DETEKSI PLAGIARISME MENGGUNAKAN KOMBINASI VECTOR SPACE MODEL DAN JACCARD COEFISIEN

SKRIPSI

Untuk memenuhi sebagian persyaratan mencapai gelar Sarjana Komputer

Disusun Oleh :

MUHAMAD EFENDI

NIM. 115060800111104

PROGRAM STUDI INFORMATIKA/ILMU KOMPUTER

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2015

LEMBAR PERSETUJUAN

DETEKSI PLAGIARISME MENGGUNAKAN KOMBINASI VECTOR SPACE MODEL DAN JACCARD COEFISIEN

SKRIPSI

LABORATORIUM KOMPUTASI CERDAS DAN VISUALISASI

Untuk memenuhi sebagian persyaratan mencapai gelar Sarjana Komputer

Disusun oleh:

MUHAMAD EFENDI

NIM. 115060800111104

Skripsi ini telah disetujui oleh dosen pembimbing pada tanggal 5 Mei 2015

Dosen Pembimbing I	**Dosen Pembimbing II**
Indriati, ST., M.Kom	**Lailil Muflikhah, S.Kom., M.Sc**
NIK. 831013 06 1 2 0035	**NIP. 19741113 200501 2 001**

LEMBAR PENGESAHAN

DETEKSI PLAGIARISME MENGGUNAKAN KOMBINASI VECTOR SPACE MODEL DAN JACCARD COEFISIEN

SKRIPSI

LABORATORIUM KOMPUTASI CERDAS DAN VISUALISASI

Untuk memenuhi sebagian persyaratan mencapai gelar Sarjana Komputer

Disusun Oleh :

MUHAMAD EFENDI
NIM. 115060800111104

Skripsi ini telah dipertahankan dan dinyatakan lulus di depan majelis penguji pada tanggal 22 Mei 2015

Penguji I	**Penguji II**
Candra Dewi, S.Kom, M.Sc	**M. Tanzil Furqon, S.Kom, M.CompSc**
NIP. 19771114 200312 2 001	**NIP. 19820930 200801 1 004**

Penguji III

Wijaya Kurniawan, S.T, M.T
NIK. 820125 16 1 1 0418

Mengetahui,
Ketua Program Studi Informatika / Ilmu Komputer

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, 22 Mei 2015

Mahasiswa

Muhamad Efendi

NIM.115060800111104

KATA PENGANTAR

Syukur dan Alhamdulillah dengan nama Allah SWT Yang Maha Pengasih dan Penyayang. Segala puji bagi Allah SWT karena atas rahmat dan hidayah-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul "Deteksi Plagiarisme Menggunakan Kombinasi Vector Space Model dan Jaccard Coefisien". Shalawat serta salam atas junjungan besar kita Nabi Muhammad S.A.W. beserta keluarga dan para sahabat sekalian.

Skripsi ini diajukan sebagai syarat ujian skripsi dalam rangka untuk memperoleh gelar Sarjana Komputer di Program Teknologi Informasi Dan Ilmu Komputer (PTIIK), Program Studi Informatika/Ilmu Komputer, Universitas Brawijaya Malang. Atas terselesaikannya skripsi ini, penulis mengucapkan terima kasih kepada:

1. Ibu Indriati, ST., M.Kom dan Ibu Lailil Muflikhah, S.Kom., M.Sc selaku dosen pembimbing tugas akhir penulis
2. Bapak Ir. Sutrisno, M.T, Bapak Ir. Heru Nurwasito, M.Kom, Bapak Himawat Aryadita, S.T, M.Sc, dan Bapak Eddy Santoso, S.Kom selaku Ketua, Wakil Ketua 1, Wakil Ketua 2 dan Wakil Ketua 3 Fakultas Ilmu Komputer Universitas Brawijaya
3. Bapak Drs. Marji, MT dan Bapak Issa Arwani, S.Kom., M.Sc selaku Ketua dan Sekretaris Program Studi Informatika.
4. Bapak / Ibu Dosen, Staff Administrasi dan Perpustakaan Fakultas Ilmu Komputer Universitas Brawijaya.
5. Kepada orang tua penulis dan seluruh keluarga yang tiada henti hentinya memberikan do'a demi terselesainya tugas akhir ini.
6. Seluruh teman-teman Program Teknologi Informasi Dan Ilmu Komputer Universitas Brawijaya.
7. Adi, Agus, Budi, Harim, Septian, Umar, Valey dan keluarga TIF-C 2011, serta teman-teman Progran Studi Informatika / Ilmu Komputer angkatan 2011.
8. Seluruh pihak yang telah membantu kelancaran penulisan skripsi ini yang tidak bisa disebutkan satu persatu.

Penulis menyadari bahwa tugas akhir ini masih banyak kekurangan. Oleh karena itu, saran dan kritik yang membangun sangat penulis harapkan. Semoga skripsi ini dapat bermanfaat bagi semua pihak yang menggunakannya.

Malang, 22 Mei 2015

Penulis

ABSTRAK

Muhamad Efendi, 2015 : Deteksi Plagiarisme Menggunakan Kombinasi Vector Space Model dan Jaccard Coefisien

Dosen Pembimbing : Indriati, ST., M.Kom dan Lailil Muflikhah, S.Kom., M.Sc

Plagiarisme merupakan masalah yang sangat berbahaya di era globlalisasi saat ini. Pada era globlalisasi konten digital mudah diakses oleh semua orang sehingga mudah mengalami *copy-paste* dan manipulasi. Diperlukan suatu sistem untuk dapat mengenali dokumen plagiarisme. Salah satu metode untuk mendeteksi plagiarisme dokumen teks adalah Vector Space Model (VSM). Metode ini terbukti cukup efektif untuk melakukan deteksi plagiarime dengan *recall* yang tinggi untuk mengelompokkan dokumen plagiarsme dan non-plagiarisme, namun kelemahannya *precision* yang dihasilkan masih rendah. Metode lain yang dapat digunakan adalah Jaccard Coefisien. Metode ini berkebalikan dengan VSM yaitu hasil pengelompokan dokumen plagiarisme dan non-plagiarisme memiliki *precision* yang tinggi, kelemahannya *recall* yang dihasilkan masih rendah.

Oleh karena itu pada penelitian ini digunakan kombinasi VSM-Jaccard untuk mendeteksi plagiarisme. Kombinasi VSM-Jaccard mampu meningkatkan *precision* dan *recall* sehingga *similarity* yang dihasilkan oleh kombinasi algoritma ini dapat mendekati akurat. Rata-rata *error similarity* yang dihasilkan oleh sistem ini adalah 2,45%, jika menggunakan VSM yaitu 8,23%, sedangkan untuk Jaccard sebesar 5,47%. Dan untuk mengantisipasi kata yang diganti dengan sinonimnya digunakan pendekatan *synonym recognition*. Penggunaan *synonym recognition* terbukti mampu menurunkan *error* deteksi plagiarisme terbukti *error* yang dihasilkan sebesar 2,45% sedangkan tanpa *synonym recognition* sebesar 3,39%. Perbedaan tersebut akan sangat terlihat terutama jika terdapat dokumen yang dirubah dengan sinonimnya.

Kata Kunci : Vector Space Model, Jaccard Coefisien, Plagiarisme

ABSTRACT

Muhamad Efendi, 2015 : Plagiarism Detection Using Combination Vector Space Model and Jaccard Coefisien

Advisors : Indriati, ST., M.Kom dan Lailil Muflikhah, S.Kom., M.Sc

Plagiarism is an issue that is very dangerous at globlalisasi era. The era of globlalisasi make digital content easily accessible to everyone so prone to copy-paste and manipulation. Needed a system to be able to recognize the document plagiarism. One method to detect plagiarism text document is the Vector Space Model (VSM). This method proved quite effective to detect plagiarism with high recall plagiarism for classifying documents into two class, namely plagiarism and non-plagiarism documents, but its weakness is the precisions result still low. Another method that can be used is Jaccard Coefisien. This method contrasts with VSM, the result of classifying plagiarism and non-plagiarism documents has high precision, but the recall of result is still low.

Therefore, this research used a combination of VSM-Jaccard to detect plagiarism. VSM-Jaccard combination can improve the precision and recall that similarity produced by the combination of these algorithm can be anywhere near accurate. The average error similarity produced this system is 2.45% compared with VSM is 8.23%, while for Jaccard of 5.47%, and to anticipate the words are replaced by synonyms this research used synonym recognition approach. Using of synonyms recognition proven reduce errors plagiarism detection with error generated by 2.45% while without synonyms recognition of 3.39%. The difference will be very visible, especially when there is document that was changed with its synonyms

Kata Kunci : Vector Space Model, Jaccard Coefisien, Plagiarism

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SOURCE CODE

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi dan informasi mengalami pertumbuhan yang sangat pesat dari tahun ke tahun di seluruh penjuru dunia. Hal tersebut menyebabkan proses pertukaran data dan informasi menjadi mudah dan cepat. Kemudahan dan cepatnya pertukaran data sangat menguntungkan bagi manusia untuk mendapatkan informasi dan pengetahuan secara luas. Meskipun demikian, kemudahan tersebut justru menjadikan konten digital mudah disalin (*copy-paste*) dan dimanipulasi. Hal inilah yang menyebabkan sangat memungkinkan terjadi plagiarisme.

Plagiarisme adalah penjiplakan atau pengambilan karangan, pendapat dan hasil karya orang lain serta menjadikannya seolah hasil karya sendiri tanpa mencantumkan referensi ke sumber aslinya [KBB-97]. Salah satu indikasi dokumen plagiarisme adalah memiliki kesamaan (*similarity*) yang tinggi dalam susunan kata dan makna kalimat dengan dokumen lain. Jadi semakin tinggi nilai kesamaannya semakin tinggi pula kemungkinan dokumen tersebut dikategorikan plagiat [ANZ-11]. Plagiarisme merupakan tindakan yang merugikan orang lain karena mengambil karya tanpa izin. Selain itu tindakan ini juga dapat menyebabkan timbulnya sifat malas dan menurunkan tingkat kreativitas. Pada saat ini tingkat plagiat di Indonesia terbilang tinggi. Para plagiat tidak hanya berasal dari mahasiswa tetapi juga kalangan doktor dan guru besar perguruan tinggi [RID-03]. Untuk menangani plagiarisme di Indonesia, pemerintah telah mengaturnya dalam kebijakan hukum seperti HAKI (Hak Atas Kekayaan Intelektual) dan UU ITE. Demi mengurangi tindak plagiarisme diperlukan aplikasi yang dapat mendeteksi adanya plagiarisme secara akurat.

Algoritma yang digunakan untuk mendeteksi plagiarisme cukup beragam antara lain Rabbin Karp, Jaccard Coefisien dan Vector Space Model (VSM). Pada penelitian menggunakan Vector Space Model yang dilakukan oleh Asif Ekbal,

Sriparna Saha dan Gaurav Choudhary pada tahun 2012 yaitu menggunakan perbandingan frekuensi kemunculan kata dan *inverse* dokumen frekuensi. Kemudian direpresentasikan ke dalam vektor kata untuk mendapatkan prosentase kemiripan antara dokumen uji dan dokumen asli. Pada penelitian ini memiliki kelebihan yaitu *recall* yang dihasilkan cukup tinggi untuk mengelompokkan dokumen plagiarisme dan bukan plagiarisme. Kelemahan algoritma ini adalah memiliki *precision* yang rendah jika diterapkan untuk mengelompokkan dokumen plagiarisme atau tidak [EKB-12]. Sedangkan penelitian menggunakan Jaccard Coefisien yang dilakukan oleh Ján Grman dan Rudolf Ravas pada tahun 2011 menggunakan perbandingan banyaknya kata yang sama dan total jumlah kata pada masing-masing dokumen asli dan dokumen uji. Pada penelitian ini memiliki kelebihan yaitu *precision* yang dihasilkan cukup tinggi untuk mengelompokkan dokumen plagiarisme dan bukan plagiarisme. Kelemahan algoritma ini adalah memiliki *recall* yang rendah jika diterapkan untuk mengelompokkan dokumen plagiarisme atau tidak [GRM-11].

Berdasarkan permasalahan dan penelitian sebelumnya, penulis ingin mengembangkan aplikasi untuk mendeteksi plagiarisme dokumen menggunakan kombinasi algoritma Jaccard Coefisien dan VSM. Berdasarkan penelitian yang dilakukan oleh Shuai Wang, Haoliang Qi, Leilei Kong dan Cuixia Du pada tahun 2013, kombinasi algoritma tersebut dapat menghasilkan deteksi plagiarisme yang lebih baik dari pada menggunakan Jaccard Coesfisien atau VSM saja. Kombinasi Jaccard Coefisien dan VSM mampu meningkatkan *precision* dan *recall* pada kumpulan dokumen, sehingga *similarity* yang dihasilkan lebih akurat [WAN-13]. Selain itu penambahan *synonym recognition* pada penelitian Mudafiq Riyan Pratama, Eko Budi Cahyono dan Gita Indah Marthasari diharapkan dapat meningkatkan deteksi plagiarisme terutama pada dokumen yang telah dirubah dengan sinonimnya [PRA-11].

Pada penelitian ini, sebelum kedua dokumen dapat dideteksi prosentase kemiripannya, terlebih dahulu dilakukan *pre-processing*. *Pre-processing* yang digunakan antara lain *tokenizing, filtering* dan *stemming*. Setelah itu dilakukan proses *synonym recognition* untuk meningkatkan deteki pada dokumen hasil plagiat yang dirubah dengan kata sinonimnya. Kemudian hasil dari *pre-processing*

dan *synonym recognition* dari kedua dokumen dilakukan pembobotan kata, kesamaan kata dan total kata penyusun pada masing-masing dokumen. Selanjutnya dilakukan perhitungan *similarity* menggunakan algoritma VSM lalu dengan Jaccard Coefisien. Perhitungan dari kedua algoritma tersebut kemudian dikombinasikan dengan suatu formula untuk didapatkan prosentase kemiripan yang baru.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, maka didapat rumusan masalah sebagai berikut:

1 Bagaimana menerapkan kombinasi algoritma VSM dan Jaccard Coefisien untuk dapat melakukan deteksi plagiarisme pada dokumen teks?
2 Bagaimana tingkat perbandingan algoritma VSM, Jaccard Coefisien dan kombinasi VSM-Jaccard dari segi *error similarity* yang dihasilkan?
3 Bagaimana pengaruh *synonym recognition* untuk mendeteksi plagiarisme?

1.3 Batasan Masalah

Batasan masalah yang digunakan untuk menghindari lebarnya permasalahan adalah sebagai berikut:

1. Penelitian hanya menguji data berupa teks dengan format txt.
2. Aplikasi tidak memperhatikan kesalahan penulisan pada dokumen.
3. Dokumen yang digunakan menggunakan Bahasa Indonesia.
4. Penelitian ini difokuskan pada plagiarisme parafrase dan *copy-paste* yang tidak mencantumkan sumber referensi.

1.4 Tujuan Penelitian

Tujuan dilakukan penelitian ini adalah mendeteksi plagiarisme pada dokumen teks bahasa Indonesia menggunakan kombinasi algoritma Vector Space Model dan Jaccard Coefisien.

1.5 Manfaat Penelitian

Penelitian ini diharapkan mempunyai manfaat sebagai berikut:

- Bagi Peneliti
  1. Sebagai media untuk pengimplementasian ilmu pengetahuan teknologi yang didapat di perkuliahan.
  2. Menambah wawasan tentang penerapan kombinasi metode Jaccard Coefisien dan VSM dalam deteksi plagiarisme pada dokumen guna mendapatkan hasil *similarity* dokumen yang sesuai.
- Bagi masyarakat
  1. Membantu mendeteksi plagiarisme dokumen teks bahasa Indonesia
  2. Sebagai bahan acuan saat ingin menerbitkan hasil karyanya.

1.6 Sistematika Peneltian

BAB I Pendahuluan

Berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat.

BAB II Tinjauan Pustaka

Menguraikan kajian pustaka dan teori-teori yang berhubungan dengan plagiarisme, Jaccard Coefisen, VSM dan teori-teori lain yang mendukung penelitian ini.

BAB III Metode dan Perancangan Sistem

Berisi tentang gambaran umum tahapan perancangan kombinasi algoritma Jaccard Coefisien dan VSM pada deteksi plagiarisme.

BAB IV Implementasi

Menjelaskan tentang implementasi sistem deteksi plagiarisme yaitu implementasi antarmuka dan *source code*.

BAB V Pengujian Dan Analisa

Berisi tentang proses dan hasil pengujian terhadap sistem yang telah diimplementasikan.

BAB VI Penutup

Berisi tentang kesimpulan dari hasil penelitian, serta saran-saran untuk pengembangan lebih lanjut.

BAB II
TINJAUAN PUSTAKA

Pada bab ini terdiri dari kajian pustaka dan dasar teori. Kajian pustaka membahas tentang metode dan hasil penelitian yang telah dilakukan orang lain. Kajian pustaka bertujuan untuk memahami penelitian sebelumnya tentang plagiarisme dengan metode VSM dan Jaccard. Dasar teori membahas teori yang diperlukan untuk menyusun penelitian. Dasar teori yang dibutuhkan berdasarkan latar belakang dan rumusan masalah adalah konsep plagiarisme, *text mining*, algoritma Jaccard Coefisien dan metode VSM serta perhitungan *similairity* kombinasi VSM-Jaccard.

2.1 Kajian Pustaka

Pada penelitian yang dilakukan oleh Asif Ekbal, Sriparna Saha dan Gaurav Choudhary dengan judul "*Plagiarism Detection in Text using Vector Space Model*" menyajikan penerapakan algoritma VSM untuk melakukan deteksi plagiarisme. Pada penelitian ini suatu dokumen akan dibandingkan dengan kumpulan dokumen untuk ditentukan tingkat plagiasinya dengan metode VSM. Dokumen yang dianggap memiliki *similarity* yang besar dengan dokumen uji (melebihi *threshold*) dianggap sebagai plagiat. Dari pengelompokan tersebut akan diperoleh *precision* dan *recall*. Hasil pengelompokan dokumen plagiarisme pada penelitian ini memiliki *recall* yang cukup tingggi namun kelemahannya *precision* yang dihasilkan rendah. *Recall* terbesar yang didapat adalah 26,93, sedangkan *precision* yang dihasilkan adalah 2,10. Dokumen yang digunakan pada penelitian ini berasal dari PAN-PC-11 [EKB-12].

Pada penelitian yang dilakukan oleh Ján Grman dan Rudolf Ravas dengan judul "*Improved Implementation for Finding Text Similarities in Large Collections of Data*" menyajikan penerapakan *improve* Jaccard Coefisien untuk melakukan deteksi plagiarisme. Pada penelitian ini suatu dokumen akan dibandingkan dengan kumpulan dokumen untuk ditentukan tingkat plagiasinya dengan metode Jaccard Coefisien. Dokumen yang dianggap memiliki *similarity*

yang besar dengan dokumen uji (melebihi *threshold*) dianggap sebagai plagiarisme. Dari pengelompokan tersebut akan diperoleh *precision* dan *recall*. Berkebalikan dengan VSM, hasil pengelompokan dokumen plagiarisme pada penelitian ini memiliki *precision* yang tinggi namun kelemahannya *recall* yang dihasilkan lebih rendah. *Precision* terbesar yang didapat adalah 0,892744, sedangkan *recall* yang diperoleh 0,473128. Dokumen yang digunakan pada penelitian ini berasal dari PAN-PC-11 [GRM-11].

Berdasarkan kekurangan kedua algoritma tersebut Shuai Wang, Haoliang Qi, Leilei Kong dan Cuixia Du dalam penelitiannya yang berjudul "*Combination of VSM and Jaccard Coefficient for External Plagiarism Detection"* menyajikan kombinasi algoritma VSM and Jaccard Coefficient. Hal tersebut dilakukan untuk mendapatkan *precision* dan *recall* yang sama-sama tinggi sehingga didapatkan *similarity* yang mendekati akurat. Pada penelitian tersebut didapatkan bahwa hasil *recall* dan *precision* yang diperoleh relatif seimbang yaitu untuk *precision* sebesar 0,89485 dan *recall* yang diperoleh sebesar 0,69429. Hal ini menunjukkan bahwa keakuratan algoritma dalam perhitungan *similarity* dipastikan mendekati akurat [WAN-13].

Sebagai tambahan untuk meningkatkan keakuratan *similarity* pada dokumen yang diubah dengan sinonimnya, Mudafiq Riyan Pratama, Eko Budi Cahyono dan Gita Indah Marthasari menggunakan *synonym recognition* dalam penelitiannya yang berjudul "Aplikasi Pendeteksi Duplikasi Dokumen Teks Bahasa Indonesia Menggunakan Algoritma Winnowing Dengan Metode K-Gram dan *Synonym Recognition*". *Synonym recognition* adalah metode yang dipakai untuk mendeteksi kata-kata yang mengandung sinonim. Pendekatan ini dilakukan untuk membantu meningkatkan tingkat kesamaan yang ditemukan dalam mendeteksi plagiarisme. Penggunaan *synonym recognition* terbukti mampu meningkatkan deteksi kesamaan kata dengan perbedaan mencapai kurang lebih 0,82 % lebih besar daripada tanpa menggunakan *synonim recognition* terutama terhadap dokumen yang dirubah dengan bentuk sinonimnya [PRA-11].

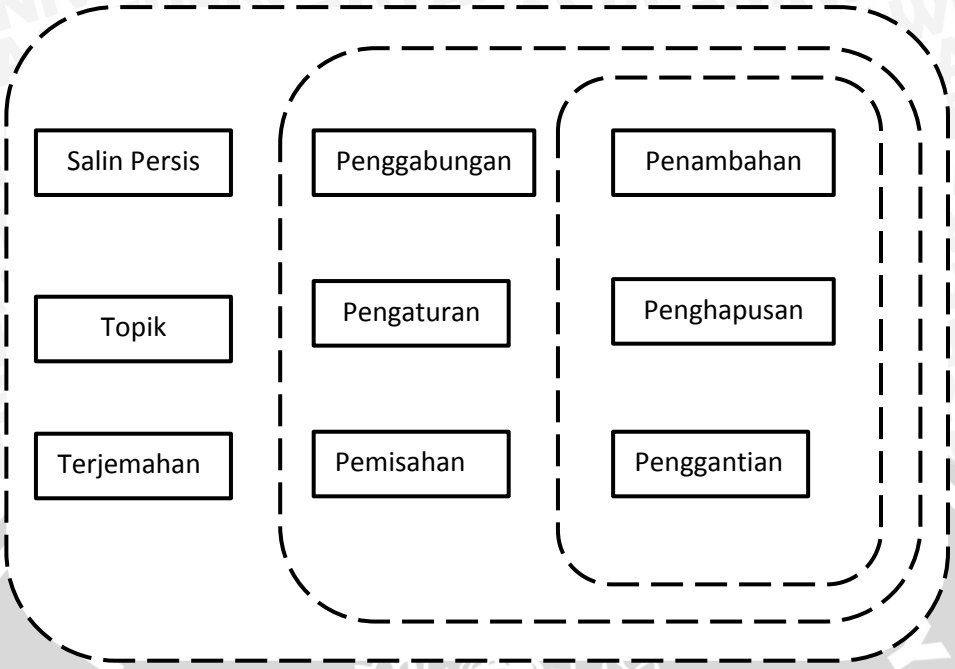
Perbedaan skripsi ini dengan penelitian yang dikaji adalah pada metode dan objek yang diteliti. Pada skripsi ini akan menggunakan gabungan metode VSM dan Jaccard Coefisien hanya saja tidak melakukan klasifikasi untuk

mengelompokkan dokumen plagiarisme dan non-plagiarisme, tetapi menggunakan prosentase kemiripan dengan suatu formula yang diperoleh pada penelitian tersebut. Selain itu juga penambahan *synonim recognition* pada penelitian Mudafiq dkk diharapkan akan meningkatkan prosentase *similarity* pada deteksi plagiarisme, terutama terhadap dokumen yang diubah menggunakan sinonimnya. Objek yang dipakai pada skripsi ini adalah dokumen teks Indonesia.

2.2 Plagiarisme

Plagiarisme berasal dari bahasa latin *Plagiari(us)* atau *Plagi(um)* yang berarti penculik/menculik, pembajak atau merampok. Menurut Kamus Besar Bahasa Indonesia (KBBI), plagiarisme adalah penjiplakan atau pengambilan karangan, pendapat atau ide orang lain dan menjadikannya seolah karangan atau pendapat sendiri [KBB-97]. Sedangkan menurut Sastroasmoro, plagiarisme adalah tindakan menyerahkan (*submitting*) atau menyajikan (*presenting*) ide maupun kata/kalimat orang lain tanpa menyebut sumbernya. Salah satu indikasi dokumen plagiarisme adalah memiliki kesamaan (*similarity*) yang tinggi dalam susunan kata dan makna dengan dokumen lain. Jadi semakin tinggi nilai kesamaannya semakin tinggi pula kemungkinan dokumen tersebut dikategorikan plagiat [ANZ-11].

