *stemming*) (%)
1.	Dok1	100% sama		
2.		75% sama		
3.		50% sama		
4.		25% sama		

Sumber : Perancangan

Penjelasan tabel 3.15:

- Kolom dokumen uji menunjukkan dokumen untuk menguji dan membandingkan dengan dokumen lainnya dalam satu *cluster*.
- Kolom dokumen latih menunjukkan dokumen yang akan dibandingkan dengan dokumen uji.
- Kolom prosentase menunjukkan hasil prosentase kemiripan dokumen uji dan dokumen latih.

4. Pengujian prosentase *error*. Prosentase *error* yaitu pengujian terhadap sistem dengan menghitung nilai *similarity* (kemiripan) yang dihasilkan sistem dibandingkan nilai *similarity* yang diharapkan sehingga dapat dilihat apakah sistem yang dibuat telah sesuai dengan hasil yang diharapkan.

Tabel 3. 16 Contoh tabel pengujian prosentase *error*

No	Dokumen Uji	Pemotongan kata sebelum di *stemming*		Pemotongan kata setelah di *stemming*	
		Similarity (%)	Prosentase *error* (%)	*Similarity* (%)	Prosentase *error* (%)
1	dok1				
2	dok2				
3	dok3				
4	dok4				

Sumber: Perancangan

Penjelasan tabel 3.16

- Kolom dokumen uji menunjukkan dokumen untuk menguji dan membandingkan dengan dokumen lainnya dalam satu *cluster*.
- Kolom *similarity* menunjukkan prosentase kemiripan dokumen dari hasil *output* sistem.
- Kolom prosentase *error* menunjukkan hasil galat antara dokumen uji dan dokumen latih.

3.6.1.4 Perancangan Dokumen Uji dan Dokumen Latih

Dokumen latih yang digunakan untuk penelitian ini ada beberapa jenis dokumen, ketentuan dari dokumen latih yang digunakan adalah sebagai berikut:

1. Dokumen latih dalam satu *cluster* yang sama dengan dokumen uji.
2. 100% kata sama: adalah dokumen latih yang memiliki isi teks sama dengan dokumen uji
3. 25% kata: adalah dokumen uji yang isi teksnya dihilangkan sebanyak 25% kata secara acak sehingga menghasilkan 75% kata yang sama.
4. 50% kata: adalah dokumen uji yang isi teksnya dihilangkan sebanyak 50% kata secara acak sehingga menghasilkan 50% kata yang sama
5. 75% kata: adalah dokumen uji yang isi teksnya dihilangkan sebanyak 75% kata secara acak sehingga menghasilkan 25% kata yang sama.

Jenis dokumen latih adalah dokumen uji yang telah dirubah sesuai kebutuhan pengujian. Tujuan dari mengubah dokumen uji untuk mengecek apakah sistem yang telah dibuat telah sesuai. Dokumen uji diambil satu dokumen dari masing-masing *cluster* untuk dibandingkan dengan dokumen lain pada *cluster* yang sama dengan dokumen uji. Berikut ini adalah contoh data dokumen uji dan dokumen-dokumen latih.

a. Dokumen uji (dok2):

PENINGKATAN KINERJA CLUSTERING DOKUMEN TEKS MENGGUNAKAN PEMBOBOTAN SAMPEL Algoritma clustering berbasis pembobotan sampel (sample weighting) saat ini banyak diteliti. Ada beberapa model pembobotan yang pada prinsipnya bertujuan untuk merubah nilai vektor sampel dan formula similaritas vektor sampel dengan pusat clusternya. Dalam dokumen teks pembobotan dapat berupa

konektifitas antar dokumen, misalnya dalam dokumen akademik yang ada koneksi referensi. Namun dalam dokumen berita koneksi referensi mungkin jarang ditemukan. Dalam makalah ini teknik pembobotan baru diajukan, yaitu menggunakan kata-kata yang muncul dalam kata kunci (keyword) dan judul (title) dari suatu dokumen teks. Eksperimen dilakukan terhadap abstrak dokumen akademik sebanyak 500 dokumen dan dokumen berita. Sebanyak 3000 dokumen Algoritma yang diuji kinerjanya adalah algoritma K-Means clustering dan algoritma Fuzzy C-Means clustering. Parameter kinerja algoritma digunakan nilai F-measure dari hasil clustering sebelum dilakukan pembobotan sampel dan setelah dilakukan pembobotan sampel. Hasil eksperimen menunjukkan bahwa pembobotan sampel dapat meningkatkan kinerja clustering sebesar 12,8% untuk pembobotan dengan keyword dan title dan meningkatkan kinerja clustering 9.8% untuk pembobotan dengan title saja. clustering dokumen, pembobotan sampel, kinerja clustering, F-measure

b. Dokumen latih dok2-00 : dokumen latih yang sesuai dengan dokumen asli.

PENINGKATAN KINERJA CLUSTERING DOKUMEN TEKS MENGGUNAKAN PEMBOBOTAN SAMPEL Algoritma clustering berbasis pembobotan sampel (sample weighting) saat ini banyak diteliti. Ada beberapa model pembobotan yang pada prinsipnya bertujuan untuk merubah nilai vektor sampel dan formula similaritas vektor sampel dengan pusat clusternya. Dalam dokumen teks pembobotan dapat berupa konektifitas antar dokumen, misalnya dalam dokumen akademik yang ada koneksi referensi. Namun dalam dokumen berita koneksi referensi mungkin jarang ditemukan. Dalam makalah ini teknik pembobotan baru diajukan, yaitu menggunakan kata-kata yang muncul dalam kata kunci (keyword) dan judul (title) dari suatu dokumen teks. Eksperimen dilakukan terhadap abstrak dokumen akademik sebanyak 500 dokumen dan dokumen berita. Sebanyak 3000 dokumen Algoritma yang diuji kinerjanya adalah algoritma K-Means clustering dan algoritma Fuzzy C-Means clustering. Parameter kinerja algoritma digunakan nilai F-measure dari hasil clustering sebelum dilakukan pembobotan sampel dan setelah dilakukan pembobotan sampel. Hasil eksperimen menunjukkan bahwa pembobotan sampel dapat meningkatkan kinerja clustering sebesar 12,8% untuk pembobotan dengan keyword dan title dan meningkatkan kinerja clustering 9.8% untuk pembobotan dengan title saja. clustering dokumen, pembobotan sampel, kinerja clustering, F-measure

c. Dokumen latih dok2-25 : dokumen latih dengan menghilangkan 25% kata.

PENINGKATAN KINERJA CLUSTERING DOKUMEN TEKS MENGGUNAKAN PEMBOBOTAN SAMPEL Algoritma clustering berbasis banyak diteliti. Ada beberapa model pembobotan yang pada prinsipnya untuk merubah nilai vektor sampel dan formula similaritas vektor sampel dengan pusat clusternya. dapat berupa konektifitas antar dokumen, misalnya dalam dokumen akademik yang ada koneksi referensi. berita koneksi referensi mungkin jarang ditemukan. Dalam makalah ini teknik diajukan, yaitu kata-kata yang muncul dalam kata kunci (keyword) dan judul (title) dari suatu dokumen teks. Eksperimen dilakukan akademik sebanyak 500 dokumen dan dokumen berita. K-

Means clustering dan algoritma Fuzzy C-Means clustering. Parameter kinerja algoritma digunakan nilai F-measure dari hasil clustering pembobotan sampel dan setelah dilakukan pembobotan sampel. Hasil eksperimen pembobotan sampel dapat sebesar 12,8% untuk pembobotan dengan dan meningkatkan kinerja clustering 9.8% untuk pembobotan dengan title saja. clustering dokumen, kinerja clustering, F-measure

d. Dokumen latih dok2-50 : dokumen latih dengan menghilangkan 50% kata.

PENINGKATAN KINERJA CLUSTERING DOKUMEN TEKS MENGGUNAKAN PEMBOBOTAN SAMPEL Algoritma clustering banyak diteliti. Ada beberapa model pembobotan prinsipnya untuk merubah nilai vektor vektor sampel dengan pusat clusternya. antar dokumen, akademik yang ada koneksi referensi. berita koneksi referensi. Dalam makalah ini teknik diajukan, yaitu kata kunci (keyword) dan (title) dari suatu dokumen teks. Eksperimen dilakukan akademik sebanyak 500 dokumen dan berita. K-Means clustering dan algoritma Fuzzy C-Means clustering. nilai hasil clustering pembobotan sampel dan. 12,8% untuk pembobotan dengan kinerja clustering 9.8% title saja. clustering dokumen, kinerja clustering, F-measure

e. Dokumen latih dok2-75 : dokumen latih dengan pemotongan 75% kata.

PENINGKATAN CLUSTERING DOKUMEN SAMPEL Algoritma clustering banyak diteliti. Ada beberapa model pembobotan untuk vektor dengan pusat clusternya. yang ada koneksi. berita koneksi. diajukan, yaitu kata kunci dari suatu dokumen teks. Eksperimen sebanyak 500 dokumen. clustering Fuzzy C-Means clustering. hasil pembobotan sampel 12,8% untuk pembobotan title saja. kinerja clustering, F-measure

Analisis terhadap sistem dilakukan untuk menarik kesimpulan dari hasil pembuatan aplikasi dan pengujian sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM). Analisa dan evaluasi sistem penentuan tingkat plagiarisme dokumen penelitian ini bertujuan untuk menguji kelayakan sistem.

3.7 Penarikan Kesimpulan

Penarikan kesimpulan dilakukan setelah semua tahapan perancangan, implementasi dan pengujian sistem aplikasi telah selesai dilakukan. Kesimpulan diambil dari hasil pengujian dan analisa terhadap sistem yang dibangun. Tahap terakhir dari penulisan adalah saran. Saran bertujuan untuk memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan aplikasi selanjutnya.

BAB IV
IMPLEMENTASI

Bab ini membahas implementasi perangkat lunak berdasarkan hasil yang telah diperoleh dari analisa kebutuhan dan proses perancangan perangkat lunak yang dibuat pada bab III. Bab ini terdiri dari penjelasan tentang spesifikasi sistem, batasan-batasan dalam implementasi, implementasi algoritma, dan implementasi antarmuka.

4.1 Spesifikasi Sistem

Hasil analisis kebutuhan dan perancangan perangkat lunak yang telah diuraikan pada bab III menjadi acuan untuk melakukan implementasi menjadi sistem yang dapat berfungsi sesuai dengan kebutuhan. Spesifikasi sistem diimplementasikan pada spesifikasi perangkat keras dan perangkat lunak.

4.1.1 Spesifikasi Perangkat Keras

Pengembangan sistem penentuan tingkat plagiarisme dokumen penelitian ini menggunakan sebuah komputer dengan spesifikasi perangkat keras yang dijelaskan pada Tabel 4.1.