Pelaku dari tindakan plagiarisme disebut plagiator [RID-03]. Plagiator dapat dihukum dengan berat seperti yang telah diatur pada Undang-Undang Nomor 20 Tahun 2003 dan Undang-Undang Nomor 14 Tahun 2005. Selain itu pada dunia akademik, jika plagiator merupakan mahasiswa/pelajar dapat dikeluarkan dari instansi pendidikannya. Sedangkan untuk guru besar bisa dicabut gelar pendidikannya dari instansi pendidikannya, hal ini tentu berakibat fatal pada karirnya tempat ia bekerja [ANO-14]. Suatu karya plagiat telah mengalami operasi-operasi tertentu baik keseluruhan atau pasial. Operasi-operasi tersebut antara lain salin persis, topik, terjemahan, penggabungan, pengaturan, pemisahan, penambahan, penghapusan dan penggantian. Pada Gambar 2.1 berikut ini menunjukkan pembagian operasi-operasi yang telah dilakukan pada dokumen plagiarisme.



Gambar 2.1 Operasi-operasi yang Digunakan dalam Plagiarisme
Sumber: [MAU-06]

2.2.1 Bentuk Plagiarisme

Bentuk dari plagiarisme sangat beragam yaitu meiliputi perubahan yang dilakukan secara keseluruhan atau secara parsial. Menurut Maurer yang termasuk dalam bentuk plagiarisme adalah sebagai berikut [MAU-06]:

a. Salin–tempel (*copy-paste*): penyalinan kata per kata secara langsung tanpa menyertakan sumber sama sekali. Menyalin terlalu banyak kata dan ide yang terlalu banyak juga merupakan bentuk plagiarisme meskipun disebutkan sumbernya.
b. Parafrase: mengubah kalimat dengan kata-kata berbeda namun memiliki makna yang sama tapi tidak mencantumkan sumbernya.
c. Menerjemahkan naskah asing tanpa refrensi sumber asli.
d. Plagiarisme artistik: menyajikan ide atau karya orang lain dalam bentuk lain misal tabel, gambar, diagram, dan lain-lain tanpa disebutkan sumbernya
e. Plagiarisme ide: pengambilan ide orang lain atau kelompok diskusi sebagai karya hasil pemikiran pribadi tanpa menyertakan peran/konstribusi orang lain.

f. Plagiarisme *source code*: menggunakan kode program orang lain tanpa izin atau sumber rujukan.
g. Penulisan tanda kutipan yang tidak tepat atau kesalahan pada prosedur penulisan rujukan sehingga memberikan informasi yang salah tentang referensi.

Sedangkan menurut pemerintah seperti yang dijelaskan dalam Peraturan Mentri Pendidikan Nasional Republik Indonesia Nomor 17 Tahun 2010 bahwa bentuk plagiat meliputi [PMP-10]:

a. Mengacu dan mengutip istilah, kata/kalimat, data/atau kalimat, data dan/atau informasi dari suatu sumber tanpa menyebutkan sumber dalam catatan kutipan dan/atau tanpa menyatakan sumber secara memadai.
b. Mengacu dan mengutip secara acak istilah, kata-kata dan/atau kalimat, data dan/atau informasi dari suatu sumber tanpa menyebutkan sumber dalam catatan kutipan da/atau tanpa menyatakan sumber secara memadai.
c. Menggunakan sumber gagasan, pendapat, pandangan atau teori tanpa menyatakan sumber secara memadai
d. Merumuskan dengan kata-kata dan kalimat sendiri dari sumber kata-kata dan kalimat, gagasan, pendapata atau teori tanpa menyatakan sumber secara memadai.
e. Menyerahkan suatu karya ilmiah yang dihasilkan dan/atau telah dipublikasikan oleh pihak lain sebagai karya ilmiahnya tanpa menyatakan sumber secara memadai.

2.2.2 Metode Pendeteksian Plagiarisme

Dalam mendeteksi plagiarisme diperlukan metode deteksi yang sesuai untuk dapat mengenali dokumen plagiarisme. Metode pendeteksian plagiarisme dibagi menjadi tiga, yaitu metode perbandingan teks lengkap, metode dokumen *fingerprint*, dan metode kesamaan kata kunci. Berikut ini adalah penjelasan dari masing-masing metode dan algoritma pendeteksi plagiarisme [STE-06]:

1. Perbandingan Teks Lengkap

   Metode ini diterapkan dengan membandingkan semua isi dokumen. Dapat diterapkan untuk dokumen yang kecil sampai besar. Pendekatan ini membutuhkan waktu yang lama terutama jika isi dokumen sangat banyak namun cukup efektif. Hal tersebut karena membandingkan secara keseluruhan isi dokumen. Metode perbandingan teks lengkap tidak dapat diterapkan untuk kumpulan dokumen yang tidak terdapat pada dokumen lokal. Algoritma yang digunakan pada metode ini adalah Vector Space Model, Jaccard Coefisien, Brute-Force, Edit Distance, Boyer-Moore dan Lavenshtein Distance.

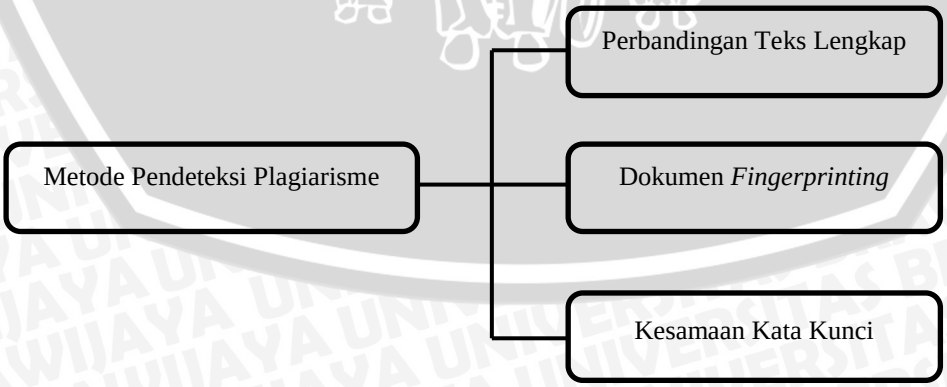
2. Dokumen *Fingerprinting*

   Dokumen *fingerprinting* merupakan metode yang digunakan untuk mendeteksi keakuratan salinan antar dokumen atau hanya sebagian teks saja. Prinsip kerja dari metode dokumen *fingerprinting* ini adalah dengan menggunakan teknik *hashing*. Teknik *hashing* adalah sebuah fungsi yang mengkonversi setiap string menjadi bilangan. Misalnya Rabin-Karp, Winnowing dan Manber.

3. Kesamaan Kata Kunci

   Prinsip dari metode ini adalah mengekstraksi kata kunci dari dokumen dan kemudian dibandingkan dengan kata kunci pada dokumen yang lain.

Pada Gambar 2.2 berikut menunjukkan pembagian metode deteksi plagiarisme.



Gambar 2.2 Metode Pendeteksi Plagiarisme
Sumber: [STE-06]

2.3 *Text Mining*

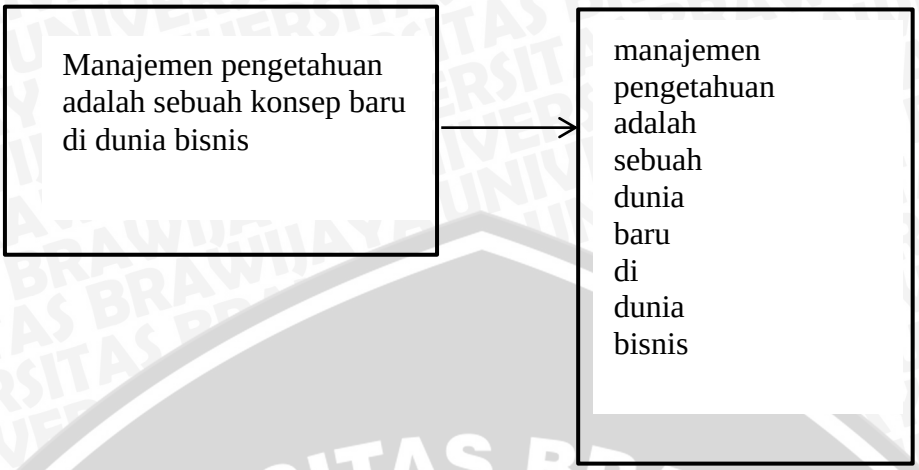
Text mining adalah proses penemuan pola yang sebelumnya tidak terlihat pada dokumen atau sumber tertentu menjadi pola yang diinginkan untuk tujuan tertentu [MUS-09]. *Text mining* sering digunakan untuk analisis informasi, pengambilan keputusan, dan tugas-tugas manajemen informasi lainnya yang terkait dengan data teks dalam jumlah besar. Sistem memanfaatkan peningkatan jumlah data yang tidak terstruktur dalam bentuk teks. Data tersebut diolah untuk memenuhi kebutuhan informasi menggunakan berbagai metode klasifikasi, klustering, analisis sentimen, dll [MOO-05].

Text mining adalah bagian dari *data mining.* Perbedaanya adalah proses *text mining* memerlukan lebih banyak tahapan dibanding *data mining.* Hal ini disebabkan karena data teks memiliki karakteristik yang lebih kompleks daripada data biasa sekalipun data teks tersebut sudah terstruktur. Berdasarkan ketidakteraturan struktur data teks, maka proses *text mining* memerlukan beberapa tahap awal (*pre-processing*) yang pada intinya adalah mempersiapkan agar teks dapat diubah menjadi lebih terstruktur [MUS-09].

2.4 *Pre-processing* Text Mining

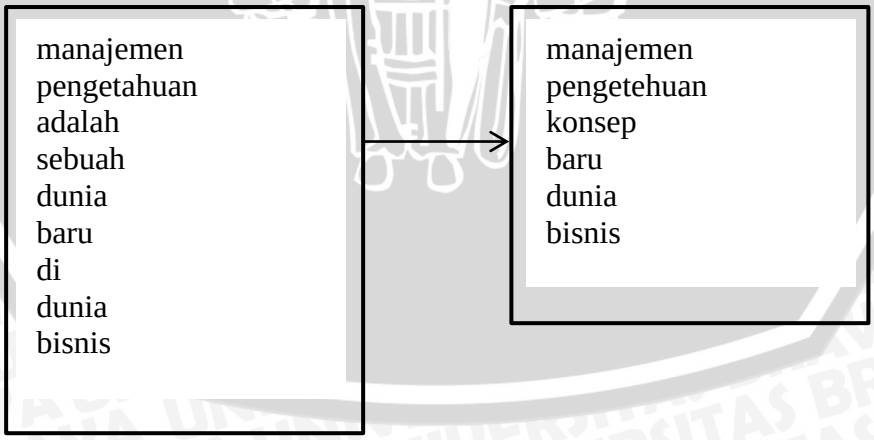
Pre-processing adalah langkah awal yang perlu dilakukan pada pemrosesan teks untuk menemukan fitur-fitur kata yang mewakili dokumen, Secara umum tahap *pre-processing* yang dilakukan dalam *text mining* pada dokumen, yaitu *tokenizing, filtering*, dan *stemming* [MUS-09].

Tahap *tokenizing* adalah tahap pemotongan *string* input berdasarkan tiap kata penyusunnya, penghilangan tanda baca dan merubah semua huruf ke dalam huruf kecil (*caseholding*) [MOO-05]. Proses *tokenizing* ditunjukkan pada Gambar 2.3 berikut ini.



Gambar 2.3 Proses *Tokenizing*
Sumber: [MOO-05]

Tahap *filtering* adalah tahap mengambil kata-kata penting dari hasil *tokenizing*. Prngambilan kata penting dapat menggunakan algoritma *stoplist* (membuang kata yang kurang penting) atau *wordlist* (menyimpan kata penting). *Stoplist/stopword* adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan *bag-of-word*. Contoh *stopwords* adalah "yang", "dan", "di", "dari", dan seterusnya [MOO-05]. Data *stopword* yang digunakan pada penlitian ini diambil dari jurnal Fadillah Z tala yang berjudul " *A Study of Stemming Effects on Information Retreival in Bahasa Indonesia*" [TAL-03]. Proses *filtering* seperti pada Gambar 2.4.



Gambar 2.4 Proses *Filtering*
Sumber: [MOO-05]

Tahap *stemming* adalah tahap mencari *root* kata dari tiap kata hasil *filtering*. Pada tahp ini dilakukan proses pengambilan berbagai bentukan kata kedalam

suatu representasi yang sama. Tahap ini kebanyakan dipakai untuk teks berbahasa Inggris dan lebih sulit diterapkan pada teks berbahasa Indonesia. Hal ini dikarenakan bahasa Indonesia tidak memiliki rumus bentuk baku yang permanen [MOO-05]. Pada penelitian ini, kamus kata dasar yang digunakan berasal dari Tesaurus Bahasa Indonesia Pusat Bahasa Indonesia. Proses tahapan *stemming* pada teks berbahasa Indonesia seperti pada Gambar 2.5.

Gambar 2.5 Proses *Stemming*
Sumber: [MOO-05]

Meskipun tidak memiliki rumus bentuk baku yang permanen, *stemming* Arifin terbukti efektif untuk merubah kata Indonesia kebentuk dasarnya. Adapun langkah-langkah *stemming* yang dilakukan oleh Arifin adalah sebagai berikut [ARI-06]:

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.

a. Langkah 1 :
   Cek apakah kata ada dalam kamus
   Ya : Sukses
   Tidak : lakukan pemotongan AW I
   Kata = permainkannya
b. Langkah 2 :
   Cek apakah kata ada dalam kamus
   Ya : Sukses
   Tidak : lakukan pemotongan AW II
   Kata = mainkannya
c. Langkah 3 :
   Cek apakah kata ada dalam kamus
   Ya : Sukses
   Tidak : lakukan pemotongan AK I
   Kata = mainkan
d. Langkah 4 :
   Cek apakah kata ada dalam kamus
   Ya : Sukses
   Tidak : lakukan pemotongan AK II
   Kata = main
   e. Langkah 5 :
   Cek apakah kata ada dalam kamus
   Ya : Sukses
   Tidak : lakukan pemotongan AK III. Dalam

hal ini AK III tidak ada, sehingga kata tidak diubah.

Kata = main

e. Langkah 6:

Cek apakah kata ada dalam kamus

Ya : Sukses

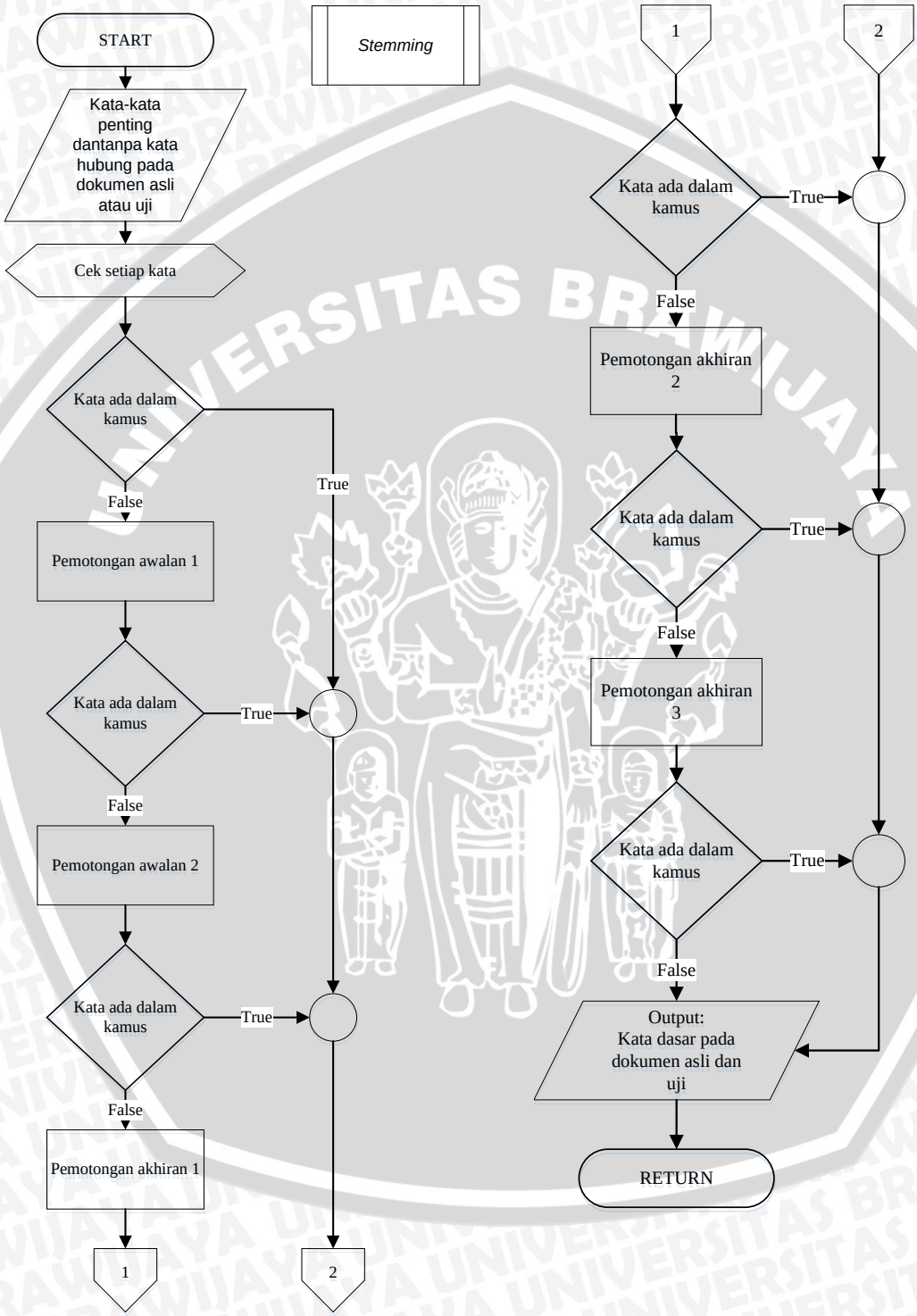
Tidak : "Kata tidak ditemukan"

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. 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 kata sudah tidak diperlukan lagi. Pemeriksaan 12 kombinasi ini diperlukan, karena adanya fenomena over*stemming* 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

dilakukan dapat dikembalikan sesuai posisinya. Alur tahapan *stemming* Arifin ditunjukkan pada Gambar 2.6

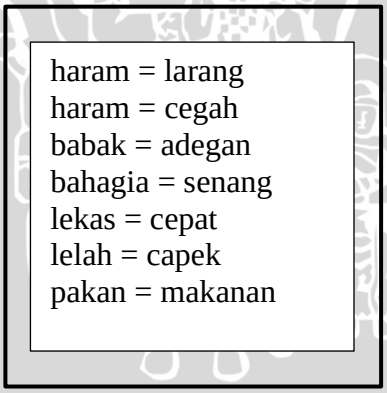


Gambar 2.6 Flowchart Tahap *Stemming* Arifin
Sumber: [ARI-06]

2.5 *Synonym recognition*

Synonym recognition atau pengenalan sinonim adalah salah satu pendekatan yang digunakan untuk membantu proses deteksi plagiarisme. *Synonym recogntion* digunakan untuk membantu meningkatkan prosentase *similarity* terutama pada dokumen yang telah mengalami perubahan ke dalam kata sinonimnya. Dokumen yang telah mengalami *pre-processing* dicocokkan dengan kamus sinonim. Jika kata tersebut terdapat pada kamus sinonim maka kata tersebut akan digantikan dengan sinonimnya. Namun jika kata tersebut tidak memiliki sinonim maka kata tersebut tidak perlu digantikan. Langkah ini akan terus berulang sampai seluruh pencocokan *string* pada dokumen selesai [PRA-11].

Pada penelitian ini, kamus sinonim yang digunakan berasal dari Tesaurus Bahasa Indonesia Pusat Bahasa Indonesia. Pada kamus tersebut terdapat berbagai istilah dalam bahasa Indonesia dilengkapi dengan sinonim maupun antonim. Kata sinonim yang digunakan hanya berupa kata dasar karena telah melewati tahapan *stemming*. Contoh kata-kata yang mengandung sinonim perhatikan Gambar 2.7 berikut ini [TES-08].

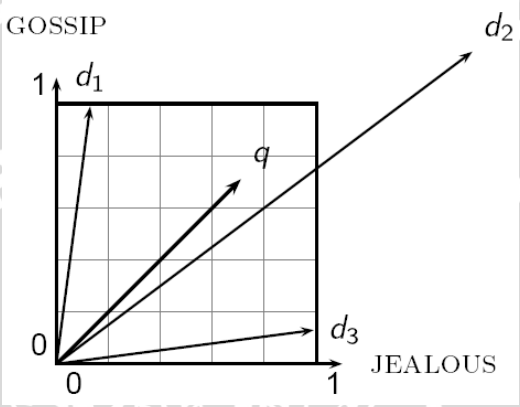


Gambar 2.7 Proses *Synonym Recognition*
Sumber: [TES-08]

2.6 Vector Space Model

Vector Space Model (VSM) adalah representasi kumpulan dokumen sebagai vektor dalam sebuah ruang vektor. Vector Space Model merupakan teknik dasar dalam perolehan informasi yang dapat digunakan untuk penelitian relevansi dokumen terhadap kata kunci pencarian (*query*) pada mesin pencarian, klasifikasi dokumen, dan pengelompokan dokumen. Kumpulan kata-kata dan dokumen

direpresentasikan dalam bentuk matriks kata dokumen. Pada Gambar 2.8 menunjukkan simulasi saat membuat vektor kata pada VSM, misalnya terdapat kumpulan dokumen(d_1, d_2, d_3) dan *query* (q). Lalu dibuat vektor kata-kata penyusunnya yaitu *gossip* dan *jealous*, sehingga akan terbentuk vektor dua dimensi. Vektor-vektor yang terbentuk pada masing-masing dokumen dan *query* berasal dari titik koordinat pembobotan kata *gossip* dan *jealous*.



Gambar 2.8 Proses Pembentukan Vektor Kata Dua Dimensi pada VSM
Sumber: [TES-08]

Pembobotan yang digunakan adalah TF.IDF. Pembobotan ini sangat bergantung pada frekuensi kata penyusun dokumen. Jika hanya membandingkan dua dokumen dapat menggunakan formula bobot TF.IDF di bawah ini [ARI-06]:

$$W_{tf} \begin{cases} 1 + \log TF & , jika\ TF > 0 \\ 0 & , lainnya \end{cases} \quad \text{.....(2.1)}$$

dimana:

W_{tf} : Bobot frekuensi kata

TF : Frekuensi kata penyusun dokumen

Pada vektor kata yang terbentuk berdasarkan pembobotannya dapat dilihat kedekatan dokumen berdasarkan sudut yang terbentuk antar vektor. Untuk menghitung kedekatan sudut antar dokumen pada VSM umumnya menggunakan *cosine similarity*. *Cosine similarity* merupakan kedekatan vektor kata dokumen berdasarkan sudut yang terbentuk antar dokumen. Tidak seperti *euclidean distance*, *cosine similarity* tidak memperhatikan panjang suatu vektor tetapi menghitung kedekatan sudut antar dua vektor yang terbentuk. Hal ini yang menyebabkan *cosine similarity* sering diterapkan pada pengaplikasian metode

VSM (Vector Space Model). Persamaan VSM (*cosine similarity*) adalah sebagai berikut [LHS-12]:

$$D(D_a, D_b) = \frac{\sum w_{a,j} . w_{b,j}}{\sqrt{\sum w_{a,j}^2 . \sum w_{b,j}^2}} \quad (2.2)$$

dimana:

$D(D_a, D_b)$: *similarity* VSM dokumen a dan dokumen b

$w_{a,j}$: TF.IDF kata ke-j di dokumen a

$w_{b,j}$: TF.IDF kata ke-j di dokumen b

2.7 Jaccard Coefisien

Jaccard Coefisien adalah statistik yang digunakan untuk membandingkan kesamaan dan keragaman sampel set. Metode ini awalnya ditemukan oleh Paul Jaccard yang ingin menguji keragaman sampel penelitiannya. Jaccard Coefisien didefinisikan sebagai ukuran persimpangan dibagi dengan ukuran persatuan sampel set. Secara teori *text mining*, Jaccard Coefisien adalah hasil bagi antara *irisan* dari dua dokumen dengan *union* dari dua dokumen tersebut. Persamaan Jaccard Coefisien adalah sebagai berikut [GRM-12]:

$$J(D_a, D_b) = \frac{D_a \cap D_b}{D_a \cup D_b} \quad (2.3)$$

dimana:

$J(D_a, D_b)$: *similarity* Jaccard dokumen a dan b

$D_a \cap D_b$: banyaknya kata yang sama antara dokumen a dan b

$D_a \cup D_b$: gabungan kata antara dokumen a dan b

Pada penelitian "*Combination of VSM and Jaccard Coefficient for External Plagiarism Detection*" menggunakan Jaccard Coefisien yang telah dimodifikasi karena koefisien Jaccard asli menghilangkan perbedaan panjang sampel asli, padahal panjang sampel merupakan salah satu fitur plagiarisme yang penting dalam deteksi plagiarisme. *Improve* Jaccard Coefisien tersebut dilakukan dengan menjumlah minimum kata yang sama dikalikan dengan dua lalu dibagi dengan total jumlah kata pada kedua dokumen. Pendefisian rumus *improve* Jaccard Coefisien adalah sebagai berikut [WAN-13]:

$$J(D_a, D_b) = \frac{2\,(\sum \min(D_a, D_b)}{D_a + D_c} \quad (2.4)$$

dimana:

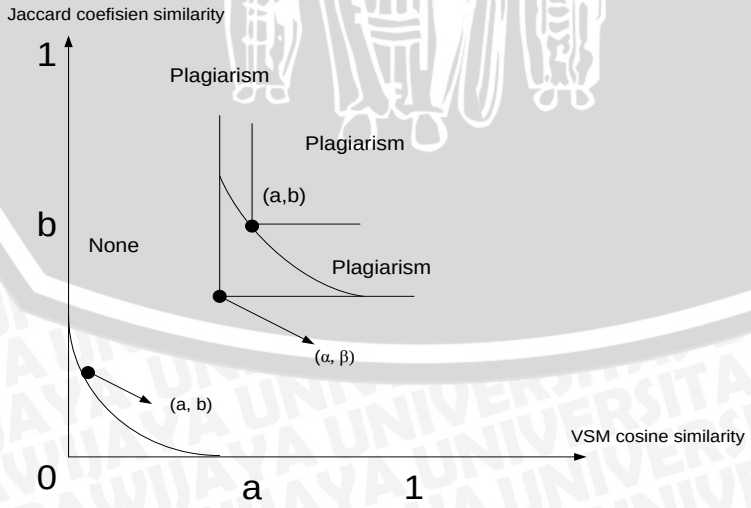
$J(D_a, D_b)$: *similarity* Jaccard dokumen a dan b

$\min(D_a, D_b)$: minimum jumlah kata yang sama pada dokumen a dan b

$D_a + D_b$: jumlah semua kata antara dokumen a dan b

2.8 Perhitungan Kesamaan Dokumen (*Similarity*) dan *Error Similarity*

Perhitungan *similaity* didapatkan dengan mengkombinasikan *similarity* dari Jaccard Coefisien dan VSM. Perhitungan *similarity* dengan VSM atau Jacard saja tidak cukup untuk melakukan deteksi plagiarisme pada dokumen. Terbukti dengan hasil perhitungan menggunakan Jaccard saja memiliki *precision* yang rendah sedangkan jika menggunakan VSM saja memiliki *recall* yang rendah. Oleh karena itu digunakan formula yang mengkombinasikan antara algoritma VSM dan Jaccard Coefisien. Kombinasi tersebut dapat memisahkan antara dokumen plagiarisme dan bukan plagiarisme dengan fungsi yang sesuai (*fitting function*). Setelah itu model *fitting function* dilakukan pelatihan pada koordinat fungsi invers yaitu (α, β) untuk membangun *fitting function* yang benar-benar dapat memisahkan plagiarisme dan non-plagiarisme. Sedangkan sebagai konstanta *tortuosity degree* (K) tidak berpengaruh [WAN-13]. Pada Gambar 2.9 menunjukkan penambahan parameter α dan β untuk memisahkan dokumen plagiarisme dan non-plagiarisme.



Gambar 2.9 Proses Batasan Koordinat Plagiarisme dan Non-plagiarisme
Sumber: [WAN-13]

Nilai parameter α, β dan K diperoleh dari pengujian untuk didapatkan hasil yang optimal. Nilai variabel α dan β merupakan bilangan bulat kecil dengan rentang antara 0,1-0,9. Sedangkan nilai variabel K merupakan bilangan bulat kecil dengan rentang antara 0,01-0.09. Maka persamaan *hybrid similarity* dapat dituliskan sebagai berikut [WAN-13]:

$$Sim(D_a, D_b) = \frac{(D(D_a,D_b)-\alpha)*(J(D_a,D_b)-\beta)}{K} \quad (2.5)$$

dimana:

$Sim(D_a, D_b)$	:	*Hybrid similarity* dokumen a dan b
$D(D_a, D_b)$	:	*Similarity* VSM dokumen a dan dokumen b
$J(D_a, D_b)$	:	*similarity* jaccard dokumen a dan b
α	:	titik seimbang horisontal *(offset of horizontal axis)*
β	:	titik seimbang vertikal *(offset of vertical axis)*
K	:	faktor skala (*scale factor*)

Setelah didapatkan nilai *hybrid similarity* diperlukan normalisasi hasil untuk mendapatkan *similarity* dokumen asli dan dokumen uji. Normalisasi adalah transformasi nilai bilangan untuk mendapatkan *range* tertentu atau antara 0 sampai 1. Normalisasi yang digunakan adalah normalisasi *min-max* karena normalisasi ini memungkinkan hasil *hybrid similarity* tidak bernilai bias (keluar dari range). Formula dari normalisasi ini adalah sebagai berikut [WAN-13]:

$$Norm\ Sim(D_a, D_b) = \frac{Sim(D_a,D_b)-min}{max-min} \quad (2.6)$$

$$\max = \frac{(1-\alpha)(1-\beta)}{K} \quad (2.7)$$

dimana:

$Norm\ Sim(D_a, D_b)$	:	*Similarity* dokumen a dan b hasil normalisasi
min	:	Nilai *similarity* minimum yaitu selalu bernilai 0
max	:	Nilai *similarity* maksimal yang mungkin

Pada penelitian ini juga dilakukan perhitungan *error similarity*. *Error similarity* adalah jumlah kesalahan deteksi *similarity* yang dihasilkan oleh sistem. *Error similarity* juga dapat diartikan sebagai selisih antar *similarity* yang dihasilkan oleh sistem dan *similarity* sebenarnya. Perhitungan *error similarity*

dilakukan untuk mengetahui kesalahn deteksi yang dilakukan oleh sistem. Formula untuk menghitung *error similarity* adalah sebagai berikut [SAN-14]:

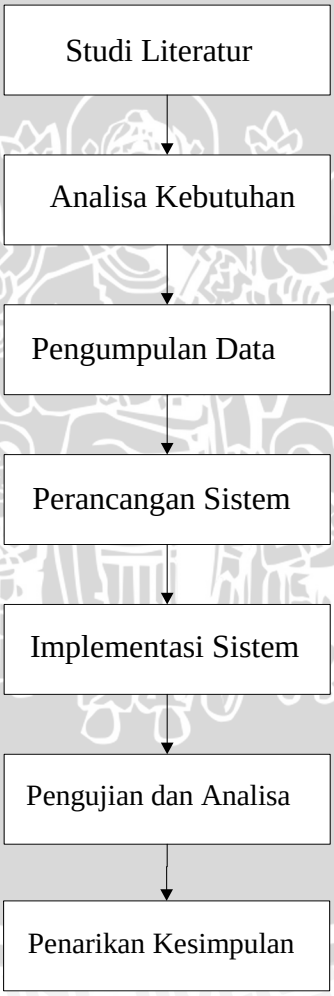
$$Error\ Sim = |\ Norm\ Sim\ (D_a, D_b) - Similarity\ Asli(D_a, D_b)\ |\qquad (2.8)$$

dimana:

$Error\ Sim$	:	Kesalahan deteksi yang diperoleh sistem
$Norm\ Sim\ (D_a, D_b)$	:	*Similarity* dokumen a dan b yang diperoleh sistem
$Similarity\ Asli(D_a, D_b)$	:	*Similarity* sebenarnya dokumen a dan b

BAB III
METODE DAN PERANCANGAN SISTEM

Pada bab ini akan dibahas metode yang digunakan dalam penelitian dengan menggunakan kombinasi algoritma Jaccard Coefisien dan VSM. Alur metode yang akan digunakan adalah kajian literatur, analisa kebutuhan, pengumpulan data, desain sistem, implementasi sistem, pengujian dan analisa serta penarikan kesimpulan.



Gambar 3.1 Diagram Alir Metode Penelitian
Sumber: [Perancangan]

3.1 Studi Literatur

Studi literatur bertujuan untuk mempelajari tentang cara perhitungan algoritma Jaccard Coefisien dan VSM serta penerapannya dalam mendeteksi plagiarisme. Selain itu studi literatur juga digunakan untuk membandingkan dengan penelitian-penelitian sebelumnya. Literatur-literatur yang digunakan tersebut diperoleh dari buku, paper nasional maupun internasional dan dokumentasi internet. Literatur pendukung utama yang diperlukan untuk penelitian ini adalah :

a. *Text Mining*
b. Plagiarisme
c. Jaccard Coefisien
d. VSM (Vector Space Model)

3.2 Analisa Kebutuhan

Analisa kebutuhan dilakukan untuk mendapatkan semua kebutuhan yang diperlukan oleh sistem yang akan dibangun. Cara kerja aplikasi deteksi plagiarisme dokumen yaitu pertama user menginputkan dokumen asli dan dokumen yang akan diuji. Setelah itu sistem akan melakukan *pre-processing* secara otomatis pada dua dokumen tersebut. Proses *pre-processing* yang dilakukan antara lain *tokenizing*, *filtering* dan *stemming*. Proses selanjutnya sistem akan menghitung prosentasi *similarity* dengan menggunakan Algoritma VSM. Sistem juga akan menghitung *similarity* antar dokumen dengan algoritma Jaccard. Kemudian hasil dari kedua algoritma tersebut dikombinasikan dengan suatu formula untuk mendapatkan *similarity* kombinasi dan akan ditampilkan pada aplikasi.

3.3 Pengumpulan Data

Pada tahap ini dilakukan pengumpulan data berupa dokumen teks yang akan digunakan sebagai dokumen asli dan dokumen uji. Data diperoleh dari penelitian sebelumnya yang dilakukan oleh Cristian S.K.A dengan judul "Perancangan Sistem Deteksi Plagiarisme Dokumen Teks Menggunakan Algoritma Damerau Levenshtein Distance". Selain itu juga pengumpulan data

stopword yaitu berisi kata-kata yang kurang memiliki makna. *Stopword* ini digunakan untuk menyaring kata-kata pada proses *filtering*. Serta kamus kata dasar untuk merubah kata berimbuhan ke dalam kata dasarnya pada proses *stemming*. Pada tahap ini juga dilakukan penentuan kata yang memiliki sinomin untuk proses *synonim recognition* berdasarkan Kamus Pusat Bahasa Indonesia.

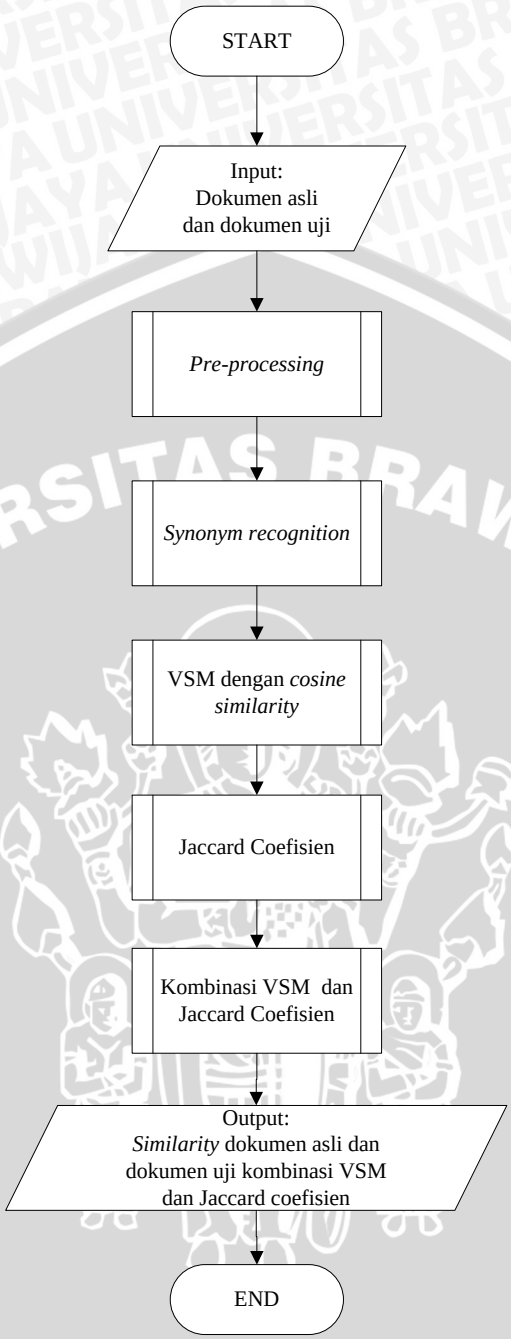
3.4 Perancangan Sistem

Perancangan sistem mengacu pada studi literatur, pengumpulan data dan analisa kebutuhan yang telah dilakukan sebelumnya. Hasil dari perancangan sistem akan digunakan sebagai pedoman pada tahap implementasi. Pada tahap perancangan sistem terdapat perancangan alur algoritma dan manualisasi program.

3.4.1 Alur Algoritma

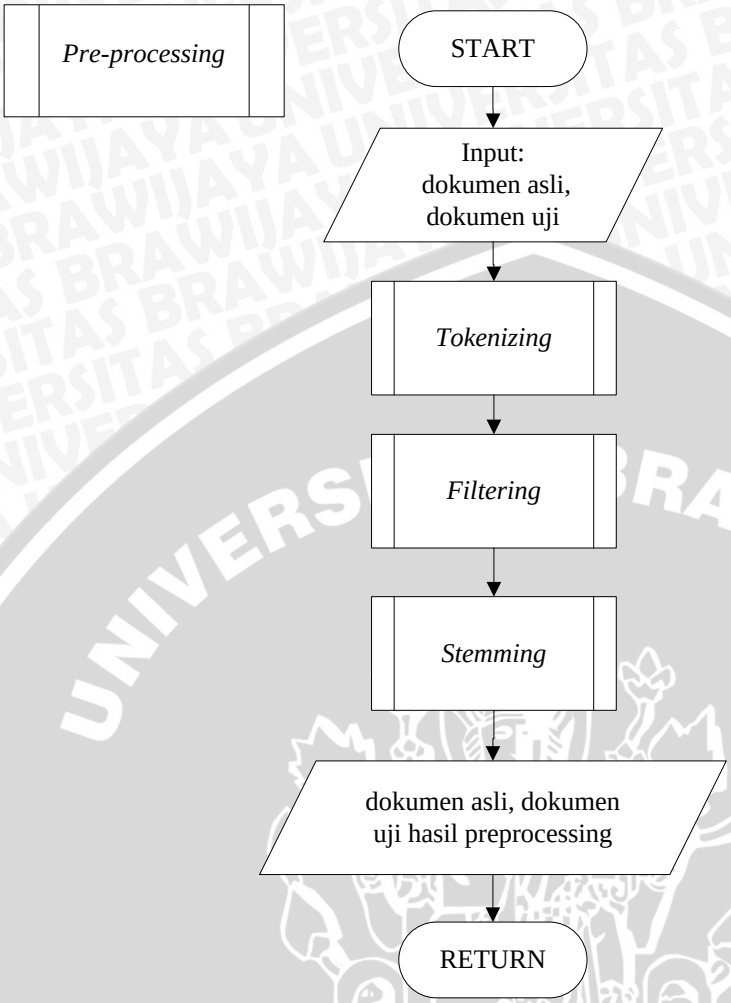
Perancangan cara kerja deteksi plagiarisme pada penelitian ini secara garis besar memiliki lima alur algoritma. Tahap pertama adalah *pre-processing*, tahap kedua adalah *synomym recognition*, tahap ketiga adalah perhitungan VSM dengan *cosine similarity*, tahap keempat adalah perhitungan Jaccard Coefisien dan tahap kelima adalah perhitungan *similarity* kombinasi kedua algoritma.

Tahap *pre-processing* bertujuan untuk mengurangi *noise* pada dokumen, seperti merubah semua huruf kedalam huruf kecil (*tokenizing*), menghilangkan simbol/tanda baca, menghilangkan kata tidak penting (*filtering*) dan merubah huruf kebentuk dasarnya (*stemming*). Selain itu dilakukan *synonim recognition* setelah tahap *pre-processing* dilakukan guna menghilangkan kata-kata berbeda dalam penulisannya namun memiliki makna sama. Setelah itu dilakukan perhitungan kesamaan dengan metode Vector Space Model. Lalu dilakukan perhitungan kesamaan dengan metode Jaccard Coefisien. Hasil kesamaan dari kedua metode tersebut dilakukan kombinasi guna mendapatkan *similarity* baru. Pada Gambar 3.2 berikut ini merupakan alur diagram keseluruhan sistem.



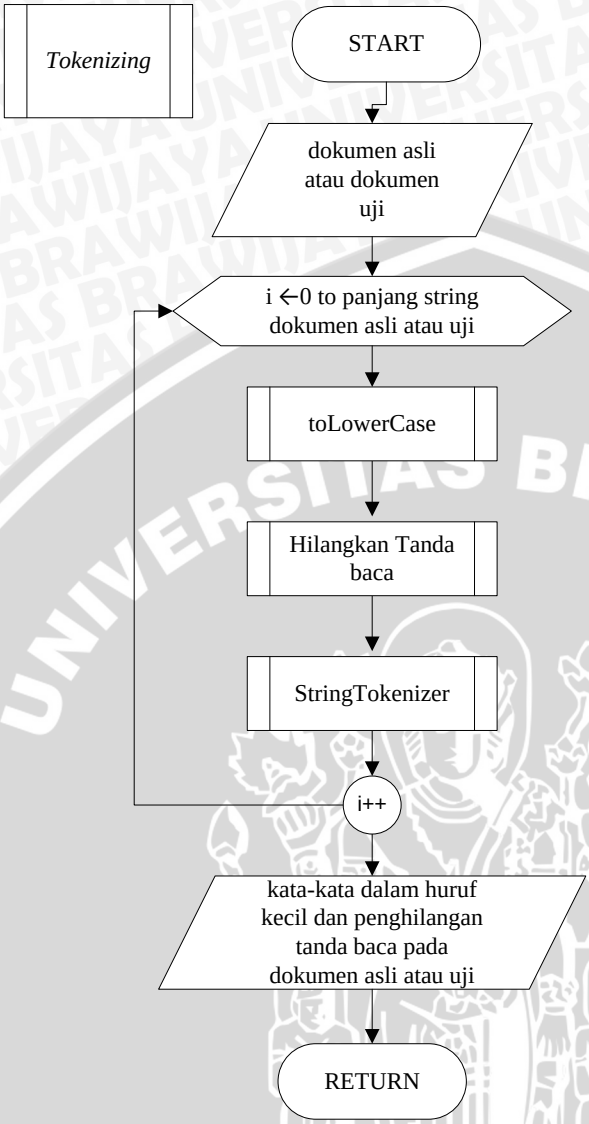
Gambar 3.2 Flowchart Keseluruhan Sistem Deteksi Plagiarisme
Sumber: [Perancangan]

Tahap awal yaitu *pre-processing*, tahap ini dilakukan pada dokumen asli dan dokumen uji. Pada tahap ini terdapat tiga tahapan yaitu *tokenizing, filtering* dan *stemming*. Pada Gambar 3.3 berikut ini merupakan diagram alir tahap *pre-processing*



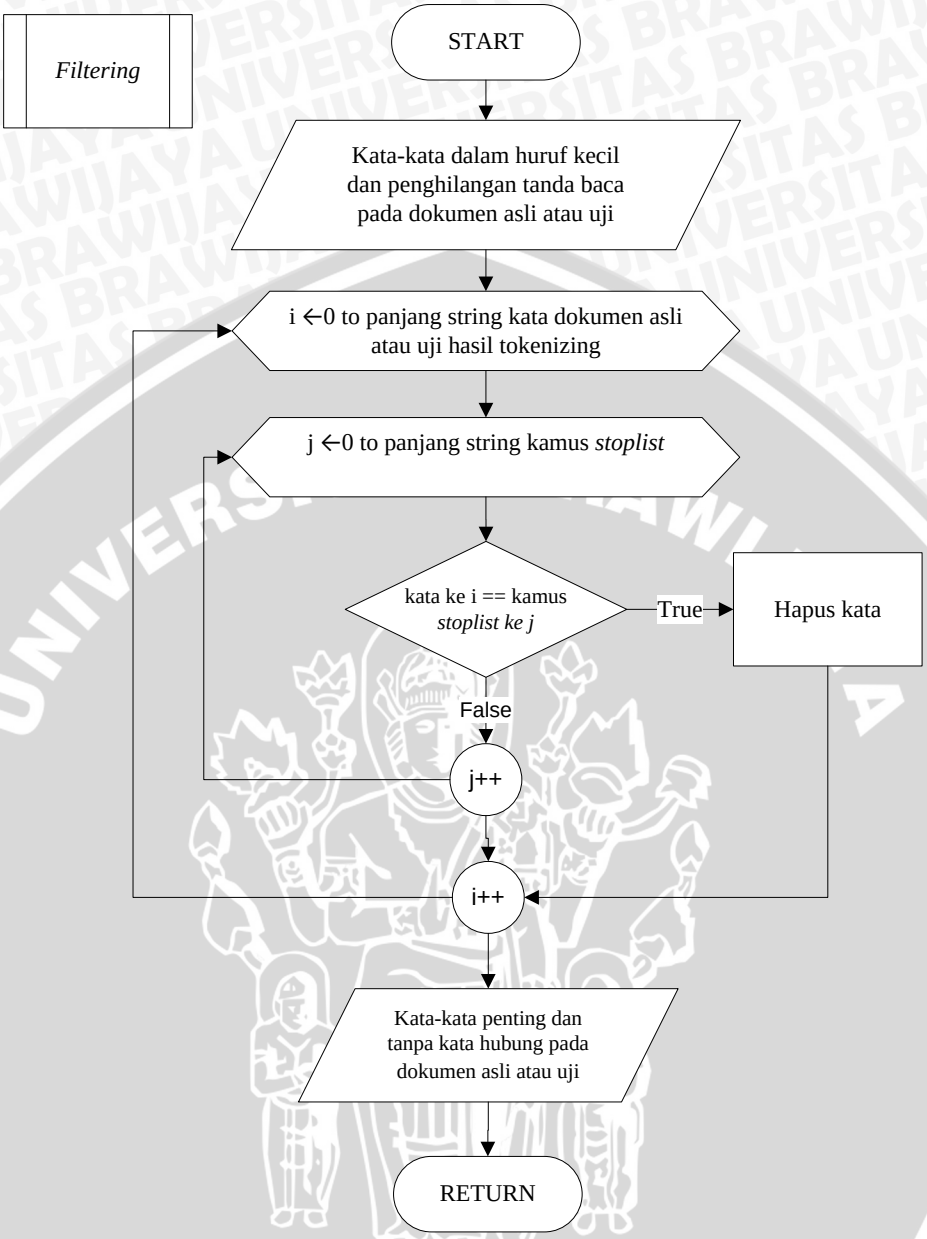
Gambar 3.3 Flowchart Tahap *Pre-processing*
Sumber: [Perancangan]

Tahap awal pada *pre-processing* adalah *tokenizing*. Pada tahap ini terjadi *case folding* (perubahan ke dalam huruf kecil) dan penghilangan tanda baca. Tanda baca yang dihilangkan antara lain titik, koma, titik dua, garis miring, tanda petik dan lain-lain. Pada Gambar 3.4 berikut ini merupakan diagram alir tahap *tokenizing*.