Tabel 4. 1 Spesifikasi Perangkat Keras Komputer

Nama Komponen	Spesifikasi
Prosesor	Intel(R) Core(TM) i5-2410M CPU @ 2.30GHz
Memori (RAM)	2 GB
Hardisk	640 GB
VGA	Nvidia geforce GT540M cuda(TM) 2GB

Sumber: Implementasi

4.1.2 Spesifikasi Perangkat Lunak

Pengembangan sistem penentuan tingkat plagiarisme dokumen penelitian ini menggunakan perangkat lunak dengan spesifikasi yang dijelaskan pada Tabel 4.2.

Tabel 4. 2 Spesifikasi Perangkat Lunak Komputer

Nama	Spesifikasi
Sistem Operasi	Microsoft Windows 7 Ultimate 64-bit
Bahasa Pemrograman	HTML dan PHP 5.3.1
Tools pemrograman	Adobe Dreamwaver CS3 versi 9.0
Server localhost	XAMPP versi 1.7.3
DBMS	MySQL versi 5.5.4
Web Browser	Google Chrome versi 25.0.1364.97m

Sumber: Implementasi

4.2 Batasan-Batasan Implementasi

Batasan implementasi adalah batasan proses yang dapat dilakukan sistem sesuai dengan perancangan awal sistem. Batasan implementasi ditampilkan agar penelitian ini memiliki ruang lingkup yang jelas dalam mengimplementasikan sistem. Beberapa batasan dalam mengimplementasikan sistem penentuan tingkat plagiarisme dokumen penelitian adalah sebagai berikut:

- Sistem penentuan tingkat plagiarisme dokumen penelitian dirancang dan dijalankan menggunakan *web application*.
- Metode penyelesaian masalah yang digunakan adalah *Centroid Linkage Hierarchical Method* (CLHM).
- Dokumen yang digunakan adalah skripsi atau tugas akhir.
- Bagian dokumen yang digunakan adalah judul, abstrak dan *keyword*.
- Dokumen yang digunakan merupakan dokumen berbahasa Indonesia.
- Data yang digunakan bertipe data .txt, dan .pdf.
- Output yang dikeluarkan berupa hasil *cluster*, dan prosentase kemiripan dokumen.
- Dokumen yang sudah terkluster akan disimpan dalam database.
- Dokumen yang sudah terkluster dapat dihapus, tetapi dokumen tidak dapat diubah.
- Penentuan tingkat plagiarisme dokumen berdasarkan pada frekuensi kemunculan kata, bukan pada struktur *semantic*.

4.3 Implementasi Algoritma

Sistem penentuan tingkat plagiarisme dokumen ini mempunyai beberapa proses utama, yaitu proses *input* dokumen dan proses mengecek kemiripan dokumen pada sekelompok *cluster* yang telah terbentuk.

4.3.1 Implementasi Algoritma Proses *Input* Dokumen

Proses *input* dokumen dilakukan dengan memasukkan *input* berupa judul, abstrak, dan *keyword* yang ada pada abstrak. Format *input* memiliki tipe data .pdf, atau .txt. Data yang dimasukkan akan diolah untuk mendapatkan informasi judul dan *term*, kemudian data diolah dengan *text processing*, menghitung bobot dokumen menggunakan TF-IDF, dan melakukan pengklasteran. Tabel daftar fungsi sistem penentuan kemiripan dokumen penelitian.

Tabel 4. 3 Daftar fungsi pada sistem

No.	Proses	Fungsi	Keterangan
1.	Text Mining / Propoccesing	- function token()	Fungsi untuk menghilangkan tanda baca seperti koma, titik, petik dua, dan lain-lain.
		- function parsing()	Fungsi untuk memecah kalimat menjadi kata yang berdiri sendiri.
		- function stopword()	Fungsi untuk menghilangkan kata yang tidak penting seperti ‘yang’,’adalah’,’itu’, dan lain – lain.
		- function stemming()	Fungsi untuk memecah kata menjadi kata dasar
2.	Pembobotan TF-IDF	- function updateTfIdf()	Fungsi untuk menghitung *Term Frequency* (tf) dan jumlah kemunculan kata pada semua dokumen (df).
		- function bobotTfIdf()	Fungsi untuk menghitung pembobotan tiap kata pada dokumen dan menghitung bobot antar dokumen.
		- Function normalisasi()	Fungsi untuk menyederhanakan bobot dokumen.
3.	Clustering CLHM	- function eucledian()	Fungsi untuk menghitung jarak terdekat antar dokumen.
		- function searchMinIndex()	Fungsi untuk mencari index dari dokumen yang memiliki jarak

			terdekat.
		- function combineArrayIndex()	Fungsi untuk menggabungkan bobot dokumen - dokumen yang telah menjadi satu *cluster*.
		- function labelingData()	Fungsi untuk memberi label nilai *cluster* dokumen – dokumen pada tiap tahap pembentukan *cluster*.
4.	Analisa Cluster	- function analisaCluster()	Fungsi untuk menghitung nilai varian pada tiap tahap pembentukan *cluster*.
5.	Hill Climbing	- function hillClimbing()	Fungsi untuk menghitung nilai beda tinggi pada tiap tahap pembentukan *cluster* berdasarkan nilai perhitungan varian..
		- function searchMaxIndex()	Fungsi untuk mencari nilai optimal pada pembentukan *cluster*.

Sumber: Implementasi

4.3.1.1 Proses *Text Mining*

Proses ini tahap pertama dalam menggali informasi suatu dokumen. Tahap ini terdiri dari *tokenizing, filtering*, dan *stemming*. *Tokenizing* berfungsi untuk menghilangkan tanda baca pada dokumen, dan memecah kalimat menjadi kata yang berdiri sendiri. *Filtering* berguna untuk menghilangkan kata-kata yang tidak penting dan mengambil kata penting menjadi *keyword*. Sedangkan *stemming* mengubah kata penting pada dokumen agar menjadi bentuk kata dasar. *Stemming* yang digunakan adalah *stemming* arifin-setiono. Pemanggilan fungsi *text preprocessing* secara keseluruhan ditunjukkan pada *sourcecode* 4.1.

```
1    //Tokenizing
2  $hasil_token = token($isi_dokumen);
3    //Parsing
4  $hasil_parsing = parsing($hasil_token);
5    //StopwordRemoval
6  $hasil_stopword = stopword($hasil_parsing);
7    //Stemming
8  $hasil_stemming = stemming($hasil_stopword);
9  mysql_query("update daftar_dokumen set stemming =
10 '$hasil_stemming' where nama_dokumen = '$nama_dokumen'");
```

Sourcecode 4. 1 Implementasi algoritma *text mining* pada dokumen

Sumber: Implementasi

Fungsi *token* dan fungsi *parsing* merupakan tahap *tokenizing*. Fungsi *token* menghilangkan tanda baca, sedangkan *parsing* memecah kata berdasarkan spasi (*white space*).

```
1  function token($data){
2
3    $isi_dokumen1 = strtolower($data);
4    $ubah=array
5  ("`","~","!","@","#","$","%","^","&","*","(",")","-
6  ","_","+","=","{","}","|","[","]",":",";","'","<",">",".",","
7  ,"?","/","\"","0","1","2","3","4","5","6","7","8","9","<br/>"
8  ,"<br/>","<strong>","</strong>","<span>","</span>","<p>","</p
9  >"," ","<!-- pagebreak --
10 >","<em>","</em>","&quot;","\\","\n","\r");
11   $jadi=array (" ");
12   $token = str_ireplace($ubah, $jadi, $isi_dokumen1);
13   return $token;
14 }
```

Sourcecode 4. 2 Implementasi algoritma *tokenizing*
Sumber: Implementasi

Fungsi *token* yaitu merubah semua huruf menjadi huruf kecil dengan menggunakan *strtolower*, kemudian menghilangkan tanda baca dan simbol yang telah ditentukan dalam variabel bertipe *array*. Jika ditemukan maka simbol atau tanda baca pada teks akan dihapus.

```
1  function stopword($stopword){
2
3   for ($j=0; $j < count($stopword); $j++){
4    if($stopword[$j] != ''){
5     $sql = mysql_query("select daftar_stopword from stopword
6  where daftar_stopword = '$stopword[$j]'");
7     if (mysql_num_rows($sql)== 1){
8      $hasil_stopword = "stopword.txt";
9      $fh = fopen($hasil_stopword, 'a+') or die ("File tidak
10 dapat dibuka");
11     fwrite($fh, "");
12     fclose($fh);
13    }
14    else{
15     $hasil_stopword = "stopword.txt";
16     $fh = fopen($hasil_stopword, 'a+') or die ("File tidak
17 dapat dibuka");
18     fwrite($fh, $stopword[$j]." ");
19     fclose($fh);
20    }
```

```
21      }
22    }
23      $baca_isi = "stopword.txt";
24      $fh = fopen($baca_isi, 'r+') or die ("file tidak dapat
25  dibuka");
26      $isi_stopword = fread($fh, filesize($baca_isi));
27      fclose ($fh);
28      return $isi_stopword;
29    }
30  }
```

Sourcecode 4. 3 Implementasi algoritma *stopword removal*
Sumber: Implementasi

Tahap *filtering* pada sistem ini ditunjukkan dengan fungsi *stopword*. Tahap pertama proses *stopword* adalah mengambil tiap kata pada dokumen kemudian dibandingkan dengan kamus *stopword* pada *database*, jika ditemukan kata tidak penting maka akan dihapus, sehingga nantinya akan menghasilkan kata unik yang disebut sebagai *keyword*. Setelah dilakukan proses *filtering* maka akan dilanjutkan dengan proses *stemming*. *Stemming* yaitu suatu proses untuk mengubah bentuk kata menjadi kata dasar. *Stemming* yang digunakan dalam penelitian ini adalah *stemming* Arifin-Setiono.