Gambar 3.4 Flowchart Tahap *Tokenizing*
Sumber: [Perancangan]

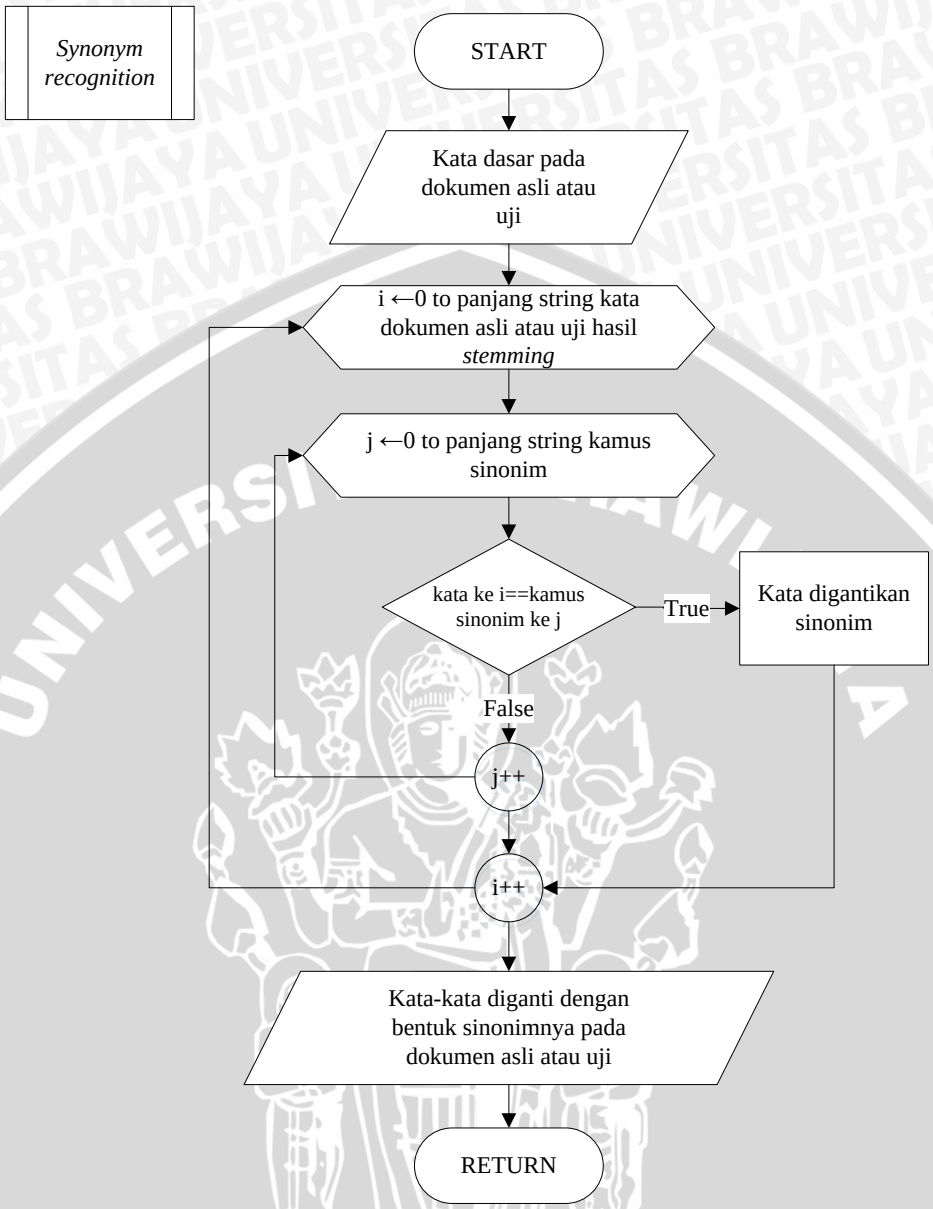
Tahap kedua pada *pre-processing* adalah *filtering*. Pada tahap ini terjadi penghilangan kata yang tidak penting, seperti kata penghubung (dari, di, yang, dll). Kata yang tidak penting tersimpan pada kamus *stoplist*. Pada Gambar 3.5 berikut ini merupakan diagram alir tahap *filtering*.



Gambar 3.5 Flowchart Tahap *Filtering*
Sumber: [Perancangan]

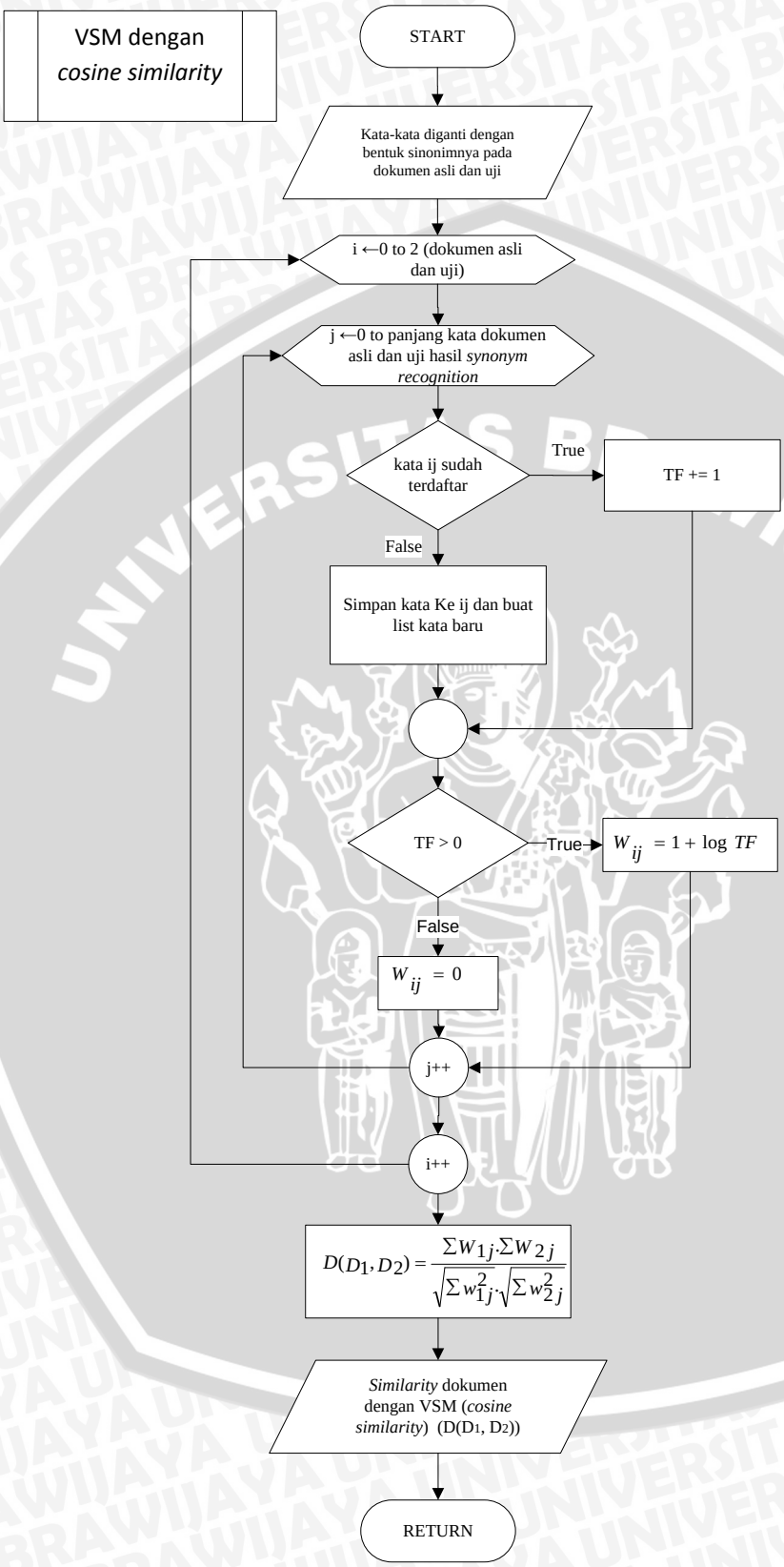
Tahap terakhir pada *pre-processing* adalah *stemming*. Pada tahap ini terjadi pengembalian kata yang lolos pada tahap *filtering* ke dalam kata dasarnya. Alur algoritma *stemming* Arifin dijelaskan pada Gambar 2.7 bab dua.

Setelah melalui tahap *pre-processing* dilakukan *synonym recognition*. Tahap ini merubaha kata yang telah mengalami *pre-processing* ke dalam bentuk sinonimnya. Pada Gambar 3.6 berikut ini merupakan diagram alir tahap *synonym recognition*.



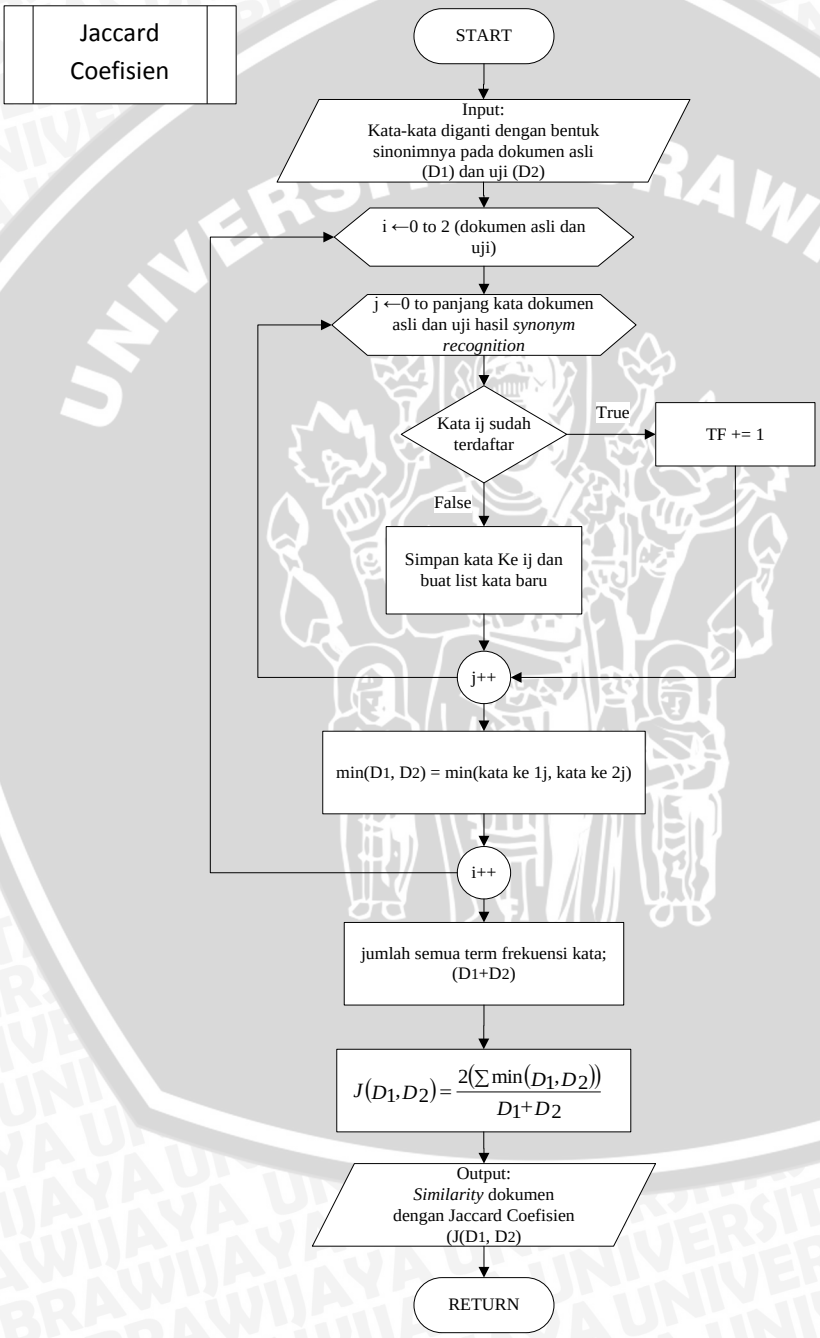
Gambar 3.6 Flowchart Tahap *Synonym recognition*
Sumber: [Perancangan]

Setelah didapatkan dokumen hasil *preprocesing* dan *synonym recognition*, tahap selanjutnya yaitu menghitung *similarity* antara dokumen asli dan dokumen uji dengan algoritma VSM. Tahap awal pada algoritma ini adalah membuat vektor pada semua kata dalam *weighting* (pembobotan TF.IDF) pada masing-masing dokumen asli dan dokumen uji. Setelah didapat pembobotannya, dapat dimasukkan dalam formula VSM. Pada Gambar 3.7 berikut ini merupakan diagram alir tahap perhitungan *similarity* dengan VSM (*cosine similarity*).



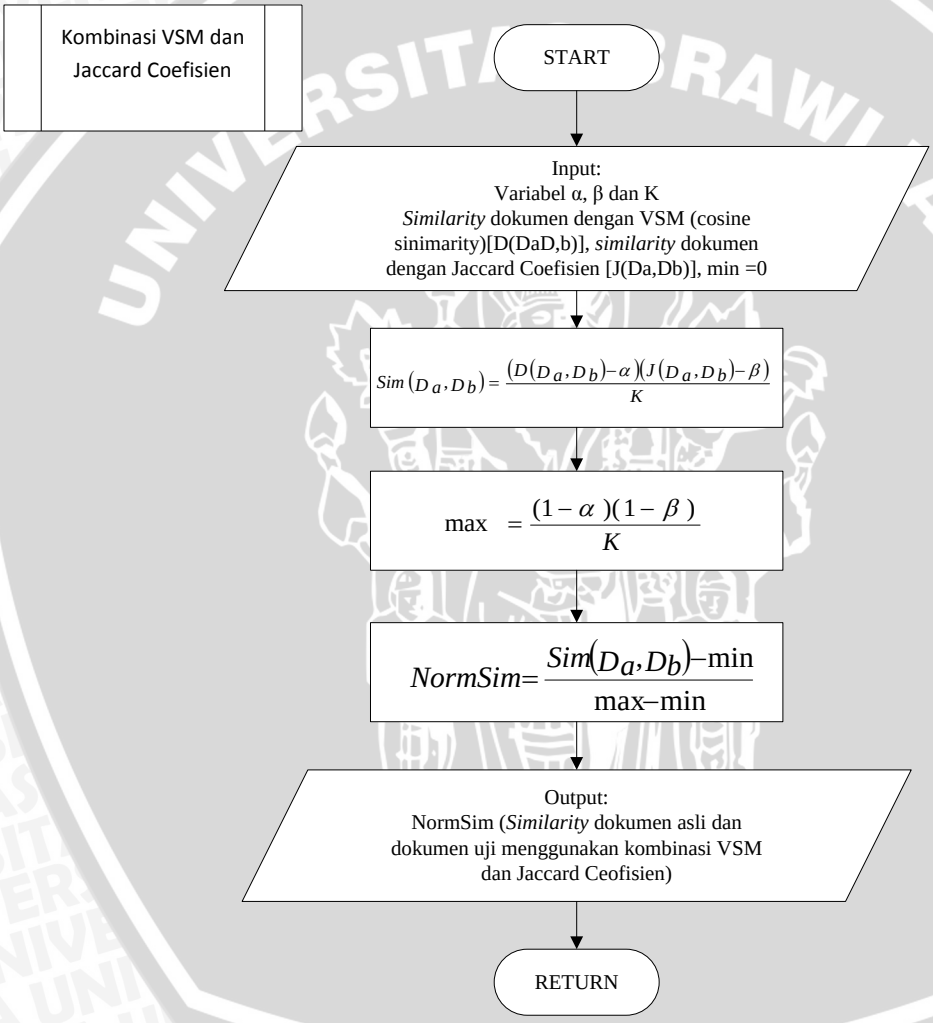
Gambar 3.7 Flowchart Tahap Perhitungan *Symilariy* Dokumen dengan VSM (*cosine similarity*)
Sumber: [Perancangan]

Pada tahap keempat yaitu menghitung *similarity* antara dokumen asli dan dokumen uji dengan algoritma Jaccard Coefisien. Menghitung jumlah kata yang sama dan total semua kata antara dokumen asli dan dokumen uji hasil *synonym recognition*. Setelah itu dihitung dengan formula Jaccard Coefisien. Pada Gambar 3.8 berikut ini merupakan diagram alir *similarity* dengan Jaccard Coefisien.



Gambar 3.8 Flowchart Tahap Perhitungan *Symilarity* Dokumen dengan Jaccard Coefisien
Sumber: [Perancangan]

Tahap terakhir yaitu dilakukan kombinasi antara algoritma VSM dan Jaccard Coefisien. Kombinasi keduanya didasarkan pada penelitian yang dilakukan Shuai Wang, Haoliang Qi, Leilei Kong, Cuixia Du yaitu *similarity* VSM dikurangi dengan koefisien α dikalikan dengan hasil *similarity* Jaccard dikurangi koefisien β kemudian dibagikan dengan koefisien K. Ketiga koefisien tersebut diperoleh berdasarkan uji coba. Pada Gambar 3.9 berikut ini menjelaskan alur kombinasi algoritma VSM dan Jaccard Coefisien.



Gambar 3.9 Flowchart Tahap Perhitungan *Similarity* Dokumen Kombinasi VSM dan Jaccard Coefisien
Sumber: [Perancangan]

3.4.2 Perhitungan Manual:

Pada sub bab ini akan dijelaskan perhitungan secara manual algoritma deteksi plagiarisme pada sistem ini, dimulai dari tahapan *pre-processing* sampai didapatkan *similarity* dokumen uji dan dokumen asli kombinasi VSM-Jaccard. Contoh perhitungan manualnya yaitu diketahui dua dokumen yang akan dihitung tingkat plagiat dokumen uji terhadap dokumen asli sebagai berikut.

Dokumen Asli (A-Asli):

Penderita asma yang kekurangan vitamin D akan mengalami pemburukan penyakit asma.

Defisiensi vitamin ini diduga akan menghalangi reaksi terhadap pengobatan steroid yang sering dipakai dalam obat pelega dan pengontrol asma.

Dalam risetnya, para peneliti di National Jewish Health (NJH) di Denver mengukur tingkat vitamin D dari 50 penderita asma dan menilai fungsi paru-paru, hyper-responsiveness saluran udara, yang lazim terjadi di dalam pengerutan saluran udara, dan reaksi terhadap pengobatan steroid.

Ternyata, pasien yang mengalami defisiensi vitamin D memperlihatkan hasil yang buruk dalam pemeriksaan fungsi paru-paru dan hyper-responsiveness saluran udara.

Lebih dari itu, pasien yang kadar vitamin D dalam tubuhnya di bawah 30 nanogram/mL kadar hyper-responsiveness saluran udara hampir dua kali lipat, dibandingkan dengan mereka yang memiliki lebih banyak vitamin D di dalam darah mereka.

Tingkat vitamin D yang rendah juga berkaitan dengan reaksi yang lebih buruk terhadap pengobatan steroid dan peningkatan produksi sitokin pro-peradangan, TNF-alpha.

Hal itu meningkatkan kemungkinan bahwa tingkat vitamin D yang rendah berkaitan dengan peningkatan peradangan saluran udara, kata para peneliti itu.

Selain itu, kadar vitamin D meramalkan seberapa baik seseorang akan bereaksi terhadap pengobatan steroid bagi asma.

"Itu mungkin terjadi karena vitamin D bertindak sebagai pengubah reaksi steroid dengan cara yang sejalan dengan orang yang menderita asma," kata Dr E

Rand Sutherland, dari divisi paru-paru dan perawatan medis kritis di NJH.

Peserta paling parah memiliki tingkat vitamin D paling rendah, kata mereka.

Asma berkaitan dengan kegemukan, dan kekurangan vitamin D mungkin menjadi faktor yang menghubungkan kedua kondisi tersebut, kata Sutherland.

"Memulihkan tingkat normal vitamin D pada orang yang menderita asma mungkin membantu meningkatkan asma mereka," kata Sutherland.

Namun, tidak diketahui apakah asupan vitamin D akan membantu penderita asma, katanya.

Asupan vitamin D buat orang dewasa yang disarankan ialah 300 IU sampai 600 IU, tergantung pada usia, kata US National Institutes of Health.

Dokumen Uji (A.20-kata):

Penderita asma kekurangan vitamin D mengalami pemburukan penyakit.

Defisiensi vitamin ini diduga akan menghalangi reaksi terhadap pengobatan steroid dipakai obat dan pengontrol asma.

Dalam para peneliti di Jewish Health di Denver tingkat vitamin D dari 50 asma dan menilai fungsi paru-paru, hyper-responsiveness saluran udara, yang lazim terjadi di dalam saluran udara, dan reaksi terhadap pengobatan.

Ternyata, pasien yang mengalami vitamin D yang buruk dalam pemeriksaan fungsi paru-paru dan hyper-responsiveness.

Lebih itu, pasien yang kadar vitamin D dalam tubuhnya di bawah 30 nanogram/mL hyper-responsiveness hampir dua lipat, dibandingkan dengan mereka yang memiliki lebih banyak vitamin D di dalam darah mereka.

Tingkat D yang rendah berkaitan dengan reaksi lebih buruk terhadap pengobatan dan sitokin pro-peradangan, TNF-alpha.

Hal itu meningkatkan kemungkinan bahwa tingkat D yang rendah berkaitan peningkatan peradangan saluran udara, kata peneliti itu.

itu, kadar D meramalkan seberapa baik seseorang akan bereaksi terhadap asma.

"Itu mungkin karena D bertindak sebagai pengubah reaksi dengan yang sejalan dengan orang yang menderita asma," Dr E Rand dari divisi paru-paru dan perawatan medis kritis di NJH.

Peserta parah memiliki tingkat vitamin D paling rendah, mereka.

Asma berkaitan kegemukan, dan kekurangan vitamin D mungkin menjadi faktor yang menghubungkan kedua kondisi Sutherland.
"Memulihkan tingkat vitamin D orang menderita mungkin membantu asma mereka," kata Sutherland.
Namun, tidak diketahui asupan vitamin D membantu penderita katanya.
vitamin D buat orang dewasa yang disarankan ialah 300 IU sampai 600 IU, tergantung pada usia, kata US Institutes of Health.

- Langkah 1: *Pre-processing* Dokumen Asli dan Dokumen Uji

Dokumen Asli:

- *Tokenizing*

penderita asma yang kekurangan vitamin d akan mengalami pemburukan penyakit asma defisiensi vitamin ini diduga akan menghalangi reaksi terhadap pengobatan steroid yang sering dipakai dalam obat pelega dan pengontrol asma dalam risetnya para peneliti di national jewish health njh di denver mengukur tingkat vitamin d dari penderita asma dan menilai fungsi paru paru hyper responsiveness saluran udara yang lazim terjadi di dalam pengerutan saluran udara dan reaksi terhadap pengobatan steroid ternyata pasien yang mengalami defisiensi vitamin d memperlihatkan hasil yang buruk dalam pemeriksaan fungsi paru paru dan hyper responsiveness saluran udara lebih dari itu pasien yang kadar vitamin d dalam tubuhnya di bawah nanogram ml kadar hyper responsiveness saluran udara hampir dua kali lipat dibandingkan dengan mereka yang memiliki lebih banyak vitamin d di dalam darah mereka tingkat vitamin d yang rendah juga berkaitan dengan reaksi yang lebih buruk terhadap pengobatan steroid dan peningkatan produksi sitokin pro peradangan tnf alpha hal itu meningkatkan kemungkinan bahwa tingkat vitamin d yang rendah berkaitan dengan peningkatan peradangan saluran udara kata para peneliti itu selain itu kadar vitamin d meramalkan seberapa baik seseorang akan bereaksi terhadap pengobatan steroid bagi asma itu mungkin terjadi karena vitamin d bertindak sebagai pengubah reaksi steroid dengan cara yang sejalan dengan orang yang menderita asma kata dr e rand sutherland dari divisi paru paru dan perawatan medis kritis di njh peserta paling parah memiliki tingkat vitamin d

paling rendah kata mereka asma berkaitan dengan kegemukan dan kekurangan vitamin d mungkin menjadi faktor yang menghubungkan kedua kondisi tersebut kata sutherland memulihkan tingkat normal vitamin d pada orang yang menderita asma mungkin membantu meningkatkan asma mereka kata sutherland namun tidak diketahui apakah asupan vitamin d akan membantu penderita asma katanya asupan vitamin d buat orang dewasa yang disarankan ialah iu sampai iu tergantung pada usia kata us national institutes of health

- *Filtering*

penderita asma kekurangan vitamin d mengalami pemburukan penyakit asma defisiensi vitamin diduga menghalangi reaksi pengobatan steroid dipakai obat pelega pengontrol asma risetnya para peneliti national jewish health njh denver mengukur tingkat vitamin d penderita asma menilai fungsi paru paru hyper responsiveness saluran udara lazim terjadi pengerutan saluran udara reaksi pengobatan steroid ternyata pasien mengalami defisiensi vitamin d memperlihatkan hasil buruk pemeriksaan fungsi paru paru hyper responsiveness saluran udara lebih pasien kadar vitamin d tubuhnya bawah nanogram ml kadar hyper responsiveness saluran udara hampir dua lipat dibandingkan mereka memiliki lebih banyak vitamin d darah mereka tingkat vitamin d rendah juga berkaitan reaksi lebih buruk pengobatan steroid peningkatan produksi sitokin pro peradangan tnf alpha hal meningkatkan kemungkinan bahwa tingkat vitamin d rendah berkaitan peningkatan peradangan saluran udara kata para peneliti selain kadar vitamin d meramalkan seberapa baik seseorang bereaksi pengobatan steroid bagi asma mungkin terjadi vitamin d bertindak sebagai pengubah reaksi steroid cara sejalan orang menderita asma kata dr e rand sutherland divisi paru paru perawatan medis kritis njh peserta paling parah memiliki tingkat vitamin d paling rendah kata mereka asma berkaitan kegemukan kekurangan vitamin d mungkin menjadi faktor menghubungkan kedua kondisi tersebut kata sutherland memulihkan tingkat normal vitamin d orang menderita asma mungkin membantu meningkatkan asma mereka kata sutherland namun tidak diketahui apakah asupan vitamin d membantu penderita asma katanya asupan vitamin d buat orang dewasa disarankan ialah iu sampai iu tergantung usia kata us national institutes of health

- *Stemming*

derita asma kurang vitamin d alami buruk penyakit asma defisiensi vitamin duga halang reaksi obat steroid pakai obat lega control asma riset para teliti national jewish health njh denver ukur tingkat vitamin d derita asma nilai fungsi paru paru hyper responsiveness salur udara lazim jadi kerut salur udara reaksi obat steroid nyata pasien alami defisiensi vitamin d lihat hasil buruk periksa fungsi paru paru hyper responsiveness salur udara lebih pasien kadar vitamin d tubuh bawah nanogram ml kadar hyper responsiveness salur udara hampir dua lipat banding mereka milik lebih banyak vitamin d darah mereka tingkat vitamin d rendah juga kait reaksi lebih buruk obat steroid tingkat produksi sitok pro kadang tnf alpha hal tingkat mungkin bahwa tingkat vitamin d rendah kait tingkat kadang salur udara kata para teliti lain kadar vitamin d ramal apa baik orang reaksi obat steroid bagi asma mungkin jadi vitamin d tindak bagai ubah reaksi steroid cara jalan orang derita asma kata dr e rand sutherland divisi paru paru awat medis kritis njh peserta paling parah milik tingkat vitamin d paling rendah kata mereka asma kait gemuk kurang vitamin d mungkin jadi faktor hubung dua kondisi tersebut kata sutherland pulih tingkat normal vitamin d orang derita asma mungkin bantu tingkat asma mereka kata sutherland namun tidak tahu apa asup vitamin d bantu derita asma kata asup vitamin d buat orang dewasa saran ia iu sampai iu gantung usia kata us national institutes of health

Dokumen Uji:

- *Tokenizing*

penderita asma kekurangan vitamin d mengalami pemburukan penyakit defisiensi vitamin ini diduga akan menghalangi reaksi terhadap pengobatan steroid dipakai obat dan pengontrol asma dalam para peneliti di jewish health di denver tingkat vitamin d dari asma dan menilai fungsi paru paru hyper responsiveness saluran udara yang lazim terjadi di dalam saluran udara dan reaksi terhadap pengobatan ternyata pasien yang mengalami vitamin d yang buruk dalam pemeriksaan fungsi paru paru dan hyper responsiveness lebih itu pasien yang kadar vitamin d dalam tubuhnya di bawah nanogram ml hyper responsiveness hampir dua lipat dibandingkan dengan mereka yang memiliki

lebih banyak vitamin d di dalam darah mereka tingkat d yang rendah berkaitan dengan reaksi lebih buruk terhadap pengobatan dan sitokin pro peradangan tnf alpha hal itu meningkatkan kemungkinan bahwa tingkat d yang rendah berkaitan peningkatan peradangan saluran udara kata peneliti itu itu kadar d meramalkan seberapa baik seseorang akan bereaksi terhadap asma itu mungkin karena d bertindak sebagai pengubah reaksi dengan yang sejalan dengan orang yang menderita asma dr e rand dari divisi paru paru dan perawatan medis kritis di njh peserta parah memiliki tingkat vitamin d paling rendah mereka asma berkaitan kegemukan dan kekurangan vitamin d mungkin menjadi faktor yang menghubungkan kedua kondisi sutherland memulihkan tingkat vitamin d orang menderita mungkin membantu asma mereka kata sutherland namun tidak diketahui asupan vitamin d membantu penderita katanya vitamin d buat orang dewasa yang disarankan ialah iu sampai iu tergantung pada usia kata us institutes of health

- *Filtering*

penderita asma kekurangan vitamin d mengalami pemburukan penyakit defisiensi vitamin diduga menghalangi reaksi pengobatan steroid dipakai obat pengontrol asma para peneliti jewish health denver tingkat vitamin d asma menilai fungsi paru paru hyper responsiveness saluran udara lazim terjadi saluran udara reaksi pengobatan ternyata pasien mengalami vitamin d buruk pemeriksaan fungsi paru paru hyper responsiveness lebih pasien kadar vitamin d tubuhnya bawah nanogram ml hyper responsiveness hampir dua lipat dibandingkan mereka memiliki lebih banyak vitamin d darah mereka tingkat d rendah berkaitan reaksi lebih buruk pengobatan sitokin pro peradangan tnf alpha hal meningkatkan kemungkinan bahwa tingkat d rendah berkaitan peningkatan peradangan saluran udara kata peneliti kadar d meramalkan seberapa baik seseorang bereaksi asma mungkin d bertindak sebagai pengubah reaksi sejalan orang menderita asma dr e rand divisi paru paru perawatan medis kritis njh peserta parah memiliki tingkat vitamin d paling rendah mereka asma berkaitan kegemukan kekurangan vitamin d mungkin menjadi faktor menghubungkan kedua kondisi sutherland memulihkan tingkat vitamin d orang

menderita mungkin membantu asma mereka kata sutherland namun tidak diketahui asupan vitamin d membantu penderita katanya vitamin d buat orang dewasa disarankan ialah iu sampai iu tergantung usia kata us institutes of health

- *Stemming*

derita asma kurang vitamin d alami buruk penyakit defisiensi vitamin duga halang reaksi obat steroid pakai obat control asma para teliti jewish health denver tingkat vitamin d asma nilai fungsi paru paru hyper responsiveness salur udara lazim jadi salur udara reaksi obat nyata pasien alami vitamin d buruk periksa fungsi paru paru hyper responsiveness lebih pasien kadar vitamin d tubuh bawah nanogram ml hyper responsiveness hampir dua lipat banding mereka milik lebih banyak vitamin d darah mereka tingkat d rendah kait reaksi lebih buruk obat sitok pro kadang tnf alpha hal tingkat mungkin bahwa tingkat d rendah kait tingkat kadang salur udara kata teliti kadar d ramal apa baik orang reaksi asma mungkin d tindak bagai ubah reaksi jalan orang derita asma dr e rand divisi paru paru awat medis kritis njh peserta parah milik tingkat vitamin d paling rendah mereka asma kait gemuk kurang vitamin d mungkin jadi faktor hubung dua kondisi sutherland pulih tingkat vitamin d orang derita mungkin bantu asma mereka kata sutherland namun tidak tahu asup vitamin d bantu derita kata vitamin d buat orang dewasa saran ia iu sampai iu gantung usia kata us institutes of health

- Langkah 2: Melakukan *synonym recognition* pada dokumen asli dan dokumen uji

Dokumen asli:

derita asma kurang vitamin d alami buruk penyakit asma defisiensi vitamin duga halang reaksi obat steroid pakai obat lega control asma riset para teliti national jewish health njh denver ukur tingkat vitamin d derita asma nilai **tujuan** paru paru hyper responsiveness salur udara lazim jadi erut salur udara reaksi obat steroid nyata pasien alami defisiensi vitamin d lihat hasil buruk periksa **tujuan** paru paru hyper responsiveness salur udara lebih pasien kadar vitamin d tubuh bawah nanogram ml kadar hyper responsiveness salur udara hampir dua lipat banding mereka milik lebih banyak vitamin d darah mereka

tingkat vitamin d rendah juga kait reaksi lebih buruk obat steroid tingkat produksi sitok pro kadang tnf alpha hal tingkat mungkin bahwa tingkat vitamin d rendah kait tingkat kadang salur udara kata para teliti lain kadar vitamin d **prediksi** apa baik orang reaksi obat steroid bagi asma mungkin jadi vitamin d tindak seperti ubah reaksi steroid cara jalan orang derita asma kata dr e rand sutherland divisi paru paru rawat medis kritis njh peserta paling parah milik tingkat vitamin d paling rendah kata mereka asma kait gemuk kurang vitamin d mungkin jadi faktor hubung dua kondisi tersebut kata sutherland sehat tingkat normal vitamin d orang derita asma mungkin **tolong** tingkat asma mereka kata sutherland namun tidak tahu apa asup vitamin d **tolong** derita asma kata asup vitamin d buat orang dewasa saran ia iu sampai iu gantung **umur** kata us national institutes of health

Dokumen uji:

derita asma kurang vitamin d alami buruk penyakit defisiensi vitamin duga halang reaksi obat steroid pakai obat control asma para teliti jewish health denver tingkat vitamin d asma nilai **tujuan** paru paru hyper responsiveness salur udara lazim jadi salur udara reaksi obat nyata pasien alami vitamin d buruk periksa **tujuan** paru paru hyper responsiveness lebih pasien kadar vitamin d tubuh bawah nanogram ml hyper responsiveness hampir dua lipat banding mereka milik lebih banyak vitamin d darah mereka tingkat d rendah kait reaksi lebih buruk obat sitok pro kadang tnf alpha hal tingkat mungkin bahwa tingkat d rendah kait tingkat kadang salur udara kata teliti kadar d **prediksi** apa baik orang reaksi asma mungkin d tindak seperti ubah reaksi jalan orang derita asma dr e rand divisi paru paru rawat medis kritis njh peserta parah milik tingkat vitamin d paling rendah mereka asma kait gemuk kurang vitamin d mungkin jadi faktor hubung dua kondisi sutherland sehat tingkat vitamin d orang derita mungkin **tolong** asma mereka kata sutherland namun tidak tahui asup vitamin d **tolong** derita kata vitamin d buat orang dewasa saran ia iu sampai iu gantung umur kata us institutes of health

- Langkah 3: Perhitungan *similarity* dengan VSM
  - List kata pada dokumen beserta TF.IDF

NO.	KATA	TF		W_TF	
		ASLI	UJI	ASLI	UJI
1	derita	5	4	1,69897	1,60206
2	asma	10	7	2	1,845098
3	kurang	2	2	1,30103	1,30103
4	vitamin	15	11	2,176091	2,041393
5	d	14	14	2,146128	2,146128
6	alami	2	2	1,30103	1,30103
7	buruk	3	3	1,477121	1,477121
8	penyakit	1	1	1	1
9	defisiensi	2	1	1,30103	1
10	duga	1	1	1	1
11	cegah	1	1	1	1
12	reaksi	5	5	1,69897	1,69897
13	obat	5	4	1,69897	1,60206
14	steroid	5	1	1,69897	1
15	pakai	1	1	1	1
16	lega	1	0	1	0
17	control	1	1	1	1
18	riset	1	0	1	0
19	para	2	1	1,30103	1
20	teliti	2	2	1,30103	1,30103
21	national	2	0	1,30103	0
22	jewish	1	1	1	1
23	health	2	2	1,30103	1,30103
24	njh	2	1	1,30103	1
25	denver	1	1	1	1
26	ukur	1	0	1	0
27	tingkat	9	7	1,954243	1,845098
28	nilai	1	1	1	1
29	tujuan	2	2	1,30103	1,30103
30	paru	6	6	1,778151	1,778151
31	hyper	3	3	1,477121	1,477121
32	responsiveness	3	3	1,477121	1,477121
33	salur	5	3	1,69897	1,477121
34	udara	5	3	1,69897	1,477121
35	lazim	1	1	1	1
36	jadi	3	2	1,477121	1,30103
37	kerut	1	0	1	0
38	nyata	1	1	1	1
39	pasien	2	2	1,30103	1,30103
40	lihat	1	0	1	0

NO.	KATA	TF		W_TF	
		ASLI	UJI	ASLI	UJI
41	hasil	1	0	1	0
42	periksa	1	1	1	1
43	lebih	3	3	1,477121	1,477121
44	kadar	3	2	1,477121	1,30103
45	tubuh	1	1	1	1
46	bawah	1	1	1	1
47	nanogram	1	1	1	1
48	ml	1	1	1	1
49	hampir	1	1	1	1
50	dua	2	2	1,30103	1,30103
51	lipat	1	1	1	1
52	banding	1	1	1	1
53	mereka	4	4	1,60206	1,60206
54	milik	2	2	1,30103	1,30103
55	banyak	1	1	1	1
56	darah	1	1	1	1
57	rendah	3	3	1,477121	1,477121
58	juga	1	0	1	0
59	kait	3	3	1,477121	1,477121
60	produksi	1	0	1	0
61	sitok	1	1	1	1
62	pro	1	1	1	1
63	kadang	2	2	1,30103	1,30103
64	tnf	1	1	1	1
65	alpha	1	1	1	1
66	hal	1	1	1	1
67	mungkin	4	4	1,60206	1,60206
68	bahwa	1	1	1	1
69	kata	7	4	1,845098	1,60206
70	lain	1	0	1	0
71	prediksi	1	1	1	1
72	apa	2	1	1,30103	1
73	baik	1	1	1	1
74	orang	4	4	1,60206	1,60206
75	bagi	1	0	1	0
76	tindak	1	1	1	1
77	seperti	1	1	1	1
78	ubah	1	1	1	1
79	cara	1	0	1	0
80	jalan	1	1	1	1

NO.	KATA	TF		W_TF	
		ASLI	UJI	ASLI	UJI
81	dr	1	1	1	1
82	e	1	1	1	1
83	rand	1	1	1	1
84	sutherland	3	2	1,477121	1,30103
85	visi	1	1	1	1
86	awat	1	1	1	1
87	medis	1	1	1	1
88	kritis	1	1	1	1
89	peserta	1	1	1	1
90	paling	2	1	1,30103	1
91	parah	1	1	1	1
92	gemuk	1	1	1	1
93	faktor	1	1	1	1
94	hubung	1	1	1	1
95	konmedisi	1	1	1	1
96	tersebut	1	0	1	0
97	sehat	1	1	1	1
98	normal	1	0	1	0
99	tolong	2	2	1,30103	1,30103
100	namun	1	1	1	1
101	tidak	1	1	1	1
102	tahu	1	1	1	1
103	asup	2	1	1,30103	1
104	buat	1	1	1	1
105	dewasa	1	1	1	1
106	saran	1	1	1	1
107	ia	1	1	1	1
108	iu	2	2	1,30103	1,30103
109	sampai	1	1	1	1
110	gantung	1	1	1	1
111	umur	1	1	1	1
112	us	1	1	1	1
113	institutes	1	1	1	1
114	of	1	1	1	1

Untuk perhitungan W_{tf}, menggunakan rumus:

$$W_{tf} \begin{cases} 1 + \log TF, if\ TF > 0 \\ 0 \qquad\qquad , lainnya \end{cases}$$

Misalnya: kata siswa memiliki *Term Frequency* pada dokumen asli sebanyak 5 dan pada dokumen uji sebanyak 4.

Maka $W_{tf} dokumen\ asli = 1 + \log 5 = 1 + 0{,}6989 = 1{,}6989$

Dan $W_{tf} dokumen\ uji = 1 + \log 4 = 1 + 0{,}6020 = 1{,}6020$

- Lakukan perkalian $W_{tf} dokumen\ asli$ dan $W_{tf} dokumen\ uji$ lalu jumlahkan semua hasil perkaliannya

NO.	KATA	W_TF		W_TF_ASLI x W_TF_UJI
		ASLI	UJI	
1	derita	1,69897	1,60206	2,72185187
2	asma	2	1,845098	3,69019608
3	kurang	1,30103	1,30103	1,69267905
4	vitamin	2,176091	2,041393	4,442256778
5	d	2,146128	2,146128	4,605865546
6	alami	1,30103	1,30103	1,69267905
7	buruk	1,477121	1,477121	2,181887201
8	penyakit	1	1	1
9	defisiensi	1,30103	1	1,301029996
10	duga	1	1	1
11	cegah	1	1	1
12	reaksi	1,69897	1,69897	2,886499076
13	obat	1,69897	1,60206	2,72185187
14	steroid	1,69897	1	1,698970004
15	pakai	1	1	1
16	lega	1	0	0
17	control	1	1	1
18	riset	1	0	0
19	para	1,30103	1	1,301029996
20	teliti	1,30103	1,30103	1,69267905
21	national	1,30103	0	0
22	jewish	1	1	1
23	health	1,30103	1,30103	1,69267905
24	njh	1,30103	1	1,301029996
25	denver	1	1	1
26	ukur	1	0	0
27	tingkat	1,954243	1,845098	3,605769024
28	nilai	1	1	1
29	tujuan	1,30103	1,30103	1,69267905
30	paru	1,778151	1,778151	3,161821869
31	hyper	1,477121	1,477121	2,181887201

NO.	KATA	W_TF		W_TF_ASLI x W_TF_UJI
		ASLI	UJI	
32	responsiveness	1,477121	1,477121	2,181887201
33	salur	1,69897	1,477121	2,509584705
34	udara	1,69897	1,477121	2,509584705
35	lazim	1	1	1
36	jadi	1,477121	1,30103	1,92177906
37	kerut	1	0	0
38	nyata	1	1	1
39	pasien	1,30103	1,30103	1,69267905
40	lihat	1	0	0
41	hasil	1	0	0
42	periksa	1	1	1
43	lebih	1,477121	1,477121	2,181887201
44	kadar	1,477121	1,30103	1,92177906
45	tubuh	1	1	1
46	bawah	1	1	1
47	nanogram	1	1	1
48	ml	1	1	1
49	hampir	1	1	1
50	dua	1,30103	1,30103	1,69267905
51	lipat	1	1	1
52	banding	1	1	1
53	mereka	1,60206	1,60206	2,566596216
54	milik	1,30103	1,30103	1,69267905
55	banyak	1	1	1
56	darah	1	1	1
57	rendah	1,477121	1,477121	2,181887201
58	juga	1	0	0
59	kait	1,477121	1,477121	2,181887201
60	produksi	1	0	0
61	sitok	1	1	1
62	pro	1	1	1
63	kadang	1,30103	1,30103	1,69267905
64	tnf	1	1	1
65	alpha	1	1	1
66	hal	1	1	1
67	mungkin	1,60206	1,60206	2,566596216
68	bahwa	1	1	1
69	kata	1,845098	1,60206	2,95595775
70	lain	1	0	0
71	prediksi	1	1	1

NO.	KATA	W_TF		W_TF_ASLI x W_TF_UJI
		ASLI	UJI	
72	apa	1,30103	1	1,301029996
73	baik	1	1	1
74	orang	1,60206	1,60206	2,566596216
75	bagi	1	0	0
76	tindak	1	1	1
77	seperti	1	1	1
78	ubah	1	1	1
79	cara	1	0	0
80	jalan	1	1	1
81	dr	1	1	1
82	e	1	1	1
83	rand	1	1	1
84	sutherland	1,477121	1,30103	1,92177906
85	visi	1	1	1
86	awat	1	1	1
87	medis	1	1	1
88	kritis	1	1	1
89	peserta	1	1	1
90	paling	1,30103	1	1,301029996
91	parah	1	1	1
92	gemuk	1	1	1
93	faktor	1	1	1
94	hubung	1	1	1
95	konmedisi	1	1	1
96	tersebut	1	0	0
97	sehat	1	1	1
98	normal	1	0	0
99	tolong	1,30103	1,30103	1,69267905
100	namun	1	1	1
101	tidak	1	1	1
102	tahu	1	1	1
103	asup	1,30103	1	1,301029996
104	buat	1	1	1
105	dewasa	1	1	1
106	saran	1	1	1
107	ia	1	1	1
108	iu	1,30103	1,30103	1,69267905
109	sampai	1	1	1
110	gantung	1	1	1
111	umur	1	1	1

NO.	KATA	W_TF		W_TF_ASLI x W_TF_UJI
		ASLI	UJI	
112	us	1	1	1
113	institutes	1	1	1
114	of	1	1	1
JUMLAH				149,4923078

- Kuadratkan $W_{tf} dokumen\ asli$ dan W_{tf} lalu jumlahkan masing-masing hasil kuadrat $W_{tf} dokumen\ asli$ dan W_{tf}, setelah itu kalikan kedua hasil penjumlahan tersebut dan dilakukan akar kuadrat

NO.	KATA	W_TF		$(W\_TF)^2$	
		ASLI	UJI	ASLI	UJI
1	derita	1,69897	1,60206	2,886499	2,566596
2	asma	2	1,845098	4	3,404387
3	kurang	1,30103	1,30103	1,692679	1,692679
4	vitamin	2,176091	2,041393	4,735373	4,167284
5	d	2,146128	2,146128	4,605866	4,605866
6	alami	1,30103	1,30103	1,692679	1,692679
7	buruk	1,477121	1,477121	2,181887	2,181887
8	penyakit	1	1	1	1
9	defisiensi	1,30103	1	1,692679	1
10	duga	1	1	1	1
11	cegah	1	1	1	1
12	reaksi	1,69897	1,69897	2,886499	2,886499
13	obat	1,69897	1,60206	2,886499	2,566596
14	steroid	1,69897	1	2,886499	1
15	pakai	1	1	1	1
16	lega	1	0	1	0
17	control	1	1	1	1
18	riset	1	0	1	0
19	para	1,30103	1	1,692679	1
20	teliti	1,30103	1,30103	1,692679	1,692679
21	national	1,30103	0	1,692679	0
22	jewish	1	1	1	1
23	health	1,30103	1,30103	1,692679	1,692679
24	njh	1,30103	1	1,692679	1
25	denver	1	1	1	1
26	ukur	1	0	1	0
27	tingkat	1,954243	1,845098	3,819064	3,404387
28	nilai	1	1	1	1
29	tujuan	1,30103	1,30103	1,692679	1,692679
30	paru	1,778151	1,778151	3,161822	3,161822

NO.	KATA	W_TF		$(W\_TF)^2$	
		ASLI	UJI	ASLI	UJI
32	responsiveness	1,477121	1,477121	2,181887	2,181887
33	salur	1,69897	1,477121	2,886499	2,181887
34	udara	1,69897	1,477121	2,886499	2,181887
35	lazim	1	1	1	1
36	jadi	1,477121	1,30103	2,181887	1,692679
37	kerut	1	0	1	0
38	nyata	1	1	1	1
39	pasien	1,30103	1,30103	1,692679	1,692679
40	lihat	1	0	1	0
41	hasil	1	0	1	0
42	periksa	1	1	1	1
43	lebih	1,477121	1,477121	2,181887	2,181887
44	kadar	1,477121	1,30103	2,181887	1,692679
45	tubuh	1	1	1	1
46	bawah	1	1	1	1
47	nanogram	1	1	1	1
48	ml	1	1	1	1
49	hampir	1	1	1	1
50	dua	1,30103	1,30103	1,692679	1,692679
51	lipat	1	1	1	1
52	banding	1	1	1	1
53	mereka	1,60206	1,60206	2,566596	2,566596
54	milik	1,30103	1,30103	1,692679	1,692679
55	banyak	1	1	1	1
56	darah	1	1	1	1
57	rendah	1,477121	1,477121	2,181887	2,181887
58	juga	1	0	1	0
59	kait	1,477121	1,477121	2,181887	2,181887
60	produksi	1	0	1	0
61	sitok	1	1	1	1
62	pro	1	1	1	1
63	kadang	1,30103	1,30103	1,692679	1,692679
64	tnf	1	1	1	1
65	alpha	1	1	1	1
66	hal	1	1	1	1
67	mungkin	1,60206	1,60206	2,566596	2,566596
68	bahwa	1	1	1	1
69	kata	1,845098	1,60206	3,404387	2,566596
70	lain	1	0	1	0
71	prediksi	1	1	1	1

NO.	KATA	W_TF		$(W\_TF)^2$	
		ASLI	UJI	ASLI	UJI
72	apa	1,30103	1	1,692679	1
73	baik	1	1	1	1
74	orang	1,60206	1,60206	2,566596	2,566596
75	bagi	1	0	1	0
76	tindak	1	1	1	1
77	seperti	1	1	1	1
78	ubah	1	1	1	1
79	cara	1	0	1	0
80	jalan	1	1	1	1
81	dr	1	1	1	1
82	e	1	1	1	1
83	rand	1	1	1	1
84	sutherland	1,477121	1,30103	2,181887	1,692679
85	visi	1	1	1	1
86	awat	1	1	1	1
87	medis	1	1	1	1
88	kritis	1	1	1	1
89	peserta	1	1	1	1
90	paling	1,30103	1	1,692679	1
91	parah	1	1	1	1
92	gemuk	1	1	1	1
93	faktor	1	1	1	1
94	hubung	1	1	1	1
95	konmedisi	1	1	1	1
96	tersebut	1	0	1	0
97	sehat	1	1	1	1
98	normal	1	0	1	0
99	tolong	1,30103	1,30103	1,692679	1,692679
100	namun	1	1	1	1
101	tidak	1	1	1	1
102	tahu	1	1	1	1
103	asup	1,30103	1	1,692679	1
104	buat	1	1	1	1
105	dewasa	1	1	1	1
106	saran	1	1	1	1
107	ia	1	1	1	1
108	iu	1,30103	1,30103	1,692679	1,692679
109	sampai	1	1	1	1
110	gantung	1	1	1	1