4.3.1.2 Proses Pembobotan TF-IDF

Proses ini pembobotan TF-IDF berfungsi untuk mengetahui bobot tiap dokumen. Hasil dari pembobotan TF-IDF nantinya akan digunakan sebagai nilai dari masing-masing dokumen untuk menentukan kedekatan antar dokumen pada proses *clustering*. *Source code* 4.3 menunjukan proses pembobotan dokumen.

```
1   function updateTfIdf($file, $nama_dokumen){
2     $data = explode(" ", $file);
3     for ($l=0; $l < count($data); $l++){
4       $tf_df = mysql_query("select $nama_dokumen doc, df from
5   dokumen where kata = '$data[$l]'");
6       while ($cek = mysql_fetch_array($tf_df)){
7         $jumlah_tf = $cek['doc'];
8         $nilai_df = $cek['df'];
9         if($jumlah_tf == 0){
10          $total_df = $nilai_df + 1;
11          $jumlah_tf = 1;
12        }
13        else{
14          $total_df = $nilai_df;
```

```
15      $jumlah_tf = $jumlah_tf + 1;
16     }
17     mysql_query("update dokumen set $nama_dokumen =
18  $jumlah_tf, df = $total_df where kata = '$data[$l]'");
19    }
20   }
21  }
```

Sourcecode 4. 4 Implementasi algoritma *update* nilai tf dan df pada dokumen
Sumber: Implementasi

Fungsi *update* tf dan df merupakan proses pengambilan informasi pada dokumen. *Update* tf untuk menghitung frekuensi kemunculan kata pada dokumen, sedangkan *update* df adalah proses untuk menghitung kemunculan kata pada keseluruhan dokumen. Setelah didapat nilai tf dan nilai df, nilai tersebut akan di *update* kedalam *database* dan proses selanjutnya adalah menghitung bobot kata pada masing – masing dokumen.

```
1   function bobotTfIdf(){
2   $daftar_kata = mysql_query("select kata from dokumen");
3    while ($tampil = mysql_fetch_array($daftar_kata)){
4     $kata = $tampil['kata'];
5     $judul = mysql_query("select nama_dokumen from
6   daftar_dokumen");
7     while ($nama = mysql_fetch_array($judul)){
8      $nama_doku = $nama['nama_dokumen'];
9      $count = mysql_query("select count(id_dokumen) jumlah from
10  daftar_dokumen");
11     $cek = mysql_fetch_array($count);
12     $jumlah_dokumen = $cek['jumlah'];
13     $hitung = mysql_query("select $nama_doku tf, df from
14  dokumen where kata = '$kata'");
15     while ($hasil = mysql_fetch_array($hitung)){
16      $tf = floatval($hasil['tf']);
17      $df = floatval($hasil['df']);
18      if ($tf == 0)
19       $Wkj = 0;
20      else
21       $Wkj = abs((1+log10($tf))*(log10($jumlah_dokumen/$df)));
22       mysql_query("update bobot_dokumen set $nama_doku = $Wkj
23  where kata = '$kata'");
24     }
25    }
26   }
27  }
```

Sourcecode 4. 5 Implementasi algoritma pembobotan TF-IDF pada dokumen
Sumber: Implementasi

Fungsi bobotTfIdf merupakan proses perhitungan bobot kata pada tiap – tiap dokumen. Bobot akan dinormalisasi menjadi bentuk yang lebih sederhana. Hasil normalisasi akan digunakan untuk perhitungan jarak *euclidian* dan menghitung kemiripan dokumen menggunakan *cosine similarity*. Perhitungan normalisasi ditunjukkan pada sourcecode 4.6.

```
function normalisasi(){
  //normalisasi Wkj (bobot kata dalam dokumen)
 $nama = mysql_query("select nama_dokumen from
daftar_dokumen");
 while ($cek = mysql_fetch_array($nama)){
  $nama_dokumen = $cek['nama_dokumen'];
  $bobot = mysql_query("select $nama_dokumen doc0 from
bobot_dokumen");
  $jumlah = 0;
  while ($cek = mysql_fetch_array($bobot)){
   $pow = pow($cek['doc0'],2);
   $pangkat = (floatval($pow));
   $jumlah += floatval($pangkat);
  }
  $sqrt = sqrt($jumlah);
  $bawah = floatval($sqrt);
  $daftar_kata = mysql_query("select kata from dokumen");
  while ($tampil = mysql_fetch_array($daftar_kata)){
   $kata = $tampil['kata'];
   $hitung1 = mysql_query("select $nama_dokumen doc1 from
bobot_dokumen where kata = '$kata'");
   while ($cek1 = mysql_fetch_array($hitung1)){
    $atas = $cek1['doc1'];
    if($atas==0 || $bawah==0)
     $normalisasi = 0;
    else
     $normalisasi = abs($atas/$bawah);
    mysql_query("update bobot_dokumen set $nama_dokumen =
$normalisasi where kata = '$kata'");
   }
  }
 }
}
```

Sourcecode 4. 6 Implementasi algoritma normalisasi dokumen
Sumber: Implementasi

4.3.1.3 Proses Pengolahan Data *Pre-Clustering*

Proses pengolahan data sebelum dikluster adalah proses mengubah bentuk dokumen menjadi *array* tiga dimensi dimana *index* kesatu merupakan dokumen

keseluruhan. *Index* kedua merupakan pemecahan data berdasarkan nama dokumen dan kumpulan *term* dokumen. Bobot masing-masing *term* dan nama dokumen berada pada *index* ketiga. Gambaran struktur data beserta contoh dapat dilihat pada bab II. *Sourcecode* 4.7 merupakan hasil pengolahan informasi dokumen yang diambil dari *database* dan mengubahnya menjadi *array* tiga dimensi. Data berupa bobot dokumen yang sudah ternormalisasi diambil dari *database* kemudian diolah untuk mendapat informasi. Data ini nantinya akan memudahkan perhitungan dalam pencarian jarak anatar *cluster*.

```
1   function preCluster(){
2
3    $data = array();
4    $sql1 = mysql_query("select nama_dokumen from
5   daftar_dokumen");
6    while ($data1 = mysql_fetch_array($sql1)){
7     $identitas = $data1['nama_dokumen'];
8     $nilai = array();
9     $sql2 = mysql_query("select kata from dokumen");
10    while ($data2 = mysql_fetch_array($sql2)){
11     $kata = $data2['kata'];
12     $sql3 = mysql_query("select $identitas doc from
13  bobot_dokumen where kata = '$kata'") or die (mysql_error());
14     $data3 = mysql_fetch_array($sql3);
15     $bobot = $data3['doc'];
16     $nilai[] = $bobot;
17    }
18    $gab = array($nilai,array("$identitas"));
19    $data[] = $gab;
20    unset($gab);
21    unset($nilai);
22   }
23   return $data;
24  }
```

Sourcecode 4. 7 Implementasi algoritma struktur data *pre-clustering*
Sumber: Implementasi

4.3.1.4 Proses *Clustering* CLHM

Proses *clustering* CLHM digunakan untuk mengelompokkan dokumen yang memiliki kemiripan kata. Proses ini berjalan setelah proses pengolahan data *pre-cluster* selesai. Proses ini menggabungkan satu dokumen yang dianggap

sebagai *cluster* menjadi suatu *cluster* baru pada setiap iterasi. Sehingga jumlah iterasi sama dengan banyak dokumen dikurangi satu.

```
function eucledian($a, $b){

  $count = count($a);
  $sum = 0;
  for($i=0;$i<$count; $i++){
      $sum+=pow(abs($a[$i]-$b[$i]),2);
  }
  return sqrt($sum);
}
  //fungsi pencarian jarak terpendek pada tiap cluster
function searchMinIndex($array){
  $sorter = array();
  $key    = array();
  reset($array);
  for ($i=0; $i < count($array)+1 ; $i++) {
    for ($j=0+$i; $j < count($array)+1 ; $j++) {
    if($i != $j){
        array_push($key, array($i,$j));
        array_push($sorter, $array[$i][$j]);
      }
    }
  }
  asort($sorter);
  $res = array();
  foreach ($sorter as $k => $value) {
      $res = $key[$k];
        break;
  }
  return $res;
}
  //fungsi pengklasteran data menjadi data baru
function combineArrayIndex($data, $a, $b){
  $class = array();
  $count = count($data);
  $val = array();
  for($i=0;$i<count($data[$a][0]); $i++){
      $val[$i] = ($data[$a][0][$i] + $data[$b][0][$i])/2;
  }
  foreach ($data[$a][1] as $key => $value) {
      array_push($class, $value);
  }
  foreach ($data[$b][1] as $key => $value) {
      array_push($class, $value);
  }
  unset($data[$a]);
  unset($data[$b]);
```

```
47    array_push($data, array($val,$class));
48    $data = array_values($data);
49    return($data);
50  }
51    //fungsi pemberian label tiap anggota cluster function
52  labelingData($data){
53    $a = 0;
54    $clus = 0;
55    foreach ($data as $key => $val) {
56        $clus++;
57        foreach($val[1] as $key2 => $val2) {
58          $temp = $val2;
59          mysql_query("update daftar_dokumen set cluster = $clus
60  where nama_dokumen = '$temp'");
61        }
62    }
63  }
```

Sourcecode 4. 8 Implementasi algoritma *clustering* menggunakan CLHM
Sumber: Implementasi

Sourcecode 4.8 merupakan fungsi – fungsi yang digunakan pada proses *clustering* menggunakan *centroid linkage hierarchical method* (CLHM). Prinsip kerja *clustering* CLHM ini adalah dua dokumen yang memiliki jarak minimal pada tiap iterasi digabung menjadi satu *cluster*. Dokumen yang memiliki satu *cluster* dihitung titik pusat sebagai pusat *cluster* baru. Proses tersebut berulang sampai di dapat satu *cluster*. Proses iterasi berjalan sebanyak jumlah dokumen dikurangi satu (n-1).

Fungsi *eucledian* merupakan fungsi untuk menghitung jarak antar dokumen menggunakan *euclidian distance*. Perhitungan jarak digunakan untuk mengecek kedekatan dokumen berdasarkan jarak kata (*term*) yang dimiliki oleh dua dokumen. Fungsi *searchMinIndex* digunakan untuk mengambil jarak dokumen minimal setelah proses *euclidian distance*. Fungsi *combineArrayIndex* merupakan fungsi untuk menggabungkan bobot dokumen yang memiliki jarak *euclidian* yang minimal pada tiap iterasi. Fungsi *combineArrayIndex* bertujuan untuk menentukan pusat *cluster* baru pada dokumen yang telah digabungkan. Setelah proses pengklusteran pada tiap iterasi selesai kemudian dilakukan proses pemberian label pada anggota *cluster*.

4.3.1.5 Proses Analisa *Cluster*

Proses analisa varian digunakan untuk mrnghitung kepadatan *cluster* yang dibentuk (*cluster dencity*). Kepadatan suatu *cluster* dapat dihitung menggunakan *variance within cluster* (V_w) dan *variance between cluster* (V_b). *Sourcecode* analisa *cluster* ditunjukkan pada *sourcecode* 4.9.

```
function analisaCluster($data){

  $sql1 = mysql_query("select max(cluster) maxim,
count(id_dokumen) jumlah_data from daftar_dokumen");
  $hasil1 = mysql_fetch_array($sql1);
  $jumlah_data = $hasil1['jumlah_data'];
  $max = $hasil1['maxim'];
  for ($x=1; $x <= $max; $x++){
    $sql2 = mysql_query("select count(id_dokumen)
jumlah_cluster, round(sum(bobot),3) jumlah_bobot from
daftar_dokumen where cluster = '$x'");
    $hasil2 = mysql_fetch_array($sql2);
    $jumlah = $hasil2['jumlah_cluster'];
    $jumlah_bobot = $hasil2['jumlah_bobot'];
    $rerata = $jumlah_bobot/$jumlah;
    $sql3 = mysql_query("select bobot from daftar_dokumen
where cluster = '$x'");
    while($hasil3 = mysql_fetch_array($sql3)){
      $bobot = $hasil3['bobot'];
      //menghitung nilai varian tiap cluster
      $var += pow(($bobot-$rerata),2);
    }
    if ($jumlah == 1)
      //nilai varian tiap cluster jika dokumen=1
      $varian_c = 0;
    else
      //nilai varian tiap cluster
      $varian_c = $var /($jumlah-1);

      $temp_vw += ($jumlah-1)*$varian_c;
      $rerata_total += $rerata;
  }//end for
  //Perhitungan Vw
  $vw = $temp_vw / ($jumlah_data-$max);
  //Perhitungan Vb
  $rerata_all_dok = $rerata_total/$max;
  for ($x=1; $x <= $max; $x++){
    $sql4 = mysql_query("select count(id_dokumen)
jumlah_cluster, round(sum(bobot),3) jumlah_bobot from
daftar_dokumen where cluster = '$x'");
    $hasil4 = mysql_fetch_array($sql4);
```

```
42      $jumlah1 = $hasil4['jumlah_cluster'];
43      $jumlah_bobot1 = $hasil4['jumlah_bobot'];
44      $rerata1 = $jumlah_bobot1/$jumlah1;
45      $temp_vb += $jumlah*pow(($rerata1-$rerata_all_dok),2);
46    }
47    if($max==1)
48      $vb = 0;
49    else
50      $vb = $temp_vb / ($max-1);
51    //Perhitungan Varian tiap Cluster
52    if($vb==0)
53      $varian = 0;
54    else
55      $varian = round($vw/$vb,3);
56      return $varian;
37  }
```