NO.	KATA	W_TF		(W_TF)^2	
		ASLI	UJI	ASLI	UJI
111	umur	1	1	1	1
112	us	1	1	1	1
113	institutes	1	1	1	1
114	of	1	1	1	1
JUMLAH				170,8505	144,1824
SQRT(JML W_TF_ASLI * JML W_TF_UJI)				156,9510746	

- Langkah terakhir adalah melakukan pembagian antara jumlah perkalian $W_{tf}\,dokumen\ asli$ dan $W_{tf}\ dokumen\ uji$ dengan akar kuadrat dari jumlah perkalian $W_{tf}\,dokumen\ asli$ kuadrat dan $W_{tf}\ dokumen\ uji$ kuadrat. Dapat dituliskan dengan persamaan:

$$D(D_{asli}, D_{uji}) = \frac{\sum(w_{asli,j}\ x\ w_{uji,j})}{\sqrt{\sum w_{asli,j}^2\ x\ \sum w_{uji,j}^2}}$$

$$D(D_{asli}, D_{uji}) = \frac{149{,}4923}{156{,}9510}$$

$$D(D_{asli}, D_{uji}) = 0{,}9525$$

- Langkah 4: Perhitungan *similarity* dengan Jaccard Coefisien
  - Langkah pertama yaitu menentukan kata yang sama terdapat pada dokumen asli dan dokumen uji, setelah itu dicari TF kata yang paling kecil antara dokumen asli dan dokumen uji
  - Langkah selanjutnya melakukan penjumlahan total semua kata pada dokumen asli dan dokumen uji

NO.	KATA	TF		MIN
		ASLI	UJI	
1	derita	5	4	4
2	asma	10	7	7
3	kurang	2	2	2
4	vitamin	15	11	11
5	d	14	14	14
6	alami	2	2	2
7	buruk	3	3	3
8	penyakit	1	1	1
9	defisiensi	2	1	1
10	duga	1	1	1

NO.	KATA	TF		MIN
		ASLI	UJI	
11	cegah	1	1	1
12	reaksi	5	5	5
13	obat	5	4	4
14	steroid	5	1	1
15	pakai	1	1	1
16	lega	1	0	0
17	control	1	1	1
18	riset	1	0	0
19	para	2	1	1
20	teliti	2	2	2
21	national	2	0	0
22	jewish	1	1	1
23	health	2	2	2
24	njh	2	1	1
25	denver	1	1	1
26	ukur	1	0	0
27	tingkat	9	7	7
28	nilai	1	1	1
29	tujuan	2	2	2
30	paru	6	6	6
31	hyper	3	3	3
32	responsiveness	3	3	3
33	salur	5	3	3
34	udara	5	3	3
35	lazim	1	1	1
36	jadi	3	2	2
37	kerut	1	0	0
38	nyata	1	1	1
39	pasien	2	2	2
40	lihat	1	0	0
41	hasil	1	0	0
42	periksa	1	1	1
43	lebih	3	3	3
44	kadar	3	2	2
45	tubuh	1	1	1
46	bawah	1	1	1
47	nanogram	1	1	1
48	ml	1	1	1
49	hampir	1	1	1
50	dua	2	2	2
51	lipat	1	1	1

NO.	KATA	TF		MIN
		ASLI	UJI	
52	banding	1	1	1
53	mereka	4	4	4
54	milik	2	2	2
55	banyak	1	1	1
56	darah	1	1	1
57	rendah	3	3	3
58	juga	1	0	0
59	kait	3	3	3
60	produksi	1	0	0
61	sitok	1	1	1
62	pro	1	1	1
63	kadang	2	2	2
64	tnf	1	1	1
65	alpha	1	1	1
66	hal	1	1	1
67	mungkin	4	4	4
68	bahwa	1	1	1
69	kata	7	4	4
70	lain	1	0	0
71	prediksi	1	1	1
72	apa	2	1	1
73	baik	1	1	1
74	orang	4	4	4
75	bagi	1	0	0
76	tindak	1	1	1
77	seperti	1	1	1
78	ubah	1	1	1
79	cara	1	0	0
80	jalan	1	1	1
81	dr	1	1	1
82	e	1	1	1
83	rand	1	1	1
84	sutherland	3	2	2
85	visi	1	1	1
86	awat	1	1	1
87	medis	1	1	1
88	kritis	1	1	1
89	peserta	1	1	1
90	paling	2	1	1
91	parah	1	1	1
92	gemuk	1	1	1

NO.	KATA	TF		MIN
		ASLI	UJI	
93	faktor	1	1	1
94	hubung	1	1	1
95	konmedisi	1	1	1
96	tersebut	1	0	0
97	sehat	1	1	1
98	normal	1	0	0
99	tolong	2	2	2
100	namun	1	1	1
101	tidak	1	1	1
102	tahu	1	1	1
103	asup	2	1	1
104	buat	1	1	1
105	dewasa	1	1	1
106	saran	1	1	1
107	ia	1	1	1
108	iu	2	2	2
109	sampai	1	1	1
110	gantung	1	1	1
111	umur	1	1	1
112	us	1	1	1
113	institutes	1	1	1
114	of	1	1	1
JUMLAH		238	192	192

- Untuk mendapatkan *similarity* dengan metode Jaccard Coefisien digunakan rumus:

$$J(D_{asli}, D_{uji}) = \frac{2\ (\sum \min(D_{asli}, D_{uji})}{D_{asli} + D_{uji}}$$

$$J(D_{asli}, D_{uji}) = \frac{2\ (192)}{238 + 192} = \frac{384}{430} = 0{,}893$$

- Langkah 5: Melakukan kombinasi antar algoritma VSM dan Jaccard Coefisien menggunakan persamaan

$$Sim(D_a, D_b) = \frac{(D(D_a, D_b) - \alpha)(J(D_a, D_b) - \beta)}{K}$$

Misalnya nilai α yang digunakan adalah 0,1 dan nilai β adalah 0,1 serta nilai K adalah 0,01

Sehingga diperoleh hasil:

$$Sim(D_a, D_b) = \frac{(0{,}9525 - 0{,}1)(0{,}893 - 0{,}1)}{0{,}01}$$

$$Sim(D_a, D_b) = 67{,}6032$$

Setelah itu dilakukan normalisasi *similarity*, pada penelitian ini normalisasi yang digunakan adalah normalisasi maks-min dengan formula:

$$Normalisasi\ Sim(D_a, D_b) = \frac{Sim(D_a, D_b) - (min)}{(maks) - (min)}$$

Nilai maks didapatkan dari memasukkan angka maksimal *similarity* yang terbesar pada *similarity* VSM dan *similarity* Jaccard yaitu angka 1, sehingga nilai maks menjadi:

$$maks = \frac{(1 - 0{,}1)(1 - 0{,}1)}{0{,}01} = 81$$

Setelah itu masukkan angka *similarity*, maksimal *similarity* (maks) dan minimum *similarity* (min) dalam persamaan normalisasi, sehingga menjadi:

$$Normalisasi\ Sim(D_a, D_b) = \frac{Sim(D_a, D_b) - 0}{81 - 0}$$

$$Normalisasi\ Sim(D_a, D_b) = \frac{67{,}6032 - 0}{81 - 0}$$

$$Normalisasi\ Sim(D_a, D_b) = 0{,}8346$$

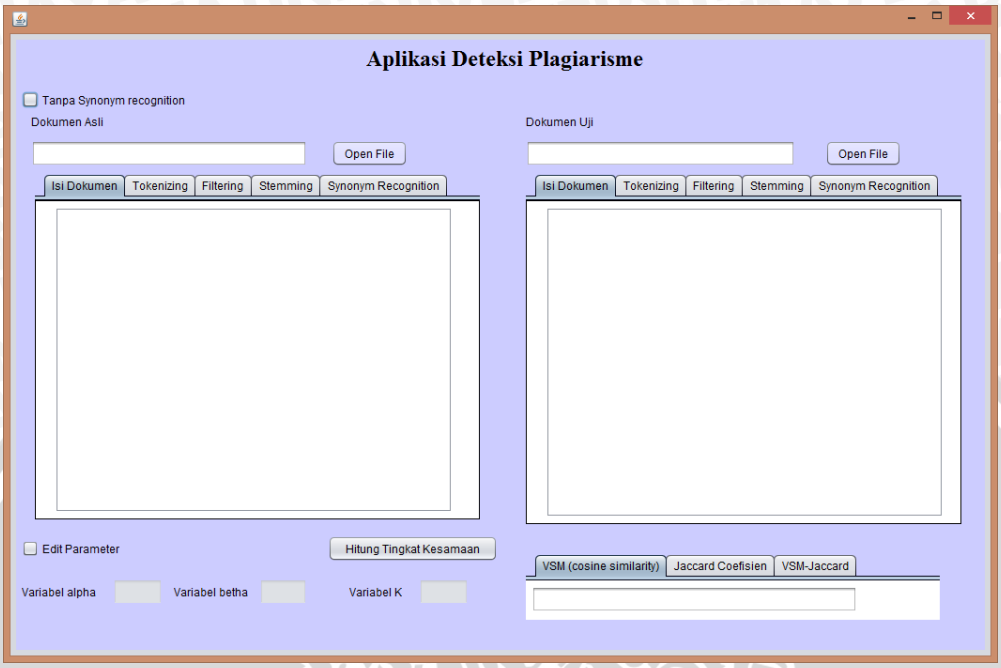
Jadi *similarity* yang diperoleh dari dokumen asli dan dokumen uji sebesar 83,46%

3.4.3 Perancangan Antarmuka

Pada tahap ini dilakukan perancangan antarmuka untuk pengimplementasian sistem. Aplikasi dibuat dengan menggunakan bahasa pemrograman Java. Pemroresan pada sistem dibuat untuk menghasilkan hasil akhir berupa *similarity* dokumen menggunakan kombinasi VSM dan Jaccard Coefisien.

Pada perancangan antarmuka aplikasi ini terdapat 2 tombol untuk memilih file yang akan diuji tingkat kesamaannya, serta 1 tombol untuk menghitung nilai kesamaan antara dokumen asli dan uji. Terdapat tab *text area* untuk menampikan isi dokumen, hasil *pre-processing* dan hasil *synonym recognition* dokumen yang akan diuji. Sedangkan tab *text area* hasil perhitungan kesamaan dibagi menjadi 3

yaitu hasil *similarity* menggunakan VSM (*cosine similarity*), Jaccard Coefisien dan kombinasi VSM-Jaccard.



Gambar 3.10 Perancangan Antarmuka Aplikasi Deteksi Plagiarisme
Sumber: [Perancangan]

3.5 Pengujian dan Analisa

Tahap ini bertujuan untuk memastikan bahwa aplikasi yang dibuat telah sesuai standar dan telah berjalan dengan baik. Pada tahap ini terdapat dua sub bab yaitu bahan pengujian dan skenario pengujian.

3.5.1 Bahan Pengujian

Pengujian yang dilakukan adalah pengujian terhadap variabel α, β dan K pada formula kombinasi algoritma VSM dan Jaccard Coefisien untuk mendapatkan hasil *similarity* terbaik. Selain itu pengaruh *synonym recognition* juga dilakukan pengujian untuk mengetahui seberapa besar pengaruhnya untuk deteksi plagiarisme. Sedangkan untuk memastikan keakuratan *similarity* yang dihasilkan oleh sistem, pengujian dilakukan dengan merubah dokumen asli dengan berbagai macam prosentasi perubahan antara 20%-80%. Setelah itu dibandingkan hasilnya apakah hasil *similarity* yang didapat sesuai dengan perubahan yang telah ditentukan. Proses perubahan dokumen asli bermacam-

macam antara lain pemotongan kata, penukaran susunan kalimmat dan kesalahan penulisan.

Selain itu pengujian juga meliputi pencocokan hasil *similarity* aplikasi dengan perhitungan manual algoritma VSM dan Jaccard. Pada pengujian ini dapat ditentukan apakah hasil *similarity* yang didapat dengan kombinasi algoritma VSM dan Jaccard pada aplikasi sesuai dengan perhitungan manual.

Selain pengujian basis pengetahuan dan algoritma, juga dilakukan pengujian sistem. Pengujian sistem dilakukan dengan memeriksa apakah sistem sudah dapat berjalan dengan baik dan tidak ada *error*.

3.5.2 Skenario Pengujian

Skenario pengujian yang dilakukan terdapat tiga macam, yaitu pengujian variabel α, β dan K, pengujian pengaruh *synonym recognition* dan pengujian perbandingan algoritma.

a. Pengujian variabel α, β dan K

Pengujian ini dilakukan untuk menemukan nilai α, β dan K yang optimal dalam mendeteksi plagiarisme. Ketiga variabel tersebut dipergunakan untuk membangun formula kombinasi VSM dan Jaccard Coefisien. Nilai yang akan diuji adalah sebagai berikut:

1. Variabel α : bilangan bulat kecil antara 0,1 – 0,9
2. Variabel β : bilangan bulat kecil antara 0,1 – 0,9
3. Variabel K : bilangan bulat kecil antara 0,01-0.09

Tabel 3.1 Perancangan Pengujian Variable α, β dan K

Dokumen asli	**Dokumen uji**	**Variabel α**	**Variabel β**	**Variabel K**	***Similarity* (%)**	***Error* (%)**
		uji coba	konstan	konstan		
		uji coba	konstan	konstan		
		Konstan	**uji coba**	konstan		
		Konstan	**uji coba**	konstan		
		Konstan	konstan	**uji coba**		
		Konstan	konstan	**uji coba**		

b. Pengujian pengaruh *synonym recognition*

Pengujian ini dilakukan untuk mengetahui sebera besar pengaruh penambahan *synonym recognition* pada deteksi plagiarisme. *Synonym recognition* digunakan untuk mendeteksi plagiarisme pada dokumen yang telah diubah menggunakan sinonimnya.

Tabel 3.2 Perancangan Pengujian Pengaruh *Synonym Recognition*

Dokumen asli	**Dokumen uji**	***Similarity*(%)**		***Error* (%)**	
		Menggunakan *Synonym recognition*	**Tanpa *Synonym recognition***	**Menggunakan *Synonym recognition***	**Tanpa *Synonym recognition***
Rata-rata *error*					

c. Pengujian *similarity* dokumen

Pengujian ini dilakukan untuk mengetahui tingkat keberhasilan sistem deteksi plagiarisme. Parameter-parameter yang digunakan (variabel α, β dan K) diberikan nilai terbaik yang didapatkan pada skenario uji coba 1. Selain itu digunakan pula *synonym recognition* jika telah terbukti memiliki pengaruh yang signifikan terhadap deteksi plagiarisme.

Tabel 3.3 Perancangan Pengujian Algoritma Deteksi Plagiarisme

Dokumen Asli	**Dokumen Uji**	***Similarity* (%)**			***Error* (%)**		
		VSM	**Jaccard Coefisien**	**VSM-Jaccard**	**VSM**	**Jaccard Coefisien**	**VSM-Jaccard**
Rata-rata *error*							

3.6 Pengambilan Kesimpulan

Pengambilan kesimpulan dilakukan setelah semua tahap perancangan, implementasi dan pengujian sistem]telah selesai dilakukan. Kesimpulan diambil dari hasil pengujian dan analisis yang dibangun. Tahap terakhir dari penulisan adalah saran yang dimaksudkan untuk memeperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan aplikasi selanjutnya.

BAB IV
IMPLEMENTASI

Berdasarkan perancangan sistem yang telah dibuat pada bab sebelumnya, bab ini menjabarkan implementasi dari sistem yang mencakup lingkungan perangkat kerasa dan perangkat lunak.

4.1 Lingkungan Perangkat Keras

Dalam mengembangkan dan mengimplementasikan sistem deteksi plagiarisme digunakan perangkat kerasa dengan spesifikasi sebagai berikut:

1. Processor Intel Core i5
2. *Random Access Memory* (RAM) 4 GB

4.2 Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan dalam mengembangkan sistem ini adalah:

1. Sistem operasi Windows 8.0 sebagai sistem operasi yang dijalankan saat membangun aplikasi.
2. Netbeans IDE 8.0 sebagai media untuk mebangun aplikaasi dengan bahasa pemograman Java.
3. Notepad sebagai media untuk membuat dan merubah isi dokumen yang digunakan untuk pengujian.

4.3 Implementasi Antarmuka

Untuk memudahkan dan dipahami user, tampilan antarmuka dirancang secara sederhana. Berdasarkan perancangan antarmuka yang telah dibuat pada bab sebelumnya, antarmuka aplikasi hanya memiliki satu tampilan utama. Judul aplikasi terletak ditengah atas. Terdapat dua kolom *tabbed panel* dengan masing-masing lima tab antara lain isi dokumen, hasil *tokenizing, filtering, stemming* dan hasil *synonym recognition*. Tab isi dokumen digunakan untuk menampilkan isi dokumen yang telah dipilih. Tab hasil *tokenizing* digunakan untuk menampilkan hasil *tokenizing* pada dokumen yang telah dipilih yaitu berupa penghilangan tanda baca dan *lowercase*. Tab hasil *filtering* digunakan untuk menempilkan hasil

filtering berupa penghilangan kata hubung dan kata tidak penting. Tab hasil *stemming* digunakan untuk menampilkan hasil *stemming* berupa kata dasar pada dokumen yang dipilih. Tab hasil *synonym recognition* digunakan untuk menampilkan hasil perubahan kata sinonim pada dokumen yang telah dipilih.

Antarmuka didesain agar user cukup melakukan 4 langkah utama untuk menjalankan program. Untuk menjalankan aplikasi terlebih dahulu user dipersilahkan memilih menggunkan *synonym recognition* atau tidak, jika ingin menggunakannya user tidak perlu mencentang *checbox* "tanpa *synonym recognition*". Setelah itu user menginputkan dokumen asli dan dokumen uji. Lalu user dapat melakukan perubahan parameter α, β dan k dengan mencentang checbox "edit parameter" jika tidak user melakukan perhitungan dengan nilai parameter yang disediakan sistem. Langkah terakhir yaitu user cukup menekan tombol agar sistem dapat menghitung *similarity* antara dokumen asli dan dokumen uji dan menempilkan *similarity* dengan metode VSM, Jaccard dan kombinasi keduanya.

Untuk memproses dua dokumen yang akan diuji tingkat kedekatnnya terdapat 2 tombol inputan. Tombol pertama digunakan untuk menginputkan dokumen asli sedangkan tombol lainnya digunakan untuk menginputkan dokumen uji. Setelah itu user dapat melihat isi dokumen, melihat hasil *tokenizing, filtering, stemming* dan melihat hasil *synonym recognition* pada masing-masing tab yang disediakan. Selain itu terdapat 2 *checbox* untuk memberi kebebasan pada user dalam penggunaan *synonym recognition* dan perubahan parameter. Terdapat pula tombol untuk melakukan perhitungan *similarity* setelah itu user dapat melihat hasil *similarity* kedua dokumen.

4.4 Implementasi Proses

Pada bagian ini akan dijabarkan implementasi dari proses sistem yang sudah dijelaskan dalam bab sebelumnya dengan menyertakan *listing code* program.

4.4.1 Implementasi Proses Input Dokumen dan Menampilkan File

Pada saat user menginputkan dokumen, user memasukkan file dokumen teks dan sistem mengolah dokumen berformat txt. Prosesnya dijelaskan pada *Source code* 4.1

```
File file = jFileChooser1.getSelectedFile();
            jTextField3.setText(file.toString());
            //jLabel19.setText(file.toString());
            docUji = file.toString();

            //menampilkan isi dokumen asli
            try {
                Scanner sc;
                sc = new Scanner(new File(docUji));
                String tmp = "";
                while (sc.hasNextLine()){
                    tmp += sc.nextLine();
                }
                jTextArea12.setText(tmp);
            } catch (FileNotFoundException ex) {

Logger.getLogger(Interface.class.getName()).log(Level.SEVERE,
null, ex);
            }
```

Source code 4.1 Implementasi Proses Input dokumen dan menampilkan isi file
Sumber: [Implementasi]

4.4.2 Implementasi *Tokenizing*

Setelah kedua dokumen dipilih oleh user, dokumen tersebut mengalami proses *pre-processing* yang pertama yaitu *tokenizing*. Proses ini merupakan pengambilan tiap kata pada dokumen, menghilangkan tanda baca dan merubah semua huruf ke dalam huruf kecil (*lowercase*). Implementasi proses tersebut dijelaskan pada *Source code* 4.2

```
ArrayList<ArrayList<String>> listReview = new
ArrayList<ArrayList<String>>();
try {
        Scanner sc = new Scanner(new File(namaFile));
        while (sc.hasNextLine()){
            ArrayList<String> list = new ArrayList<String>();
            String strLower = sc.nextLine().toLowerCase();

            StringTokenizer st = new StringTokenizer(strLower, "
                  1234567890`~!@#$%^&*)(-_=+}{][\\|;:'\",<.>/?
            ");
```

```
            while(st.hasMoreTokens()){
                String str = st.nextToken();
                list.add(str);
                //System.out.print(str+"|");
            }
            listReview.add(list);
        }
}
catch (FileNotFoundException ex) {
        System.out.println(ex);
}
```

Source code 4.2 Implementasi Proses *Tokenizing*
Sumber: [Implementasi]

4.4.3 Implementasi *Filtering*

Setelah mengalami *tokenizing*, dokumen tersebut dilakukan tahap selanjutnya yaitu *filtering*. Tahap ini menghapus kata-kata yang tidak penting pada dokumen misalnya kata hubung. Penghapusan kata pada tahap ini didasarkan pada kamus *stopword*. Implementasi proses *filtering* dijelaskan pada *Source code* 4.3

```
HashSet<String> stoplist = new HashSet<String>();
try {
        Scanner sc = new Scanner(new File(namaFile));
        while (sc.hasNextLine()) {
             String str = sc.nextLine();
             stoplist.add(str);
        }

        for(int i=0; i<listReview.size(); i++){
          for(int j=0; j<listReview.get(i).size(); j++){
            if(stoplist.contains(listReview.get(i).get(j))){
                listReview.get(i).remove(j);
                j-=1;
            }
          }
        }
}
```

```
catch (FileNotFoundException ex) {
        System.out.println(ex);
}
```

Source code 4.3 Implementasi Proses *Filtering*
Sumber: [Implementasi]

4.4.4 Implementasi *Stemming*

Tahap selanjutnya setelah *filtering* adalah *stemming*. Thap ini merupkan tahap akhir dari *pre-processing*. Pada tahap ini dokumen mengalami perubahan kata ke dalam bentuk kata dasarnya. Proses implementasi *stemming* dijelaskan pada *Source code* 4.4

```
String[] listAwalan =
        {"me","di","ke","pe","se","be","ku","ber","ter","per","
        mem","pem","men","pen","meng","peng"};
String[] listAkhiran = {"an","in",
        "mu","ku","lah","kah","pun","kan","nya"};
        HashSet<String> listKDasar = new HashSet<String>();
try {
         Scanner sc = new Scanner(new File(namaFile));
         while (sc.hasNextLine()) {
                String str = sc.nextLine();
                listKDasar.add(str);
         }

         for(int i=0; i<listReview.size(); i++){
             for(int j=0; j<listReview.get(i).size(); j++){
                 String KD = listReview.get(i).get(j);
                 String tmpKD = KD;
                 boolean KETEMU = false;
                 String[] awalan = {"",""};
                 String[] akhiran = {"","",""};

                 if(listKDasar.contains(KD))
                    continue;
                 else{
                     for(int a=0; a<2; a++){
                         for(int b=2; b<=4; b++){
```

```
                        if(KD.length()<5)
                            continue;

                            String tmpAwalan =
                            KD.substring(0, b);

                        if(Arrays.asList(listAwalan).conta
                        ins(tmpAwalan)){
                            awalan[a] = tmpAwalan;

                            tmpKD = KD.substring(b,
                            KD.length());

                       if(listKDasar.contains(tmpKD)){
                            KETEMU = true;
                            break;
                       }
                }
            }
            KD=tmpKD;
            if(a>0 && awalan[a]=="" && awalan[a-1]!="")
                            awalan[a]=awalan[a-1];

            if(KETEMU)
                break;
    }

    if(!KETEMU){
       for(int a=0; a<3; a++){
           for(int b=2; b<4; b++){
              if(KD.length()<5)
              continue;
              String tmpAkhiranKD.substring(KD.length()-b);

              if(Arrays.asList(listAkhiran).contains(tmpAkhi
              ran)){
                 akhiran[a] = tmpAkhiran;
                 tmpKD = KD.substring(0, KD.length()-b);
```

```
                if(listKDasar.contains(tmpKD)){
                    KETEMU = true;
                    break;
                }
            }
        }
        KD=tmpKD;
        if(a==1 && akhiran[a]=="" && akhiran[a-1]!="")
            akhiran[a+1]=akhiran[a]=akhiran[a-1];
         else if(a==2 && akhiran[a]=="" && akhiran[a-
         1]!=""){
            akhiran[a]=akhiran[a-1];
            akhiran[a-1]=akhiran[a-2];
         }

         if(KETEMU)
            break;
}

if(listKDasar.contains(awalan[0]+awalan[1]+KD))
     KD = awalan[0]+awalan[1]+KD;
else if
(listKDasar.contains(awalan[0]+awalan[1]+KD+akhiran[2]
))
      KD = awalan[0]+awalan[1]+KD+akhiran[2];
else if
(listKDasar.contains(awalan[0]+awalan[1]+KD+akhiran[2]
+akhiran[1]))
      KD =
      awalan[0]+awalan[1]+KD+akhiran[2]+akhiran[1];
else if(listKDasar.contains(awalan[1]+KD))
      KD = awalan[1]+KD;
else if(listKDasar.contains(awalan[1]+KD+akhiran[2]))
      KD = awalan[1]+KD+akhiran[2];
else if
(listKDasar.contains(awalan[1]+KD+akhiran[2]+akhiran[1
]))
      KD = awalan[1]+KD+akhiran[2]+akhiran[1];
```

```
                else if(listKDasar.contains("k"+KD))
                    KD = "k"+KD;
                else if(listKDasar.contains("t"+KD))
                    KD = "t"+KD;
                else if(listKDasar.contains("s"+KD))
                   KD = "s"+KD;
                else if(listKDasar.contains("p"+KD))
                   KD = "p"+KD;
              }
            }
            listReview.get(i).set(j, KD);
        }
      }
    }
    catch (FileNotFoundException ex) {
                System.out.println(ex);
    }
```