Sourcecode 4. 9 Implementasi algoritma proses analisa *cluster*
Sumber: Implementasi

Fungsi *analisaCluster* merupakan fungsi untuk menghitung nilai varian pada tiap tahap pembentukan *cluster*. Tahap pertama dalam analisa *cluster* adalah menghitung jumlah *cluster* dan jumlah dokumen yang diambil dari *database*. Kemudian dilanjutkan dengan proses pengambilan bobot dokumen dan perhitungan nilai varian pada tiap tahap pembentukan *cluster*. Proses tersebut diulangi dari *cluster* satu sampai sebanyak jumlah *cluster*. Setelah didapatkan nilai varian pada tahap pembentukan *cluster* pada satu iterasi kemudian dilakukan proses perhitungan nilai *variance within cluster* (Vw) dan perhitungan *variance between cluster* (Vb). Setelah itu dilakukan proses perhitungan nilai varian pada tiap iterasi. Nilai varian tiap iterasi telah selesai kemudian dilakukan perhitungan nilai tinggi.

4.3.1.6 Proses Hill Climbing

Proses *hill climbing* digunakan untuk mengetahui jumlah *cluster* yang optimal. Pada proses *hill climbing* ini menggunakan nilai tinggi (α) = 2, 3, dan 4. Ketiga nilai tinggi ini digunakan untuk menentukan seberapa mungkin metode ini mencapai *global optimum*. *Global optimum* digunakan untuk membentuk *cluster* secara otomatis.

```
1  //fungsi perhitungan beda nilai tinggi
2  //nilai alpha 2, 3, dan 4
3  function hillClimbing($m, $n){
4    $all_delta = array();
5    unset($all_delta);
6    for($alpha=2; $alpha<5; $alpha++){
7        $delta = $n - ($m*$alpha);
8        $all_delta[] = $delta;
9    }
10
11   return $all_delta;
12 }
```

Sourcecode 4. 10 Implementasi algoritma proses *hill climbing*
Sumber: Implementasi

Fungsi *hillClimbing* merupakan proses menghitung beda nilai tinggi dari dokumen. Pada tahap ini dilakukan perhitungan nilai tinggi (α) untuk menentukan nilai beda tinggi (∂) yang akan digunakan untuk menentukan *cluster* yang optimal. Nilai maksimal diperoleh ketika nilai beda tinggi maksimum, kemudian nilai beda tinggi di simpan ke dalam *array*.

4.3.2 Implementasi Algoritma Proses Pengecekan Kemiripan Dokumen

Proses pengecekan kemiripan dokumen atau yang biasa disebut plagiarisme ini dilakukan dengan *input* dokumen baru. Setelah itu, sistem melakukan proses *text mining*, pembobotan TF-IDF, dan melakukan *clustering* CLHM. Setelah didapat *cluster* yang sama, dokumen uji akan dibandingkan dengan dokumen latih dalam satu *cluster*. Kemudian menghitung prosentase kemiripan dokumen dengan dokumen latih.

```
1  function hitungAtas($a, $b){
2    $count = count($a);
3    $sum = 0;
4    for($i=0;$i<$count; $i++){
5        $sum += ($a[$i]*$b[$i]);
6    }
7    return $sum;
8  }
9  function hitungBawah($c){
10   $count = count($c);
11   $sum = 0;
12   for($i=0;$i<$count; $i++){
13       $sum += pow($c[$i],2);
```

```
14    }
15    return sqrt($sum);
16  }
17  if(!empty ($id_dokumen)){
18    $sql = mysql_query("select * from daftar_dokumen where
19  id_dokumen = '$id_dokumen'");
20    $data = mysql_fetch_array($sql);
21    $i_dokumen = $data['id_dokumen'];
22    $n_dokumen = $data['nama_dokumen'];
23    $c_dokumen = $data['cluster'];
24    $dataku = array();
25  $sql5 = mysql_query("select nama_dokumen from daftar_dokumen
26  where cluster = '$c_dokumen' and nama_dokumen !=
27  '$n_dokumen'");
28  while ($data5 = mysql_fetch_array($sql5)){
29    $identitas5 = $data5['nama_dokumen'];
30    $nilai5 = array();
31    $sql6 = mysql_query("select kata from dokumen");
32    while ($data6 = mysql_fetch_array($sql6)){
33        $kata6 = $data6['kata'];
34        $sql7 = mysql_query("select $identitas5 doc1 from
35  dokumen where kata = '$kata6'");
36        $data7 = mysql_fetch_array($sql7);
37        $bobot7 = $data7['doc1'];
38        $nilai5[] = $bobot7;
39    }
40    $gab2 = array($nilai5,array("$identitas5"));
41    $dataku[] = $gab2;
42    unset($gab2);
43    unset($nilai5);
44  }
45  $n_dataku = count ($dataku);
46  $n_dataku2 = count ($dataku[0][0]);
47    //perhitungan nilai df
48  $df = array();
49  for ($i=0; $i < $n_dataku2 ; $i++) {
50    $sum = 0;
51    for ($j=0; $j < $n_dataku ; $j++) {
52        if($dataku[$j][0][$i] != 0)
53          $sum += 1;
54        }
55        $df[$i] = $sum;
56  }
57  //hitung normalisasi
58  $normalisasi = array();
59  $kump_nilai = array();
60  for ($k=0; $k < $n_dataku ; $k++) {
61    for ($l=0; $l < $n_dataku2 ; $l++) {
62        $nilai = $dataku[$k][0][$l];
63        $nilai_df = $df[$l];
```

```
64          $bobot_dok = hitungBobot($nilai, $n_dataku, $nilai_df);
65          $kump_nilai[0][$l] = $bobot_dok;
66      }
67      $normalisasi[$k] = $kump_nilai;
68  }
69  $total_bawah1 = hitungBawah($normalisasi_uji[0][0]);
70  $rincian = array();
71  for ($m=0; $m < count($dataku) ; $m++) {
72      $total_atas = hitungAtas($normalisasi_uji[0][0],
73  $normalisasi[$m][0]);
74      $total_bawah2 = hitungBawah($normalisasi[$m][0]);
75      $cosine = $total_atas / ($total_bawah1 * $total_bawah2);
76      $prosentase = round(($cosine*100),3);
77   }
78  }
```

Sourcecode 4. 11 Implementasi algoritma proses pengecekan kemiripan dokumen
Sumber: Implementasi

Fungsi pendukung pengecekan similaritas adalah Fungsi *hitungAtas* dan *hitungBawah*. Fungsi *hitungAtas* adalah proses perhitungan *dot product* dari dua dokumen yang dibandingkan. Fungsi *hitungBawah* adalah perhitungan bobot dari satu dokumen untuk dikalikan dengan dokumen lainnya. Kedua fungsi ini sebagai pendukung untuk perhitungan *cosine similarity*.

Proses pengambilan informasi dan bobot kata dokumen uji dari *database*. Selanjutnya, proses mengambil bobot kata pada dokumen latih yang memiliki *cluster* yang sama dengan dokumen uji. Setelah informasi dokumen dan bobot dokumen dalam satu *cluster* diperoleh, kemudian menghitung kemiripan dokumen menggunakan *cosine similarity*. Hasil perhitungan *cosine similarity* akan dijadikan prosentase untuk menentukan kemiripan dokumen. Dokumen latih yang mendekati nilai prosentase 100 persen bisa dikatakan memiliki kemiripan yang hampir sama.

4.4 Implementasi Antar Muka

Antarmuka sistem penentuan tingkat plagiarisme dokumen digunakan oleh pengguna untuk berinteraksi dengan sistem. Antarmuka sistem penentuan plagiarisme ini dibagi menjadi antarmuka halaman cek kemiripan dokumen, halaman tambah dokumen, dan halaman daftar dokumen.

4.4.1 Tampilan Halaman Cek Kemiripan Dokumen

Tampilan halaman cek kemiripan dokumen akan menampilkan dokumen dari *database* dan *user* memilih satu dokumen yang akan dibandingkan kemiripan dengan dokumen dalam satu *cluster*. Gambar *form* cek kemiripan dokumen ditunjukkan pada gambar 4.1.

Gambar 4. 1 *Form* cek kemiripan dokumen
Sumber: Implementasi

4.4.2 Tampilan Halaman Tambah Dokumen

Tampilan halaman tambah dokumen akan menampilkan *form* pilihan input dokumen yang akan dikluster. Gambar *form* tambah dokumen ditunjukkan pada gambar 4.2.

Gambar 4. 2 *Form* tambah dokumen
Sumber: Implementasi

4.4.3 Tampilan Halaman Daftar Dokumen

Tampilan halaman daftar dokumen akan menampilkan tabel daftar dokumen beserta *cluster* yang tersimpan pada *database*. Gambar *form* daftar dokumen ditunjukkan pada gambar 4.3.

Gambar 4. 3 *Form* lihat daftar dokumen
Sumber: Implementasi

BAB V
PENGUJIAN DAN ANALISIS

Bab ini akan membahas mengenai pengujian dari aplikasi perangkat lunak yang telah dibangun. Pengujian sistem dilakukan untuk menguji kelayakan sistem penentuan tingkat plagiarisme dokumen penelitian yang telah dibuat dan menunjukkan bahwa sistem yang telah dibuat bekerja dengan baik sesuai spesifikasi sistem yang telah ditentukan.

5.1 Pengujian Ketepatan *Cluster*

Pengujian ketepatan pembentukan *cluster* secara otomatis menggunakan *hill climbing*. Jumlah *cluster* secara otomatis terbentuk ketika nilai beda tinggi (∂) adalah nilai optimal. Pengujian proses pembentukan *cluster* yang optimal menggunakan nilai tinggi $\alpha = 2$, $\alpha = 3$, dan $\alpha = 4$ [ELD-10].