Source code 4.4 Implementasi Proses *Stemming*
Sumber: [Implementasi]

4.4.5 Implementasi *Synonym Recognition*

Setelah mengalami *pre-processing*, dokumen mengalami tahap *synonym recognition*, namun tahap ini sifatnya opsional tergantung keinginan user. Pada tahap ini terjadi penggantian kata ke dalam bentuk sinonimnya. Implementasi proses tahap *synonym recognition* dijelaskan pada *Source code* 4.5

```
HashMap<String, String> kamus = new HashMap<String, String>();
try {
        Scanner sc = new Scanner(new File(namaFile));
        while (sc.hasNextLine()) {
            String[] str = sc.nextLine().split("=");
            kamus.put(str[0], str[1]);
        }

        for(int i=0; i<listReview.size(); i++){
            for(int j=0; j<listReview.get(i).size(); j++){
                if(kamus.containsKey(listReview.get(i).get(j))){
                    listReview.get(i).set(j,
                    kamus.get(listReview.get(i).get(j)));
```

```
                }
            }
        }
}
catch (FileNotFoundException ex) {
        System.out.println(ex);
}
```

Source code 4.5 Implementasi Proses *Synonym recognition*
Sumber: [Implementasi]

4.4.6 Implementasi Algoritma VSM

Tahap selanjutnya yaitu perhitungan *similarity* dokumen dengan menggunakan Vector Space Model (VSM). Kedua dokumen tersebut mengalami perhitungan kesamaan berdasarkan vektor kata penyusunnya. Implementasi proses algoritma VSM dijelaskan pada *Source code* 4.6

```
ArrayList<ArrayList<String>> tf = new
ArrayList<ArrayList<String>>();
for(int a=0; a<tempStem.size(); a++){
    ArrayList<String> termfreq = new ArrayList<String>();
    for(int b=0; b<tempStem.get(a).size(); b++){
        termfreq.add(tempStem.get(a).get(b));
    }
    tf.add(termfreq);
}
for(int a=0; a<tempStem2.size(); a++){
     ArrayList<String> termfreq = new ArrayList<String>();
     for(int b=0; b<tempStem2.get(a).size(); b++){
         termfreq.add(tempStem2.get(a).get(b));
     }
     tf.add(termfreq);
}
TermFreq wtf = new TermFreq(tf);
tempTF = wtf.getTf();

for(int a=0; a<tempTF.size(); a++){
    for(int b=0; b<tempTF.get(a).size(); b++){
       //logTF=0; kuadratLogTF=0;
```

```
        if(b==0) {
            if(tempTF.get(a).get(b)== 0) {logTF1 = 0;}
            else {logTF1 = (1 +
                (Math.log10(tempTF.get(a).get(b))));}
            kuadratLogTF = (logTF1*logTF1);
            jumlahKuadratLogTF1 = jumlahKuadratLogTF1 +
            kuadratLogTF;
        }
        else {
            if(tempTF.get(a).get(b)== 0) {logTF2 = 0;}
            else {logTF2 = (1 +
                (Math.log10(tempTF.get(a).get(b))));}
                kuadratLogTF = (logTF2*logTF2);
                jumlahKuadratLogTF2 = jumlahKuadratLogTF2 +
                kuadratLogTF;
            }

        System.out.print(tempTF.get(a).get(b)+"|");
        }
        jumlahLogTF = jumlahLogTF + (logTF1 * logTF2);
        System.out.println();
}
 vsm =
(jumlahLogTF)/(Math.sqrt(jumlahKuadratLogTF1)*Math.sqrt(jumlahKu
adratLogTF2));
```

Source code 4.6 Implementasi Proses Algoritma Vector Space Model *(cosine similarity)*
Sumber: [Implementasi]

4.4.7 Implementasi Algoritma Jaccard Coefisien

Tahap selanjutnya dilakukan perhitungan menggunakan jaccard coefisien. Kedua dokumen dihitung tingkat kesamaannya berdasarkan jumlah kata yang sama dan jumlah keseluruhan kata penyusunnya. Implementasi proses algoritma Jaccard Coefisien dijelaskan pada *Source code* 4.7

```
ArrayList<ArrayList<String>> tf = new
ArrayList<ArrayList<String>>();
for(int a=0; a<tempStem.size(); a++){
```

```
        ArrayList<String> termfreq2 = new ArrayList<String>();
        for(int b=0; b<tempStem.get(a).size(); b++){
            termfreq2.add(tempStem.get(a).get(b));
        }
        tf.add(termfreq2);
    }
    for(int a=0; a<tempStem2.size(); a++){
        ArrayList<String> termfreq2 = new ArrayList<String>();
        for(int b=0; b<tempStem2.get(a).size(); b++){
            termfreq2.add(tempStem2.get(a).get(b));
        }
        tf.add(termfreq2);
    }

    TermFreq wtf = new TermFreq(tf);
    tempTF = wtf.getTf();

    for(int a=0; a<tempTF.size(); a++){
        for(int b=0; b<tempTF.get(a).size(); b++){
            if( b == 0) {temp = tempTF.get(a).get(b);}
            else {min = Math.min(temp, tempTF.get(a).get(b));}

            System.out.print(tempTF.get(a).get(b)+"|");
            total = total+tempTF.get(a).get(b);
        }
        jumlahMin = jumlahMin+min;
        System.out.println();
    }
    jaccard = ((2*jumlahMin)/total);
    System.out.print(jaccard);

    return jaccard;
```

Source code 4.7 Implementasi Proses Algoritma Jaccard Coefiisen
Sumber: [Implementasi]

4.4.8 Implementasi Hitung Nilai Kesamaan (*Similarity*)

Tahap terakhir dari aplikasi ini adalah melakukan kombinasi antara algoritma Vector Space Model (VSM) dan Jaccard Coefisien. Kombinasi ini

dilakukan dengan memasukkan variabel α, β, k dan *similaity* yang didapatkan pada masing-masing algoritma ke dalam suatu formula. Selain itu dilakukan pula normalisasi hasil untuk mendapatkan *similarity* hasil kombinasi yang sebenarnya. Implementasi proses kombinasi algoritma Vector Space Model dan Jaccard Coefisien dijelaskan pada *Source code* 4.8.

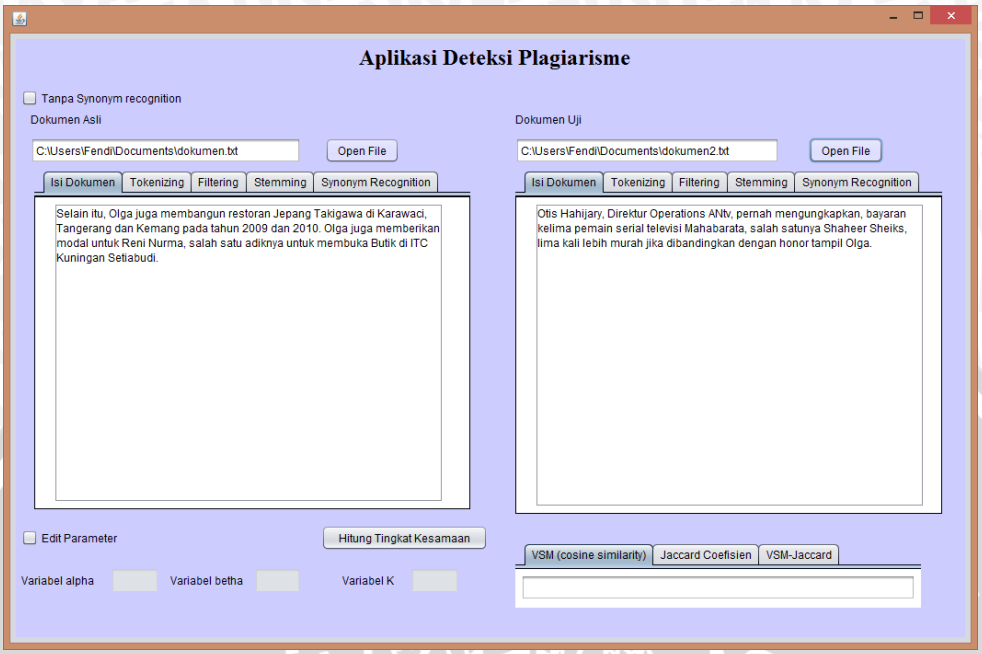
```
VSM vsm = new VSM();
tempVSM = vsm.ProsesVSM();

JaccardCoefisien jaccard = new JaccardCoefisien();
tempJaccard = jaccard.ProsesJaccard();

similarity = (tempVSM-alpha)*(tempJaccard-betha)/k;

if(alpha>betha){
       min = (1-alpha)*(0-betha)/k;
}
else{
       min = (0-alpha)*(1-betha)/k;
}
max = (1-alpha)*(1-betha)/k;

if(tempVSM == 0 && tempJaccard == 0 ) norsim = 0;
else norsim = (similarity-min)/(max-min);
```

Source code 4.8 Implementasi Proses Kombinasi Algoritma Vector Space Model (*cosne similarity*) dan Jaccard Coefisien
Sumber: [Implementasi]

4.5 Implementasi Aplikasi Deteksi Plagiarisme

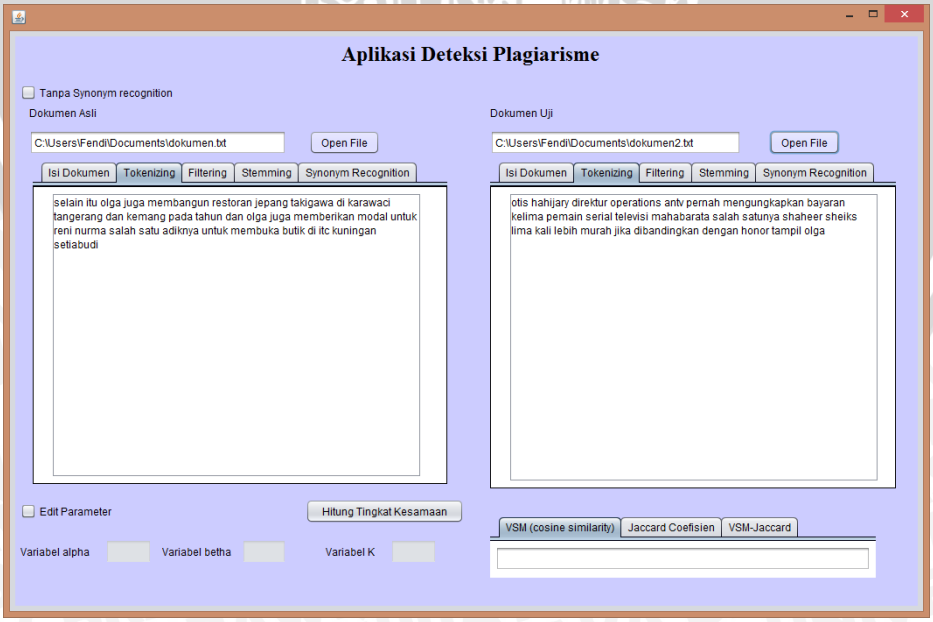
Dalam sub-bab ini akan dijelaskan mengenai langkah-langkah yang dilakukan user untuk menjalankan aplikasi ini. Langkah pertama user dapat mencentang "Tanpa synonm recognition" jika tidak ingin menggunakan *synonym recognition* pada tahapan deteksi plagiarisme. Setelah itu user harus memasukkan dokumen asli dan dokumen uji yang akan dibandingkan prosentase kemiripannya. User hanya perlu menekan tombol "Open file" pada masing-masing dokumen untuk memilih dokumen .txt yang akan dibandingkan. Jika langkah ini berhasil

maka akan muncul isi dokumen tersebut pada antarmuka yang disediakan. Langkah ini ditunjukkan pada Gambar 4.1.



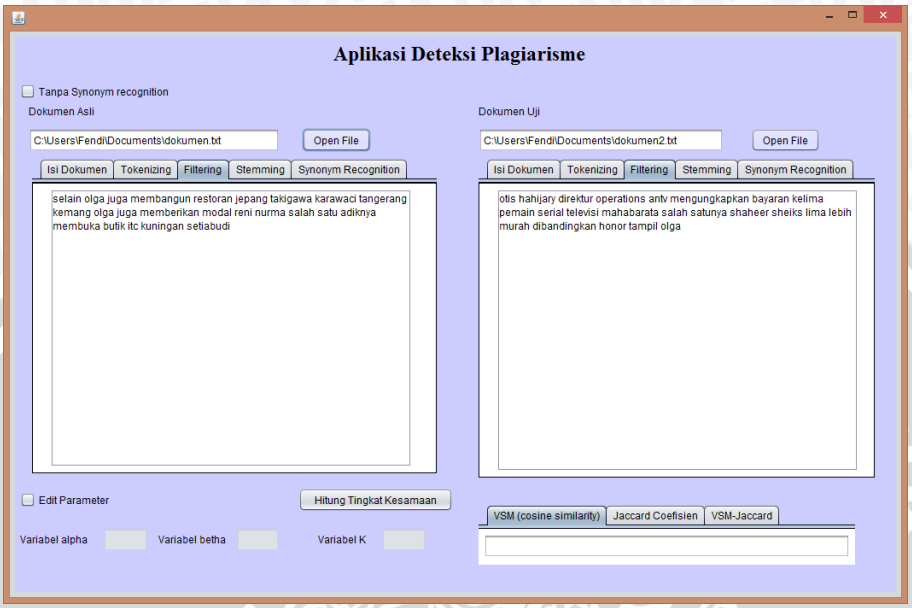
Gambar 4.1 Antarmuka Input dan Isi Dokumen Asli dan Dokumen Uji
Sumber: [Implementasi]

Selain dapat melihat isi dokumen dokumen asli dan dokumen uji, user dapat pula melihat hasil *tokenizing* pada tab area "*Tokenizing*". Gambar 4.2 menunjukkan hasil *tokenizing* pada dokumen asli dan dokumen uji.



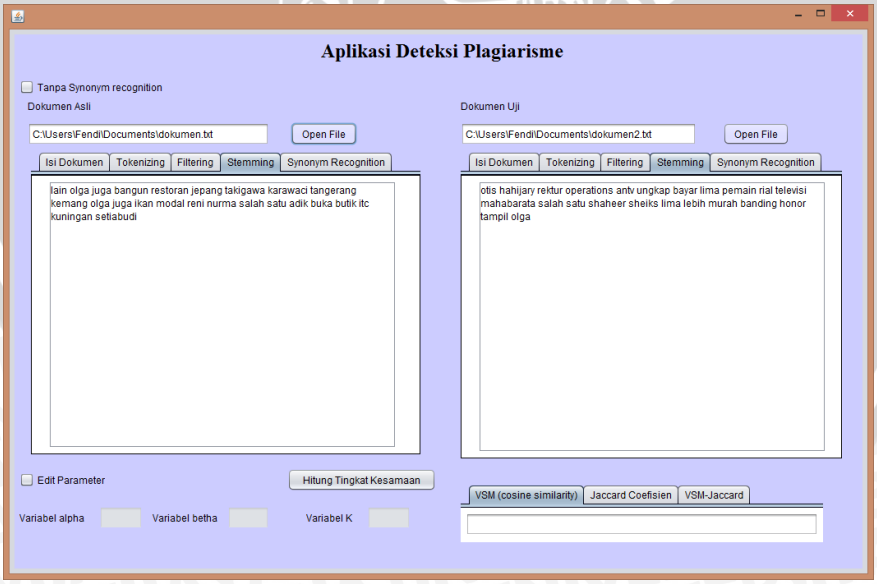
Gambar 4.2 Antarmuka Hasil *Tokenizing* Dokumen Asli dan Dokumen Uji
Sumber: [Implementasi]

Selain itu, user dapat melihat hasil *filtering* pada dokumen asli dan dokumen uji pada tab area "*Filtering*". Gambar 4.3 menunjukkan hasil *filtering* pada dokumen asli dan dokumen uji.



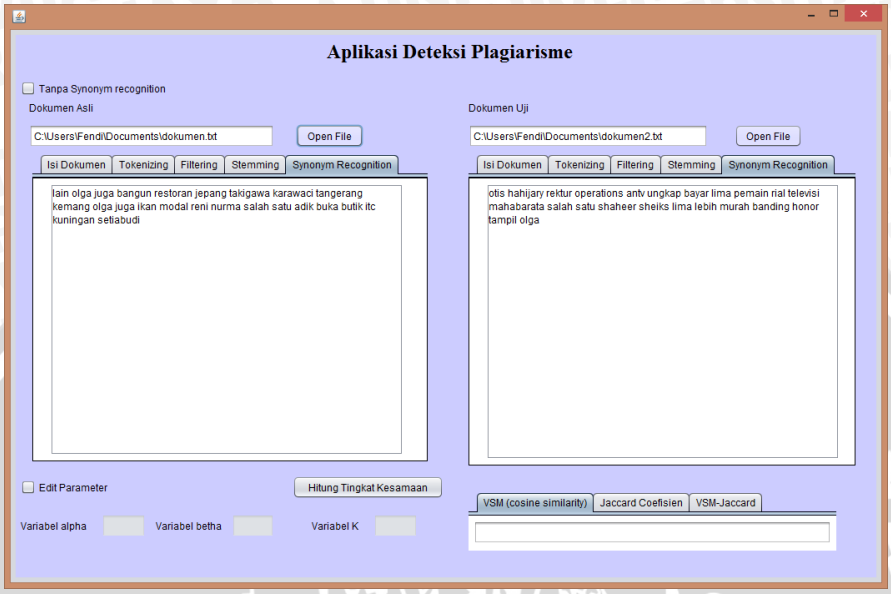
Gambar 4.3 Antarmuka Hasil *Filtering* Dokumen Asli dan Dokumen Uji
Sumber: [Implementasi]

User dapat pula melihat hasil *stemming* pada tab area "*stemming*". Gambar 4.4 menunjukkan hasil *stemming* pada dokumen asli dan dokumen uji.



Gambar 4.4 Antarmuka Hasil *Stemming* Dokumen Asli dan Dokumen Uji
Sumber: [Implementasi]

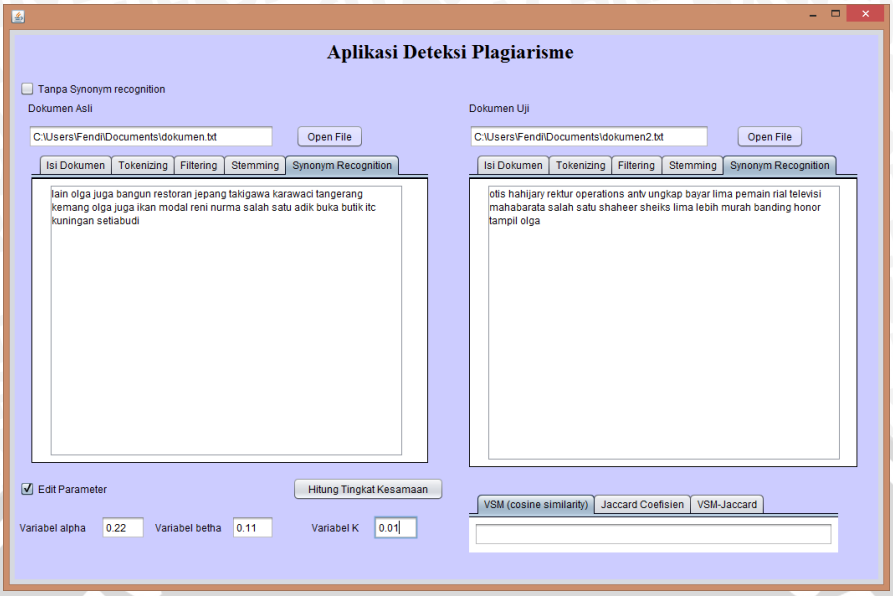
Jika user tidak mencentang "tanpa *synonym recognition*" maka user dapat melihat hasil *synonym recognition* pada dokumen asli dan dokumen uji. Gambar 4.5 menunjukkan hasil *synonym recognition* pada dokumen asli dan dokumen uji



Gambar 4.5 Antarmuka Hasil *Synonym Recognition* Dokumen Asli dan Dokumen Uji

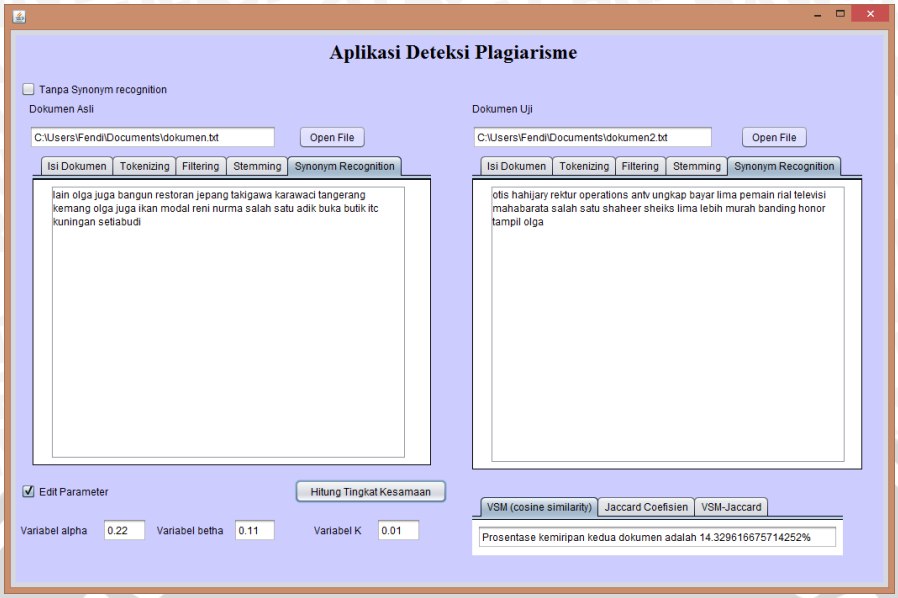
Sumber: [Implementasi]

Setelah itu user dapat merubah variabel α, β dan k dengan mencentang "edit parameter" terlebih dahulu. Jika sudah mencentangnya user dapat memasukkan masing-masing isian parameter tersebut. Namun jika tidak mencentangnya, sistem akan melakukan deteksi plagiarisme dengan variabel α, β dan K yang telah ditentukan. Gambar 4.6 menunjukkan antarmuka saat mengubah parameter α, β dan K.