Tabel 5. 1 Hasil pengujian proses pembentukan *cluster*

No.	Jumlah Dokumen	Jumlah Cluster dan (Max (δ))		
		α = 2	α = 3	α = 4
1.	5	3 (0,0389)	3 (0,05843)	3 (0,07790)
2.	10	2 (44,51462)	2 (67,0604)	2(89,60617)
3.	15	3 (181,3512)	3 (272,21085)	3 (363,07051)
4.	20	4 (11,45477)	4 (17,36781)	4 (23,28086)
5.	25	5 (2,20019)	5 (5,87169)	5 (9,54318)
6.	30	6 (13,36736)	6 (20,36448)	6 (27,36161)
7.	35	7 (13,72968)	7 (21,25473)	7 (28,77978)
8.	40	8 (9,99709)	8 (15,69833)	8 (21,39956)
9.	45	9 (6,81113)	9 (10,93481)	9 (15,05848)
10.	50	10 (5,95336)	10 (9,76561)	10 (13,57786)
11.	55	11 (4,77526)	11 (8,00301)	11 (11,23077)
12.	60	12 (5,04404)	12 (8,50183)	12 (11,95962)
13.	65	13 (5,85832)	13 (9,80162)	13 (13,74491)
14.	70	14 (4,09325)	14 (7,08818)	14 (10,08311)
15.	75	15 (4,18255)	15 (7,30955)	15 (10,43655)
16.	80	16 (3,44443)	16 (6,13863)	16 (8,83282)
17.	85	17 (3,37281)	17 (6,04054)	17 (8,70828)
18.	90	18 (3,73330)	18 (6,68462)	18 (9,63595)

19.	95	19 (4,10329)	19 (7,35060)	19 (10,59792)
20.	100	20 (4,51905)	20 (8,08611)	20 (11,65316)
21.	105	21 (4,43444)	21 (8,00002)	21 (11,56561)
22.	110	22 (5,00671)	22 (9,02292)	22 (13,03914)
23.	115	23 (4,03588)	23 (7,58710)	23 (11,13831)
24.	120	24 (4,30691)	24 (8,09938)	24 (11,89184)
25.	125	25 (4,59797)	25 (8,65294)	25 (12,70792)

Sumber : Pengujian

Dari hasil pengujian proses pembentukan *cluster*, nilai optimal pembentukan *cluster* didominasi pada saat nilai tinggi (α) = 4. Metode *hill climbing* dengan nilai tinggi 2, 3, dan 4 relatif akan menghasilkan jumlah *cluster* yang sama, tetapi dengan nilai beda tinggi (∂) maksimal yang berbeda. Jumlah *cluster* yang digunakan adalah jumlah *cluster* yang optimal yaitu *cluster* yang memiliki nilai beda tinggi yang terbesar.

5.2 Pengujian *F-measure*

Akurasi sistem penentuan tingkat plagiarisme dihitung menggunakan *f-measure*. Pengujian tingkat akurasi sistem menggunakan *presisi*, *recall*, dan *f-measure* berdasarkan jumlah *cluster* optimal dengan nilai beda tinggi maksimal pada pengujian ketepatan *cluster*. *F-measure* menghitung ketepatan *output* pada sistem dengan ketepatan *output* yang diharapkan. Tabel 5.2 menunjukkan hasil pengujian *f-measure*.

Tabel 5. 2 Hasil pengujian *f-measure*

No.	Jumlah Dokumen	Nilai α	Max (∂)	Presisi	Recall	F-Measure
1.	5	4	0,07790	0,6	0,6	0,6
2.	10	4	89,60617	1	1	1
3.	15	4	363,07051	1	1	1
4.	20	4	23,28086	1	1	1
5.	25	4	9,54318	1	1	1
6.	30	4	27,36161	1	1	1
7.	35	4	28,77978	1	1	1
8.	40	4	21,39956	1	1	1
9.	45	4	15,05848	1	1	1
10.	50	4	13,57786	1	1	1

11.	55	4	11,23077	1	1	1
12.	60	4	11,95962	1	1	1
13.	65	4	13,74491	1	1	1
14.	70	4	10,08311	1	1	1
15.	75	4	10,43655	1	1	1
16.	80	4	8,83282	1	1	1
17.	85	4	8,70828	1	1	1
18.	90	4	9,63595	1	1	1
19.	95	4	10,59792	1	1	1
20.	100	4	11,65316	1	1	1
21.	105	4	11,56561	1	1	1
22.	110	4	13,03914	1	1	1
23.	115	4	11,13831	1	1	1
24.	120	4	11,89184	1	1	1
25.	125	4	12,70792	1	1	1
Rata - rata F-Measure						**0,984**

Sumber : Pengujian

Hasil akurasi bernilai 1 artinya tingkat akurasi sistem menggunakan perhitungan *f-measure* memiliki *cluster* yang baik. Hasil pengujian pada tabel 5.2 dapat diambil kesimpulan bahwa hasil pengklusteran relatif baik dan stabil ketika jumlah dokumen lebih dari 10 dokumen. Sedangkan, proses pembentukan *cluster* pada dokumen yang berjumlah lima menghasilkan *cluster* yang memiliki nilai *f-measure* rendah. Hasil analisa pengujian *f-measure* pada proses pengklusteran menggunakan *Centroid Linkage Hierarchical Method* (CLHM) dan pembentukan *cluster* secara otomatis menggunakan *hill climbing* memiliki rata – rata *f-measure* sebesar 0,984. Semakin besar nilai *f-measure*, yaitu nilai *f-measure* yang mendekati nilai satu maka semakin baik hasil klusternya. Hal ini dapat disimpulkan bahwa pembentukan *cluster* pada CLHM menggunakan *hill climbing* memiliki akurasi yang sangat baik.

5.3 Pengujian *Similarity* Dokumen

Pengujian hasil prosentase plagiarisme dokumen adalah penentuan kemiripan dokumen uji dengan dokumen pada *database* yang diukur dengan angka. Hasil pengujian pada tabel 5.3 menunjukkan rata – rata prosentase kemiripan dokumen dengan pengujian random sebanyak lima kali percobaan.

Tabel 5. 3 Hasil pengujian prosentase plagiarisme

No.	Dokumen Uji	Dokumen Latih	Prosentase kemiripan (Pemotongan kata sebelum *stemming*) (%)	Prosentase kemiripan (Pemotongan kata setelah *stemming*) (%)
1.	Dok1	100% sama	100	100
2.		75% sama	74,652	73,188
3.		50% sama	51,207	58,363
4.		25% sama	24,131	36,456
5.	Dok2	100% sama	100	100
6.		75% sama	77,774	75,044
7.		50% sama	42,818	53,476
8.		25% sama	20,145	30,486
9.	Dok3	100% sama	100	100
10.		75% sama	76,62	3,969
11.		50% sama	54,043	56,458
12.		25% sama	32,57	32,385
13.	Dok4	100% sama	100	100
14.		75% sama	69,231	77,638
15.		50% sama	49,861	54,606
16.		25% sama	19,764	34,266
17.	Dok5	100% sama	100	100
18.		75% sama	72,909	74,069
19.		50% sama	41,393	52,005
20.		25% sama	19,288	27,215
21.	Dok6	100% sama	100	100
22.		75% sama	70,118	73,854
23.		50% sama	45,444	54,706
24.		25% sama	25,943	38,005
25.	Dok7	100% sama	100	100
26.		75% sama	74,576	71,662
27.		50% sama	54,535	48,773
28.		25% sama	32,137	30,995
29.	Dok8	100% sama	100	100
30.		75% sama	76,035	82,487
31.		50% sama	53,934	53,64
32.		25% sama	30,686	34,775
33.	Dok9	100% sama	100	100
34.		75% sama	76,814	74,634
35.		50% sama	55,542	55,157
36.		25% sama	31,412	34,827
37.	Dok10	100% sama	100	100
38.		75% sama	73,548	72,686

39.		50% sama	59,649	47,626
40.		25% sama	32,673	32,452
41.	Dok11	100% sama	100	100
42.		75% sama	74,146	76,971
43.		50% sama	55,244	56,239
44.		25% sama	28,96	36,57
45.	Dok12	100% sama	100	100
46.		75% sama	69,714	70,484
47.		50% sama	42,957	58,133
48.		25% sama	18,409	34,665
49.	Dok13	100% sama	100	100
50.		75% sama	78,026	73,482
51.		50% sama	57,834	55,64
52.		25% sama	29,478	32,234
53.	Dok14	100% sama	100	100
54.		75% sama	68,279	75,26
55.		50% sama	51,213	57,753
56.		25% sama	20,54	34,946
57.	Dok15	100% sama	100	100
58.		75% sama	75,169	75,204
59.		50% sama	41,711	50,161
60.		25% sama	25,272	36,039
61.	Dok16	100% sama	100	100
62.		75% sama	73,405	81,968
63.		50% sama	41,108	53,683
64.		25% sama	25,873	34,093
65.	Dok17	100% sama	100	100
66.		75% sama	69,339	76,39
67.		50% sama	49,827	57,384
68.		25% sama	26,827	38,199
69.	Dok18	100% sama	100	100
70.		75% sama	76,496	74,188
71.		50% sama	42,478	50,156
72.		25% sama	25,527	31,244
73.	Dok19	100% sama	100	100
74.		75% sama	69,024	80,666
75.		50% sama	44,13	52,443
76.		25% sama	24,265	26,619
77.	Dok20	100% sama	100	100
78.		75% sama	80,574	78,415
79.		50% sama	44,53	60,629
80.		25% sama	29,586	39,287

81.	Dok21	100% sama	100	100
82.		75% sama	75,622	80,063
83.		50% sama	51,176	56,46
84.		25% sama	28,344	40,752
85.	Dok22	100% sama	100	100
86.		75% sama	70,483	72,059
87.		50% sama	42,485	53,228
88.		25% sama	23,041	37,221
89.	Dok23	100% sama	100	100
90.		75% sama	78,293	65,298
91.		50% sama	50,341	52,966
92.		25% sama	22,178	29,027
93.	Dok24	100% sama	100	100
94.		75% sama	72,441	80,139
95.		50% sama	48,533	54,448
96.		25% sama	28,577	35,02
97.	Dok25	100% sama	100	100
98.		75% sama	80,61	79,03
99.		50% sama	54,485	61,013
100.		25% sama	28,304	37,776

Sumber : Pengujian

Pengujian prosentase tidak sama persis dengan prosentase pemotongan kata, misalnya kata yang dipotong 25% hasil prosentase tidak sama persis 75% dengan dokumen uji. Prosentase *similarity* pada pengujian pemotongan kata setelah *stemming* tidak sama persis seperti pemotongan kata karena bergantung pada *frekuensi* kemunculan *term* pada perhitungan kemiripan menggunakan *cosine similarity*. Hal ini terjadi juga pada pengujian pemotongan kata sebelum *stemming.* Selain itu, faktor yang mempengaruhi prosentase pada pengujian pemotongan kata sebelum *stemming* adalah dokumen mengalami proses *text mining*, seperti penghilangan *stopword* dan proses pemotongan kata pada dokumen latih secara *random*. *Random* pada kasus ini adalah pemotongan kata tidak berdasar pada kata penting saja, melainkan kemungkinan kata yang dihilangkan adalah kata yang termasuk *stopword*. Sehingga, prosentase kemiripan bisa lebih dari 75% atau kurang dari 75%. Hal ini terjadi juga pada proses pemotongan kata sebesar 50% dan 75%.

5.4 Pengujian Prosentase *Error*

Prosentase *error* yaitu pengujian terhadap sistem dengan menghitung nilai *similarity* (kemiripan) yang dihasilkan sistem dibandingkan nilai *similarity* yang diharapkan sehingga dapat dilihat apakah sistem yang dibuat telah sesuai dengan hasil yang diharapkan. Tabel 5.4 sampai dengan tabel 5.7.

Tabel 5. 4 Hasil pengujian *prosentase error* tanpa pemotongan kata

No	Dokumen Uji	Pemotongan kata sebelum di-*stemming*		Pemotongan kata setelah di-*stemming*	
		Similarity (%)	Prosentase *error* (%)	*Similarity* (%)	Prosentase *error* (%)
1	dok1	100	0	100	0
2	dok2	100	0	100	0
3	dok3	100	0	100	0
4	dok4	100	0	100	0
5	dok5	100	0	100	0
6	dok6	100	0	100	0
7	dok7	100	0	100	0
8	dok8	100	0	100	0
9	dok9	100	0	100	0
10	dok10	100	0	100	0
11	dok11	100	0	100	0
12	dok12	100	0	100	0
13	dok13	100	0	100	0
14	dok14	100	0	100	0
15	dok15	100	0	100	0
16	dok16	100	0	100	0
17	dok17	100	0	100	0
18	dok18	100	0	100	0
19	dok19	100	0	100	0
20	dok20	100	0	100	0
21	dok21	100	0	100	0
22	dok22	100	0	100	0
23	dok23	100	0	100	0
24	dok24	100	0	100	0
25	dok25	100	0	100	0
Rata-Rata		**100**	**0**	**100**	**0**

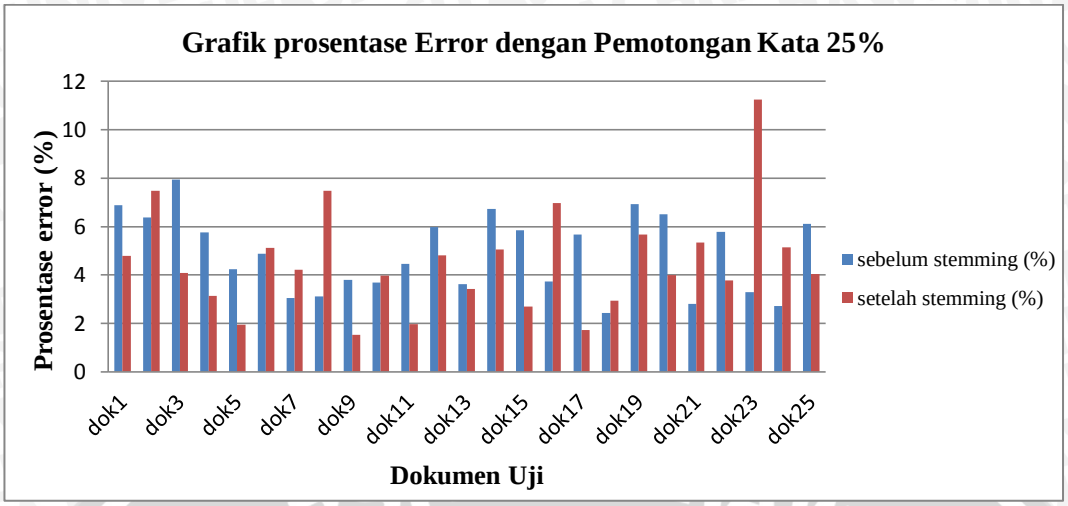
Sumber : Pengujian

Hasil pengujian prosentase *error* tanpa pemotongan kata, yaitu pengujian dokumen uji dan dokumen latih yang memiliki 100% kata yang sama memiliki prosentase *error* sebesar 0%.

Tabel 5. 5 Hasil pengujian prosentase *error* dengan pemotongan kata 25%

No	Dokumen Uji	Pemotongan kata sebelum di *stemming*		Pemotongan kata setelah di *stemming*	
		***Similarity* (%)**	**Prosentase *error* (%)**	***Similarity* (%)**	**Prosentase *error* (%)**
1	dok1	74,652	6,873	73,188	4,780
2	dok2	77,774	6,367	75,044	7,486
3	dok3	76,620	7,944	73,969	4,077
4	dok4	69,231	5,769	77,638	3,125
5	dok5	72,909	4,229	74,069	1,945
6	dok6	70,118	4,882	73,854	5,121
7	dok7	74,576	3,054	71,662	4,226
8	dok8	76,035	3,115	82,487	7,487
9	dok9	76,814	3,792	74,634	1,533
10	dok10	73,548	3,691	72,686	3,983
11	dok11	74,146	4,464	76,971	1,971
12	dok12	69,714	5,970	70,484	4,804
13	dok13	78,026	3,617	73,482	3,424
14	dok14	68,279	6,721	75,260	5,045
15	dok15	75,169	5,848	75,204	2,692
16	dok16	73,405	3,742	81,968	6,968
17	dok17	69,339	5,661	76,390	1,722
18	dok18	76,496	2,441	74,188	2,927
19	dok19	69,024	6,922	80,666	5,666
20	dok20	80,574	6,506	78,415	4,002
21	dok21	75,622	2,803	80,063	5,347
22	dok22	70,483	5,789	72,059	3,785
23	dok23	78,293	3,293	65,298	11,245
24	dok24	72,441	2,722	80,139	5,139
25	dok25	80,610	6,119	79,030	4,030
Rata-Rata		**74,156**	**4,893**	**75,554**	**4,501**

Sumber: Pengujian



Gambar 5. 1 Grafik prosentase *error* dengan pemotongan kata 25%
Sumber: Pengujian

Hasil pengujian prosentase *error* dengan pemotongan kata menghasilkan prosentase *error* lebih besar dibanding dengan dokumen dengan data latih 100%. Dokumen uji dan dokumen latih dengan pemotongan kata 25% memiliki prosentase *error* maksimum 11,689% dan prosentase *error* minimum 0,529% pada pemotongan kata sebelum *stemming*, dan prosentase *error* maksimum 6,765% dan prosentase *error* minimum 0,08% pada pemotongan kata setelah *stemming*. Gambar 5.1 menunjukkan grafik prosentase *error* dokumen dengan dengan pemotongan kata 25%.

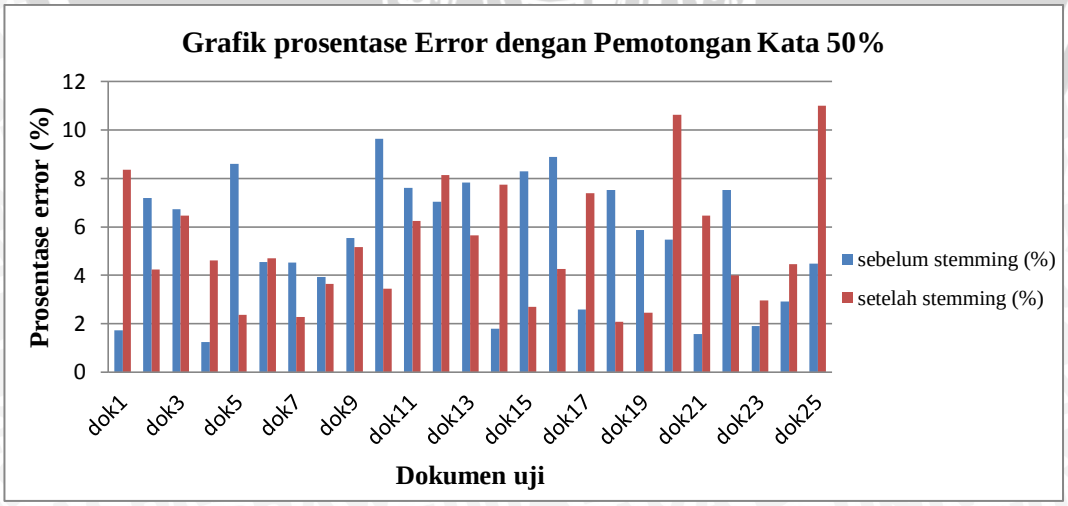
Tabel 5. 6 Hasil pengujian prosentase *error* dengan pemotongan kata 50%

No	Dokumen Uji	Pemotongan kata sebelum di-*stemming*		Pemotongan kata setelah di-*stemming*	
		***Similarity* (%)**	**Prosentase *error* (%)**	***Similarity* (%)**	**Prosentase *error* (%)**
1	dok1	51,207	1,723	58,363	8,363
2	dok2	42,818	7,182	53,476	4,240
3	dok3	54,043	6,720	56,458	6,458
4	dok4	49,861	1,242	54,606	4,606
5	dok5	41,393	8,607	52,005	2,373
6	dok6	45,444	4,556	54,706	4,706
7	dok7	54,535	4,535	48,773	2,268
8	dok8	53,934	3,934	53,640	3,640
9	dok9	55,542	5,542	55,157	5,157
10	dok10	59,649	9,649	47,626	3,437
11	dok11	55,244	7,607	56,239	6,239

12	dok12	42,957	7,043	58,133	8,133
13	dok13	57,834	7,834	55,640	5,640
14	dok14	51,213	1,800	57,753	7,753
15	dok15	41,711	8,289	50,161	2,701
16	dok16	41,108	8,892	53,683	4,265
17	dok17	49,827	2,586	57,384	7,384
18	dok18	42,478	7,522	50,156	2,072
19	dok19	44,130	5,870	52,443	2,443
20	dok20	44,530	5,470	60,629	10,629
21	dok21	51,176	1,568	56,460	6,460
22	dok22	42,485	7,515	53,228	3,999
23	dok23	50,341	1,905	52,966	2,966
24	dok24	48,533	2,907	54,448	4,448
25	dok25	54,485	4,485	61,013	11,013
Rata-Rata		**49,059**	**5,399**	**54,606**	**5,256**

Sumber: Pengujian

Hasil pengujian prosentase *error* dengan pemotongan kata menghasilkan prosentase *error* lebih besar dibanding dengan dokumen dengan data latih 100%. Dokumen uji dan dokumen latih dengan pemotongan kata 50% memiliki prosentase *error* maksimum 9,649% dan prosentase *error* minimum 1,242% pada pemotongan kata sebelum *stemming*, dan prosentase *error* maksimum 11,013% dan prosentase *error* minimum 2,072% pada pemotongan kata setelah *stemming*. Gambar 5.2 menunjukkan grafik prosentase *error* dokumen dengan dengan pemotongan kata 50%.

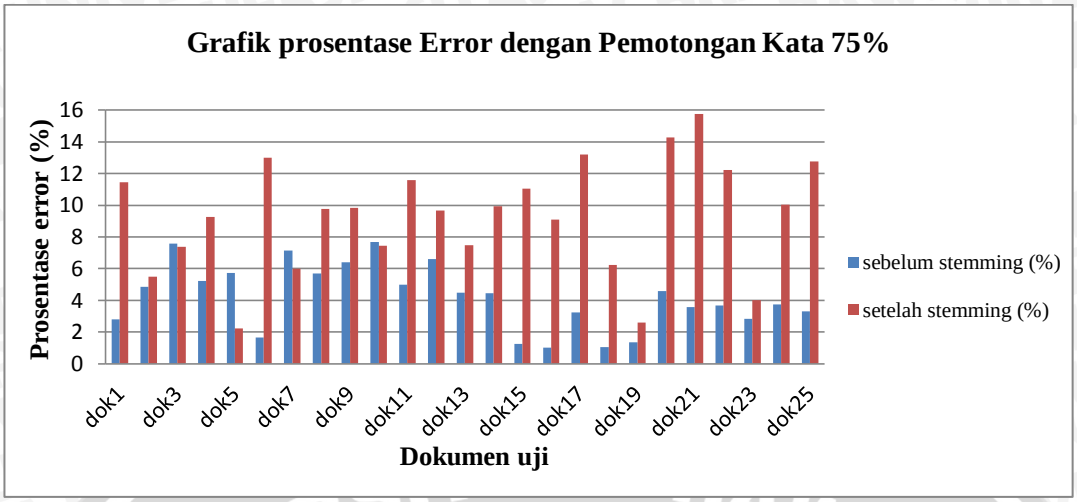


Gambar 5. 2 Grafik prosentase *error* dengan pemotongan kata 50%
Sumber: Pengujian

Tabel 5. 7 Hasil pengujian prosentase *error* dengan pemotongan kata 75%

No	Dokumen Uji	Pemotongan kata sebelum di *stemming*		Pemotongan kata setelah di *stemming*	
		***Similarity* (%)**	**Prosentase *error* (%)**	***Similarity* (%)**	**Prosentase *error* (%)**
1	dok1	24,131	2,800	36,456	11,456
2	dok2	20,145	4,855	30,486	5,486
3	dok3	32,570	7,570	32,385	7,385
4	dok4	19,764	5,236	34,266	9,266
5	dok5	19,288	5,712	27,215	2,215
6	dok6	25,943	1,670	38,005	13,005
7	dok7	32,137	7,137	30,995	5,995
8	dok8	30,686	5,686	34,775	9,775
9	dok9	31,412	6,412	34,827	9,827
10	dok10	32,673	7,673	32,452	7,452
11	dok11	28,960	4,992	36,570	11,570
12	dok12	18,409	6,591	34,665	9,665
13	dok13	29,478	4,478	32,234	7,483
14	dok14	20,540	4,460	34,946	9,946
15	dok15	25,272	1,263	36,039	11,039
16	dok16	25,873	1,013	34,093	9,093
17	dok17	26,827	3,233	38,199	13,199
18	dok18	25,527	1,063	31,244	6,244
19	dok19	24,265	1,352	26,619	2,584
20	dok20	29,586	4,586	39,287	14,287
21	dok21	28,344	3,581	40,752	15,752
22	dok22	23,041	3,665	37,221	12,221
23	dok23	22,178	2,822	29,027	4,027
24	dok24	28,577	3,757	35,020	10,020
25	dok25	28,304	3,304	37,776	12,776
Rata-Rata		**26,157**	**4,196**	**34,222**	**9,271**

Sumber: Pengujian



Gambar 5. 3 Grafik prosentase *error* dengan pemotongan kata 75%
Sumber: Pengujian

Hasil pengujian prosentase *error* dengan pemotongan kata menghasilkan prosentase *error* lebih besar dibanding dengan dokumen dengan data latih 100%. Dokumen uji dan dokumen latih dengan pemotongan kata 75% memiliki prosentase *error* maksimum 7,673% dan prosentase *error* minimum 1,013% pada pemotongan kata sebelum *stemming*, dan prosentase *error* maksimum 15,752% dan prosentase *error* minimum 2,584% pada pemotongan kata setelah *stemming*. Gambar 5.3 menunjukkan grafik prosentase *error* dokumen dengan dengan pemotongan kata 50%.

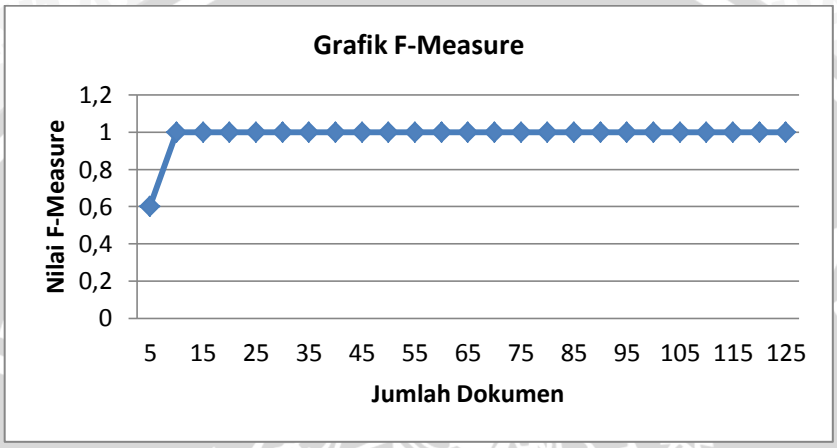
5.5 Analisa Hasil

Hasil analisa penelitian ini terdiri dari dua bahasan, yaitu ketepatan pembentukan *cluster* dan hasil *similarity* dokumen yang dibandingkan.

a. Ketepatan pembentukan *cluster*

Penentuan kemiripan topik dokumen penelitian ini menggunakan *Centroid Linkage Hierarchical Method* (CLHM) berguna untuk mengecek kemiripan dokumen terhadap sekumpulan dokumen yang terkelompok. *Clustering* dokumen pada sistem ini bertujuan untuk mengelompokkan dokumen berdasarkan kemiripan kata yang dimiliki. Selain itu, adanya *clustering* untuk mempercepat waktu eksekusi pengecekan dokumen dengan sekelompok dokumen dalam satu *cluster*. Sehingga, dokumen yang akan diuji tidak dibandingkan dengan seluruh data yang tersimpan dalam *database*.

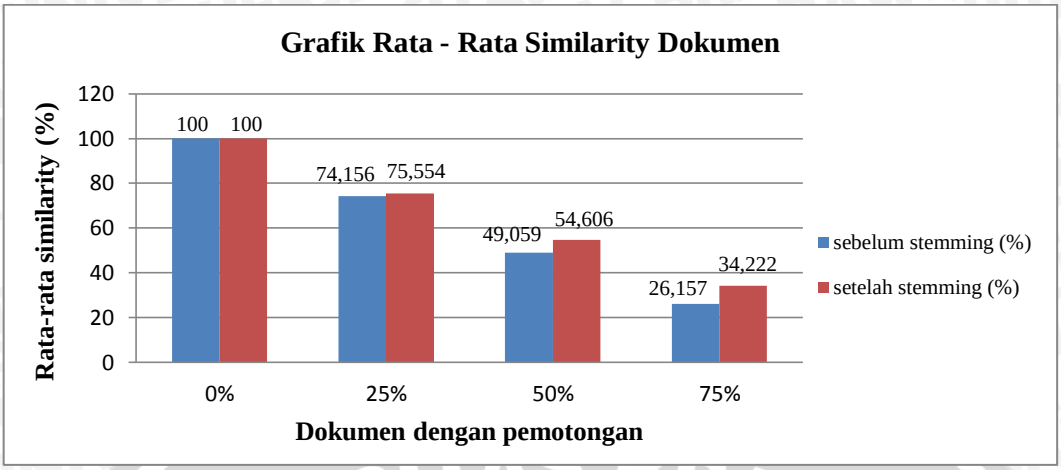
Hasil pengklusteran relatif baik dan stabil ketika jumlah dokumen lebih dari 10 dokumen. Sedangkan, proses pembentukan *cluster* pada dokumen yang berjumlah lima menghasilkan *cluster* yang memiliki nilai *f-measure* 0,6. Hal ini ditunjukkan pada gambar 5.4. Jumlah dokumen sedikit menghasilkan nilai *f-measure* rendah, sedangkan semakin banyak dokumen yang dikelompokkan proses *cluster* stabil.



Gambar 5. 4 Grafik nilai *f-measure* sistem
Sumber: Pengujian

b. Hasil *similarity* dan prosentase *error* pada dokumen

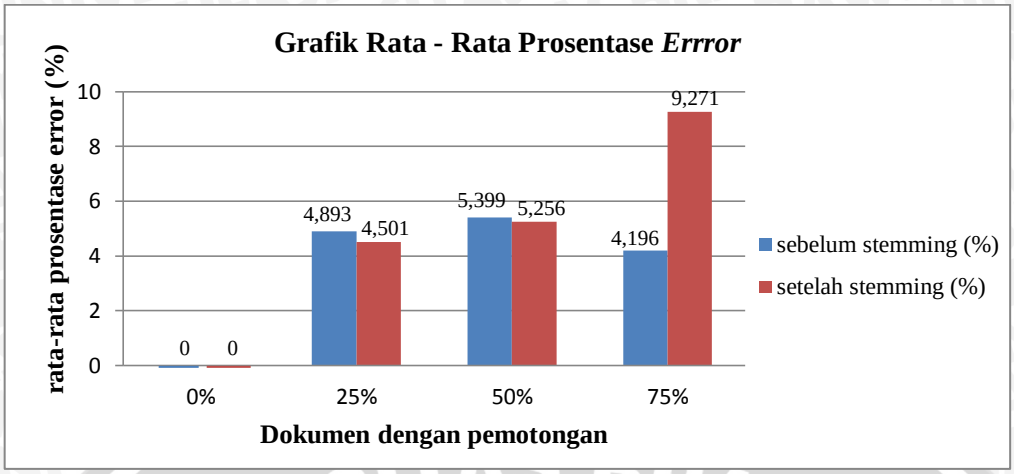
Setelah dokumen dikelompokkan berdasarkan kemiripan kata. Tahap selanjutnya adalah pengecekan kemiripan dokumen penelitian. Pengecekan dokumen pada sistem ini menggunakan *cosine similarity*. Hasil rata-rata *similarity* dari kemiripan dokumen ditunjukkan melalui grafik pada gambar 5.5. Semakin banyak kata yang sama, maka semakin tinggi prosentase kemiripannya. Semakin sedikit kata yang sama, dokumen memiliki prosentase yang semakin kecil.



Gambar 5. 5 Grafik rata-rata *similarity*
Sumber: Pengujian

Untuk menghitung nilai prosentase *error* digunakan hasil *output* sistem terhadap prosentase pemotongan. Hasil perhitungan prosentase *error* menghasilkan prosentase yang tidak seharusnya ada pada dokumen. Dokumen tanpa pemotongan memiliki prosentase *error* minimal yaitu 0%. Sedangkan untuk dokumen yang melalui pemotongan kata 25%, 50%, dan 75% memiliki prosentase *error* maksimal 9,649% untuk pengujian pemotongan kata sebelum *stemming* dan *error* maksimal 15,752% untuk pengujian pemotongan kata setelah *stemming*.

Rata-rata prosentase *error* pada semua dokumen ditunjukkan pada gambar 5.6. Hasil perhitungan rata-rata dokumen yang memiliki prosentase *error* terendah adalah dokumen tanpa pemotongan kata yaitu sebesar 0%. Sedangkan rata-rata prosentase *error* maksimal adalah dokumen dengan pemotongan sebesar 50% untuk pengujian pemotongan kata sebelum *stemming* dan dokumen dengan pemotongan sebesar 75% untuk pengujian pemotongan kata setelah *stemming*.



Gambar 5. 6 Grafik rata-rata prosentase *error*
Sumber: Perancangan

Prosentase *error* rata-rata pada dokumen tanpa pemotongan, dokumen yang dilakukan pemotongan sebesar 25%, 50%, dan 75% memiliki prosentase berturut-turut sebesar 4,893%, 5,399%, dan 4,196% untuk dokumen dengan pemotongan kata sebelum *stemming*. Sedangkan untuk dokumen dengan pemotongan kata sebelum *stemming* memiliki prosentase *error* rata-rata sebesar 4,501%, 5,256%, dan 9,271%. Prosentase *error* terjadi karena perhitungan kemiripan menggunakan *cosine similarity* yang perhitungan bobotnya berdasarkan *frekuensi* kemunculan *term* dan proses pemotongan kata pada dokumen latih secara *random*. Selain itu faktor yang mempengaruhi prosentase pada pengujian pemotongan kata sebelum *stemming* adalah dokumen mengalami proses *prepocessing*, seperti penghilangan *stopword* dan proses pemotongan kata pada dokumen latih secara *random*. Pengaruh hasil perubahan *similarity* pada dokumen dengan pemotongan kata secara random yang dilakukan sebanyak lima kali ditunjukkan pada tabel 5.8.

Tabel 5. 8 Pengaruh pemotongan kata sebesar 25% secara random

No.	Dokumen	Similarity (%)					Rata-Rata
		rand1	rand2	rand3	rand4	rand5	
1	Dok1	64,738	67,450	74,759	78,315	87,997	74,652
2	Dok2	68,763	72,255	82,289	85,439	80,126	77,774
3	Dok3	63,355	70,835	83,464	85,353	80,095	76,620
4	Dok4	69,228	64,196	70,147	73,449	69,134	69,231
5	Dok5	62,097	78,277	72,104	75,735	76,334	72,909
6	Dok6	66,583	70,603	71,948	67,719	73,736	70,118

7	Dok7	72,222	80,120	72,963	71,119	76,455	74,576
8	Dok8	71,655	79,112	73,146	79,099	77,164	76,035
9	Dok9	81,393	77,193	73,174	71,881	80,431	76,814
10	Dok10	72,195	69,420	70,900	74,627	80,598	73,548
11	Dok11	62,189	84,003	75,022	74,587	74,931	74,146
12	Dok12	76,709	64,087	71,589	65,964	70,220	69,714
13	Dok13	81,114	75,473	83,431	76,589	73,521	78,026
14	Dok14	64,515	66,536	67,814	71,205	71,323	68,279
15	Dok15	68,183	71,517	71,103	84,653	80,391	75,169
16	Dok16	76,673	73,381	69,138	78,693	69,138	73,405
17	Dok17	61,858	73,167	69,523	73,722	68,427	69,339
18	Dok18	75,177	75,171	79,605	72,639	79,890	76,496
19	Dok19	73,718	77,364	60,609	65,836	67,592	69,024
20	Dok20	86,243	80,019	72,670	84,958	78,982	80,574
21	Dok21	82,087	74,526	71,211	76,476	73,810	75,622
22	Dok22	68,553	73,230	66,226	78,180	66,226	70,483
23	Dok23	76,541	77,529	83,704	77,727	75,965	78,293
24	Dok24	65,711	74,395	74,572	75,408	72,118	72,441
25	Dok25	81,986	85,509	78,353	73,728	83,475	80,610

Sumber: Pengujian

Hasil pengujian secara acak menunjukkan nilai rata-rata kemiripan yang berbeda pada setiap kali percobaan. Hal ini dikarenakan perbandingan kemiripan dokumen menggunakan *cosine similarity* yang memperhitungkan frekuensi kemunculan kata (*term*) pada masing-masing dokumen.

BAB VI
PENUTUP

Bab ini akan membahas mengenai kesimpulan dan saran yang dapat diambil dari pembuatan sistem penentuan plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM).

6.1 Kesimpulan

Kesimpulan dari hasil penelitian sistem penentuan palgiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM) sebagai berikut:

1. Sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan CLHM telah dibuat sesuai perancangan dan dapat digunakan untuk membandingkan kemiripan dokumen terhadap sekelompok dokumen. Proses perbandingan dokumen yang akan dicek kemiripannya hanya akan dilakukan pada dokumen dalam satu *cluster* yang memiliki similaritas terdekat dengan dokumen uji.
2. Hasil akurasi sistem penentuan tingkat plagiarisme dokumen penelitian adalah sebagai berikut:
   a. *Sistem* menggunakan CLHM dengan analisa varian *hill climbing* menghasilkan nilai *f-measure* sebesar 0,984.
   b. Prosentase *error* pada pengujian pemotongan kata sebelum *stemming* dan setelah *stemming* relatif sama. Pemotongan kata sebesar 25%, 50%, dan 75% menghasilkan prosentase *error* berturut – turut sebesar 4,893%, 5,399%, dan 4,196% pada dokumen dengan pemotongan kata sebelum *stemming*, dan pemotongan kata setelah *stemming* menghasilkan prosentase *error* 2,501%, 5,256%, dan 9,271%. Pemotongan kata secara *random* berpengaruh pada nilai *similarity* dan prosentase *error* yang dihasilkan.
   c. Variasi kata (*term*) pada dokumen berpengaruh terhadap baik atau tidaknya proses pengelompokan dokumen pada *cluster*.

6.2 Saran

Saran dari hasil penelitian sistem penentuan tingkat plagiarisme dokumen penelitian menggunakan *Centroid Linkage Hierarchical Method* (CLHM) untuk pengembangan sistem lebih lanjut, sebaiknya digunakan sistem untuk pengecekan sinonim kata, dan pengecekan kemiripan berdasarkan makna agar didapatkan hasil yang optimal.

DAFTAR PUSTAKA

[ARI-01] Arifin, Agus Z., Setiono, Ari N. 2001. *Klasifikasi Dokumen Berita Kejadian Berbahasa Indonesia dengan Algoritma Single Pass Clustering*. Institut Teknologi Sepuluh November (ITS), Surabaya

[BER-10] Berry, Michael W. dan Kogan, Jacob. 2010. *Text Mining: Applications and Theory*. USA: Wiley

[DAM-11] Damayanti, Nadia. 2011. *Temu Kembali Informasi Berdasarkan Lokasi pada Dokumen yang Dikelompokkan Menggunakan Metode Centroid Linkage Hierarchical*. Tugas Akhir Jurusan Teknologi Informasi Politeknik Elektronika Negeri Surabaya, Surabaya

[ELD-10] Eldira, Hervilorra. 2010. *Web Mining untuk Pencarian dokumen Bahasa Inggris menggunakan Hill Climbing Automatic Cluster*. Tugas Akhir Jurusan Teknologi Informasi Politeknik Elektronika Negeri Surabaya, Surabaya

[ERN-09] Ernawati, S., Ardiyanti, A., Setiawan, Erwin B. 2009. *Klusterisasi Dokumen Berbahasa Indonesia Menggunakan Document Index Graph*. Seminar Nasional Aplikasi Teknologi Informasi (SNATI), hal. F109– F113

[HAM-07] Hamzah, A., Susanto, A., Soesianto, F., Istyanto, Jazi E. 2007. *Perbandingan Feature Kata dan Frasa dalam Clustering Dokumen Text Berbahasa Indonesia*. Seminar Nasional Aplikasi Teknologi Informasi (SNATI), hal. B53-B58

[KAO-07] Kao, A., dan Poteet, Stephen R. 2007. *Natural Language Processing and Text Mining*. London : Springer

[KUR-08] Kurniawati, A., dan Simri, I Wayan. 2008. *Perbandingan Pendekatan Deteksi Plagiarism Dokumen dalam Bahasa Inggris.* Proceeding, Seminar Ilmiah Nasional Komputer dan Sistem Intelijen (KOMMIT 2008)

[KUR-09] Kurniawan, A., Toba, H. 2009. *Pembuatan Aplikasi Bergerak Temu Ulang File Elektronik Berbahasa Indonesia dengan Memanfaatkan Java CLDC*. Seminar Nasional Aplikasi

Teknologi Informasi (SNATI)

[MAR-10] Martiana, E., Rosyid, N., Agusetia, U. 2010. *Mesin Pencari Dokumen dengan Pengklasteran secara Otomatis*. TELKOMNIKA Vol. 8, No. 1, hal. 41-48

[NOO-09] Noor, M. H., Hariadi, M. 2009. *Image Cluster Berdasarkan Warna untuk Identifikasi Kematangan Buah Tomat dengan Metode Valley Tracing.* Seminar Nasional Aplikasi Teknologi Informasi (SNATI), hal. A15-A24

[RIZ-12] Rizqa, Arroyida. 2012. *Sistem Penilaian Otomatis Jawaban Essay Menggunakan Deteksi Similarity*. Tugas Akhir, Program Studi Teknik Informatika, Universitas Pembangunan Veteran, Jawa Timur

[SAL-12] Saleh, Muhammad. 2012. *Menentukan Kemiripan Topik Tugas Akhir Berdasarkan Deskripsi Pada Jurusan Teknik Informatika Menggunakan Metode Inner Product dan Single Linkage Hierarchical*. Skripsi Teknik Informatika, Fakultas Teknik, Universitas Muhammadiyah Malang

[SUS-12] Suswatingsih, Yeni. 2012. *Implementasi Cosine Similarity dan TF-IDF pada pencarian terjemahan ayat – ayat Al-Quran dan Hadits sahahih dengan mempertimbangkan evaluasi user*. Skripsi Teknik Informatika, Fakultas Teknik, Universitas Muhammadiyah Malang

[VID-10] Vidyastana, damitha. 2010. *Penentuan Kemiripan Topik Proyek Akhir Berdasarkan Abstrak Pada Jurusan Teknik Informatika Menggunakan Metode Single Linkage Hierarchical*. Skripsi Politeknik Elektronika Negeri Surabaya (PENS), Institut Teknologi Sepuluh Nopember, Surabaya

LAMPIRAN