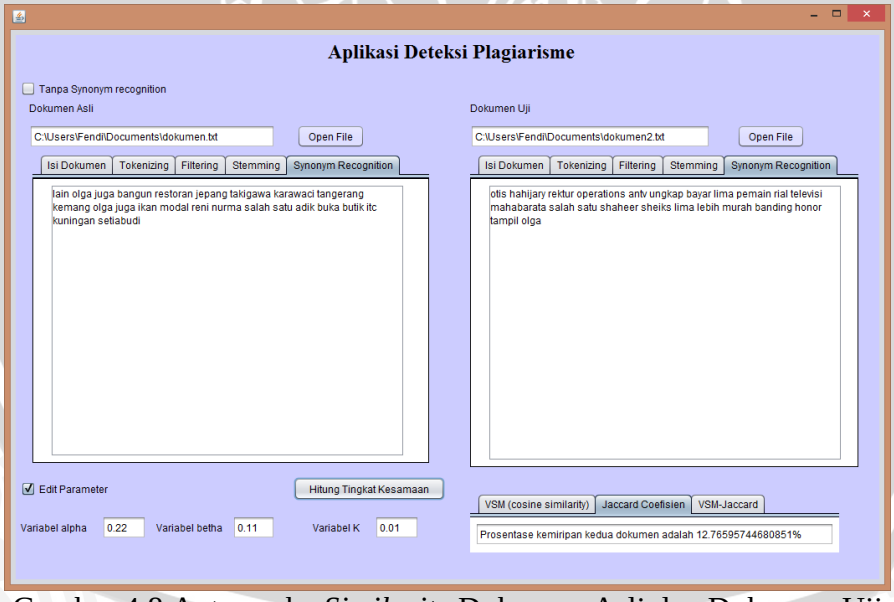
Gambar 4.6 Antarmuka Merubah Parameter α, β dan K
Sumber: [Implementasi]

Setelah itu, user dapat melihat prosentase kemiripan dokumen asli dan dokumen uji dengan menekan tombol "Hitung tingkat kesamaan". Lalu sistem akan menampilkan hasil prosentase *similarity* dengan algoritma VSM, jaccard coefisien dan kombinasi kedua algoritma tersebut. Masing-masing hasil *similarity* dari ketiga algoritma tersebut dipisahkan dengan *tab area* seperti ditunjukkan pada Gambar 4.7, 4.8 dan 4.9.



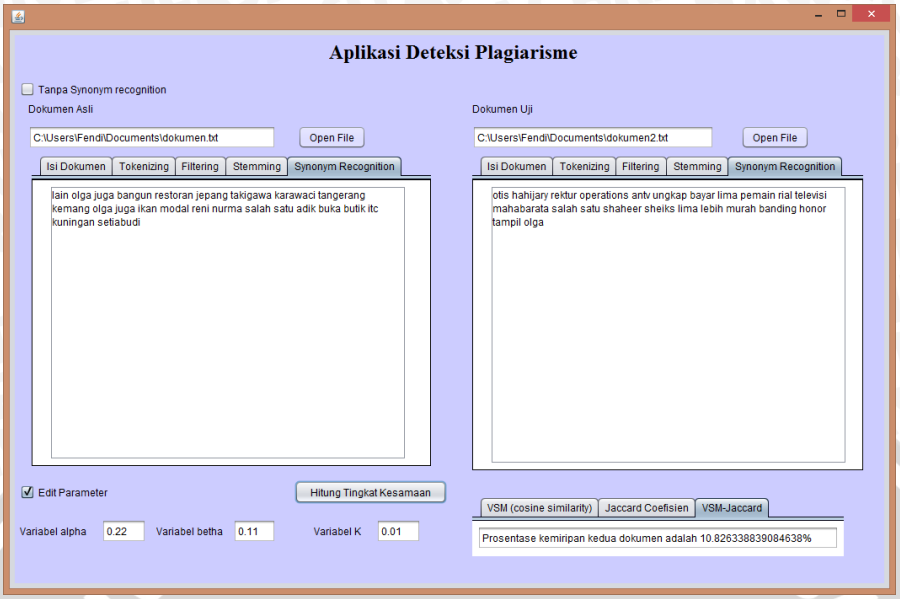
Gambar 4.7 Antarmuka *Similarity* Dokumen Asli dan Dokumen Uji Menggunakan Algoritma VSM (*cosine similarity*)
Sumber: [Implementasi]

Sedangkan hasil *similarity* Jaccard Coefisien ditampilkan pada Gambar 4.8 di bawah ini.



Gambar 4.8 Antarmuka *Similarity* Dokumen Asli dan Dokumen Uji Menggunakan Algoritma Jaccard Coefisien
Sumber: [Implementasi]

Sedangkan hasil *similarity* kombinasi VSM-Jaccard ditampilkan pada Gambar 4.9 di bawah ini.



Gambar 4.9 Antarmuka *Similarity* Dokumen Asli dan Dokumen Uji Menggunakan Kombinasi Algoritma VSM (*cosine similarity*) dan Jaccard Coefisien

Sumber: [Implementasi]

BAB V
PENGUJIAN DAN ANALISA

5.1 Pengujian Sistem

Bab ini menjelaskan tentang pengujian yang dilakukan setelah aplikasi sudah diimplementasikan. Tujuan dari pengujian adalah memastikan bahwa sistem sudah berjalan dengan baik dan sesuai dengan kebutuhan.

5.2 Data Pengujian

Data pengujian pada penelitian ini berasal dari penelitian sebelumnya yang dilakukan oleh Christian S.K.A pada skripsinya yang berjudul "Perancangan Sistem Deteksi Plagiarisme Dokumen Teks Menggunakan Algoritma Damerau Levenshtein Distance". Data penelitian tersebut dilakukan perubahan sedemikian rupa pada dokumen asli sehingga diperoleh kesamaan kata pada dokumen uji sesuai kebutuhan. Adapun dokumen teks yang dilakukan uji coba dijelaskan pada Tabel 5.1

Tabel 5.1 Data Dokumen Asli dan Dokumen Uji yang Digunakan untuk Pengujian

Dokumen Asli	**Dokumen Uji**	**Manipulasi yang Dilakukan**	***Similarity* (%)**
A-Asli.txt	A.20-kata.txt	Pemotongan kata sebanyak 20%	80
	A.40-kata.txt	Pemotongan kata sebanyak 40%	60
	A-AP.txt	Perubahan kata kerja pasif menjadi aktif dan sebaliknya	100
	A-TK.txt	Penukaran susunan kalimat	100
	A-Total.txt	Pemotongan kata sebanyak 20%,	80

Dokumen Asli	Dokumen Uji	Manipulasi yang Dilakukan	*Similarity* (%)
A-Asli.txt		penukaran susunan kalimat dan perubahan kata kerja pasif	
	A-TS.txt	Perubahan kata ke dalam bentuk sinonim	100
B-Asli.txt	B.20-kata.txt	Pemotongan kata sebanyak 20%	80
	B.40-kata.txt	Pemotongan kata sebanyak 40%	60
	B-AP.txt	Perubahan kata kerja pasif menjadi aktif dan sebaliknya	100
	B-TK.txt	Penukaran susunan kalimat	100
	B-Total.txt	Pemotongan kata sebanyak 20%, penukaran susunan kalimat dan perubahan kata kerja pasif	80
	B-TS.txt	Perubahan kata ke dalam bentuk sinonim	100
C-Asli.txt	C.20-kata.txt	Pemotongan kata sebanyak 20%	80
	C.40-kata.txt	Pemotongan kata sebanyak 40%	60

Dokumen Asli	Dokumen Uji	Manipulasi yang Dilakukan	*Similarity* (%)
C-Asli.txt	C-AP.txt	Perubahan kata kerja pasif menjadi aktif dan sebaliknya	100
	C-TK.txt	Penukaran susunan kalimat	100
	C-Total.txt	Pemotongan kata sebanyak 20%, penukaran susunan kalimat dan perubahan kata kerja pasif	80
	C-TS.txt	Perubahan kata ke dalam bentuk sinonim	100

5.3 Hasil Percobaan Sistem

Hasil percobaan sistem yang ditampilkan pada bab ini menyesuaikan dengan skenario uji coba yang direncanakan pada bab perancangan pengujian. Skenario yang diujikan antara lain pengujian variabel α, β dan K, pengujian pengaruh *synonym recognition* dan pengujian perbandingan algoritma. Prosentase *error* diperoleh dari selisih antara *similarity* sebenarnya dan *similarity* yang diperoleh sistem.

5.3.1 Hasil Uji Coba Parameter α, β dan K

Pada pengujian parameter α, β dan K dilakukan dengan membandingkan antara dokumen A-asli dan A.20-kata, A.40-kata, A-AP dan A-Total. Pada pengujian ini dilakukan secara *independen* yaitu merubah nilai parameter yang diujikan sedangkan parameter lain diberikan nilai tetap. Tabel 5.2 di bawah ini menunjukkan hasil pengujian parameter α pada dokumen A-asli dan A.20-kata.

Tabel 5.2 Hasil Pengujian Parameter α pada Dokumen A.20-kata dan A.40-kata

Parm α	Parm β	Parm K	*Similarity* (%)		*Error* (%)	
			A.20-kata	A.40-kata	A.20-kata	A.40-kata
0,1	0,1	0,01	83,46	70,95	3,46	10,95
0,2	0,1	0,01	82,88	69,72	2,88	9,72
0,3	0,1	0,01	82,13	68,13	2,13	8,13
0,4	0,1	0,01	81,13	66,01	1,13	6,01
0,5	0,1	0,01	79,74	63,04	0,26	3,04
0,6	0,1	0,01	77,65	58,59	2,35	1,41
0,7	0,1	0,01	74,16	51,18	5,84	8,82
0,8	0,1	0,01	67,18	36,34	12,82	23,66
0,9	0,1	0,01	46,24	1,11	33,76	58,89

Selanjutnya pengujian α dilakukan pada dokumen A-AP dan A-Total seperti tampak pada Tabel 5.3 di bawah ini.

Tabel 5.3 Hasil Pengujian Parameter α pada Dokumen A-AP dan A-Total

Parm α	Parm β	Parm K	*Similarity* (%)		*Error* (%)	
			A-AP	A-Total	A-AP	A-Total
0,1	0,1	0,01	94,83	79,91	5,17	0,09
0,2	0,1	0,01	94,51	79,11	5,49	0,89
0,3	0,1	0,01	94,09	78,08	5,91	1,92
0,4	0,1	0,01	93,53	76,7	6,47	3,3
0,5	0,1	0,01	92,75	74,77	7,25	5,23
0,6	0,1	0,01	91,58	71,88	8,42	8,12
0,7	0,1	0,01	89,62	67,06	10,38	12,94
0,8	0,1	0,01	85,72	57,42	14,28	22,58
0,9	0,1	0,01	74	28,51	26	51,49

Setelah mendapatkan parameter α terbaik yang memiliki *error* terkecil maka dilanjutkan dengan menguji parameter β. Hasil uji coba ditunjukkan pada Tabel 5.4 dan 5.5.

Tabel 5.4 Hasil Pengujian Parameter β pada Dokumen A.20-kata dan A.40-kata

Parm α	Parm β	Parm K	*Similarity* (%)		*Error* (%)	
			A.20-kata	A.40-kata	A.20-kata	A.40-kata
0,1	0,1	0,01	83,46	70,95	3,46	10,95
0,1	0,2	0,01	82,05	68,85	2,05	8,85
0,1	0,3	0,01	80,24	66,15	0,24	6,15
0,1	0,4	0,01	77,83	62,55	2,17	2,55
0,1	0,5	0,01	74,45	57,5	5,55	2,5
0,1	0,6	0,01	69,39	49,94	10,61	10,06
0,1	0,7	0,01	60,94	37,33	19,06	22,67
0,1	0,8	0,01	44,06	12,11	35,94	47,89
0,1	0,9	0,01	1,11	1,11	78,89	58,89

Selanjutnya pengujian β dilakukan pada dokumen A-AP dan A-Total seperti tampak pada Tabel 5.5 di bawah ini.

Tabel 5.5 Hasil Pengujian Parameter β pada Dokumen A-AP dan A-Total

Parm α	Parm β	Parm K	*Similarity* (%)		*Error* (%)	
			A-AP	A-Total	A-AP	A-Total
0,1	0,1	0,01	92,68	79,91	7,32	0,09
0,1	0,2	0,01	92,26	78,33	7,74	1,67
0,1	0,3	0,01	91,72	76,3	8,28	3,7
0,1	0,4	0,01	91	73,59	9	6,41
0,1	0,5	0,01	89,99	69,79	10,01	10,21
0,1	0,6	0,01	88,48	64,1	11,52	15,9
0,1	0,7	0,01	85,97	54,62	14,03	25,38
0,1	0,8	0,01	80,93	35,65	19,07	44,35
0,1	0,9	0,01	1,11	1,11	98,89	78,89

Setelah mendapatkan parameter β terbaik yang memiliki *error* terkecil maka dilanjutkan dengan menguji parameter K. Pengujian dilakukan secara

independen, hasil dari uji coba ditunjukkan pada Tabel 5.6 dan 5.7. Tabel 5.6 di bawah ini menunjukkan hasil uji coba K pada dokumen A.20-kata dan A.40-kata.

Tabel 5.6 Hasil Pengujian Parameter K pada Dokumen A.20-kata dan A.40-kata

Parm α	**Parm β**	**Parm K**	***Similarity* (%)**		***Error* (%)**	
			A.20-kata	**A.40-kata**	**A.20-kata**	**A.40-kata**
0,1	0,1	0,01	83,46	70,95	3,46	10,95
0,1	0,1	0,02	83,46	70,95	3,46	10,95
0,1	0,1	0,03	83,46	70,95	3,46	10,95
0,1	0,1	0,04	83,46	70,95	3,46	10,95
0,1	0,1	0,05	83,46	70,95	3,46	10,95
0,1	0,1	0,06	83,46	70,95	3,46	10,95
0,1	0,1	0,07	83,46	70,95	3,46	10,95
0,1	0,1	0,08	83,46	70,95	3,46	10,95
0,1	0,1	0,09	83,46	70,95	3,46	10,95

Selanjutnya pengujian K dilakukan pada dokumen A-AP dan A-Total seperti tampak pada Tabel 5.7 di bawah ini.

Tabel 5.7 Hasil Pengujian Parameter K pada Dokumen A-AP dan A-Total

Parm α	**Parm β**	**Parm K**	***Similarity* (%)**		***Error* (%)**	
			A-AP	**A-Total**	**A-AP**	**A-Total**
0,1	0,1	0,01	92,68	79,91	7,32	0,09
0,1	0,1	0,02	92,68	79,91	7,32	0,09
0,1	0,1	0,03	92,68	79,91	7,32	0,09
0,1	0,1	0,04	92,68	79,91	7,32	0,09
0,1	0,1	0,05	92,68	79,91	7,32	0,09
0,1	0,1	0,06	92,68	79,91	7,32	0,09
0,1	0,1	0,07	92,68	79,91	7,32	0,09
0,1	0,1	0,08	92,68	79,91	7,32	0,09
0,1	0,1	0,09	92,68	79,91	7,32	0,09

5.3.2 Hasil Uji Coba Pengaruh *Synonym Recognition*

Skenario pengujian yang kedua bertujuan untuk melihat seberapa besar pengaruh *synonym recognition* terhadap deteksi plagiarisme. Pengujian dilakukan dengan membandingkan *similarity* dokumen asli dan dokumen uji yang menggunakan *synonym recognition* dan tanpa *synonym recognition.* Tabel 5.8 di bawah ini menunjukkan hasil uji coba pengaruh *synonym recognition.*

Tabel 5.8 Hasil Pengujian Pengaruh *Synonym Recognition* terhadap Prosentase *Error*

Dokumen asli	**Dokumen uji**	***Similarity*(%)**		***Error* (%)**	
		Menggunakan *Synonym recognition*	**Tanpa *Synonym recognition***	**Menggunakan *Synonym recognition***	**Tanpa *Synonym recognition***
A-Asli	A.20-kata	83,46	83,46	3,46	3,46
	A.40-kata	70,95	70,95	10,95	10,95
	A-AP	94,83	94,83	5,17	5,17
	A-TK	100	100	0	0
	A-Total	79,91	79,91	0,09	0,09
	A-TS	100	97,78	0	2,22
B-Asli	B.20-kata	79,46	79,48	0,54	0,52
	B.40-kata	57,74	57,31	2,26	2,69
	B-AP	99,02	99,02	0,98	0,98
	B-TK	100	100	0	0
	B-Total	78,13	78,16	1,87	1,84
	B-TS	100	91,41	0	8,59
C-Asli	C.20-kata	78,14	78,17	1,86	1,83
	C.40-kata	54,21	54,02	5,79	5,98
	C-AP	95,37	95,39	4,63	4,61
	C-TK	100	100	0	0
	C-Total	75,71	75,75	4,29	4,25
	C-TS	100	94,31	0	5,69
Total *error*				41,89	58,87
Rata-rata *error*				2,33	3,27

Dari pengujian di atas menunjukkan bahwa hasil deteksi plagiarisme dengan *synonym recognition* memiliki rata-rata *error* yang lebih sedikit yaitu 2,21% dari pada tanpa menggunakan *synonym recognition* yang memiliki *error* sebesar 3,01%.

5.3.3 Hasil Uji Coba Menggunakan VSM, Jaccard Coefisien dan VSM-Jaccard

Pada skenario pengujian yang terakhir adalah membandingkan tingkat keakuratan *similarity* antara algoritma VSM, Jaccard Coefisien dan kombinasi VSM-Jaccard. Pengujian dilakukan pada semua dokumen uji kemudian dihitung prosentase plagiat (*similarity*) dan rata-rata *error* yang dihasilkan masing-masing algoritma. Pada pengujian ini digunakan proses *synonym recognition*, nilai α yang digunakan adalah 0,1, nilai β adalah 0,1 dan nilai K adalah 0,01. Perbandingan ketiga algoritma ini disajikan dalam Tabel 5.9

Tabel 5.9 Hasil Perbandingan *Error Similarity* antara VSM, Jaccard dan Kombinasi VSM-Jaccard

Dokumen asli	**Dokumen uji**	***Similarity* (%)**			***Error* (%)**		
		VSM	**Jaccard**	**VSM-Jaccard**	**VSM**	**Jaccard**	**VSM-Jaccard**
A-Asli	A.20-kata	90,24	86,42	83,46	10,24	6,42	3,46
	A.40-kata	88,99	82,76	70,95	28,99	22,76	10,95
	A-AP	97,59	97,69	94,83	2,41	2,31	5,17
	A-TK	100,0	100,0	100	0	0	0
	A-Total	93,3	87,7	79,91	13,3	7,7	0,09
	A-TS	100,0	100,0	100	0	0	0
B-Asli	B.20-kata	92,44	88,07	79,46	12,44	8,07	0,54
	B.40-kata	83,82	73,36	57,74	23,82	13,36	2,26
	B-AP	99,66	99,45	99,02	0,34	0,55	0,98
	B-TK	100,0	100,0	100	0	0	0
	B-Total	91,7	87,46	78,13	11,7	7,46	1,87
	B-TS	100,0	100,0	100	0	0	0

Dokumen asli	Dokumen uji	*Similarity* (%)			*Error* (%)		
		VSM	Jaccard	VSM-Jaccard	VSM	Jaccard	VSM-Jaccard
C-Asli	C.20-kata	91,5	87,65	78,14	11,5	7,65	1,86
	C.40-kata	80,01	72,73	54,21	20,01	12,73	5,79
	C-AP	97,98	97,8	95,37	2,02	2,2	4,63
	C-TK	100,0	100,0	100	0	0	0
	C-Total	90,24	86,42	75,71	10,24	6,42	4,29
	C-TS	100,0	100,0	100	0	0	0
Total *error*					147,01	97,63	41,89
Rata-rata *error*					8,17	5,42	2,33

Dari hasil pengujian di atas dapat dilihat bahwa prosentase rata-rata *error* terkecil terdapat pada kombinasi algoritma VSM-Jaccard yaitu sebesar 2,21%. Sedangkan prosentase rata-rata *error* terbesar terletak pada algoritma Vector Space Model (VSM) sebesar 8,23%.

5.4 Analisa Hasil

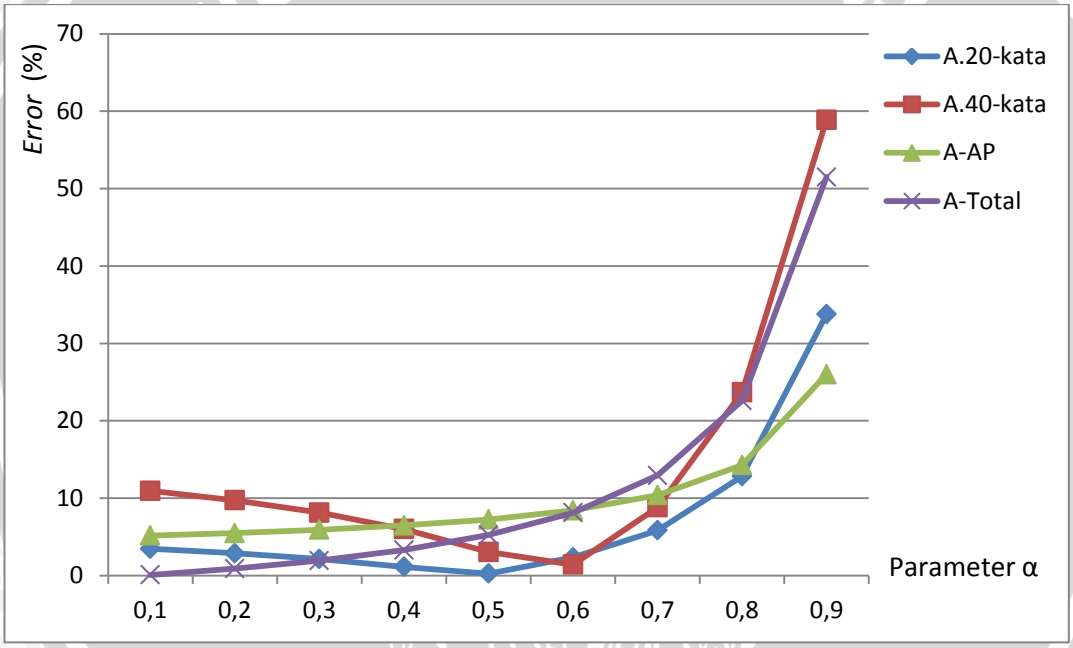
Hasil yang didapatkan menunjukkan nilai rata-rata *error* yang dihasilkan sistem menggunakan kombinasi VSM-Jaccard sebesar 2,45%. Pada dokumen uji nilai *error* terbaik yang dapat dideteksi sistem untuk tiap dokumen uji adalah 0% yaitu pada TK (Tukar Kata) dan TS (Tukar Sinonim). Hal tersebut dikarenakan algoritma VSM ataupun Jaccard tidak memperhatikan susunan kata melainkan frekuensi kata sehingga perpindahan posisi suatu kalimat pada TK tidak mempengaruhi proses deteksi *similarity*. Selain itu penambahan *synonym recognition* menyebabkan dokumen yang telah dirubah dengan kata-kata sinonimnya juga dapat dikenali sebagai plagiat, terbukti pada dokumen A-TS, B-TS dan C-TS *error* yang dihasilkan sebesar 0%. Sedangkan deteksi terburuk terjadi pada pemotongan kata 40% dengan rata-rata *error* sebesar 6,33%. Adapun analisa tiap skenario pengujian akan dijelaskan pada sub bab berikut ini.

5.4.1 Analisa Hasil Uji Coba Parameter α, β dan K

Pada analisa pengujian ini terdiri dari tiga uji coba yaitu analisa hasil parameter α, parameter β dan parameter K. Analisa hasil ketiga parameter itu dijelaskan pada sub bab di bawah ini.

5.4.1.1 Analisa Hasil Uji Coba Parameter α

Pengujian ini dilakukan untuk untuk mengetahui nilai α yang sesuai sehingga dapat menyeimbangkan *similarity* yang dihasilkan oleh algoritma Vector Space Model. Pengujian dilakukan pada dokumen A.20-kata, A.40-kata, A-AP dan A-Total. Nilai α yang diujikan sebanyak sembilan yaitu antara 0,1 sampai 0,9. Grafik pengujian parameter α tampak pada Gambar 5.1.



Gambar 5.1 Grafik Pengujian Parameter α
Sumber: [Pengujian]

Berdasarkan grafik di atas pada dokumen A.20 terjadi penurunan *error*, namun *error* mulai naik pada α = 0,5. Hal itu karena nilai *similarity* yang dihasilkan menggunakan α yang mendekati 0,5 memiliki selisih *similarity* yang kecil dengan *similarity* sebenarnya, sedangkan saat α menjauhi 0,5 selisih *similarity* yang dihasilkan mengalami peningkatan. Jadi pada pengujian dokumen ini titik α yang dapat menyeimbangkan kombinasi algoritma terletak pada α = 0,5.

Pada dokumen A.40-kata terjadi penurunan *error*, namun *error* mulai naik pada α = 0,6. Hal itu karena nilai *similarity* yang dihasilkan menggunakan α yang mendekati 0,6 memiliki selisih *similarity* yang kecil dengan *similarity* sebenarnya, sedangkan saat α menjauhi 0,6 selisih *similarity* yang dihasilkan mengalami peningkatan. Jadi pada pengujian dokumen ini titik α yang dapat menyeimbangkan kombinasi algoritma terletak pada α = 0,6.

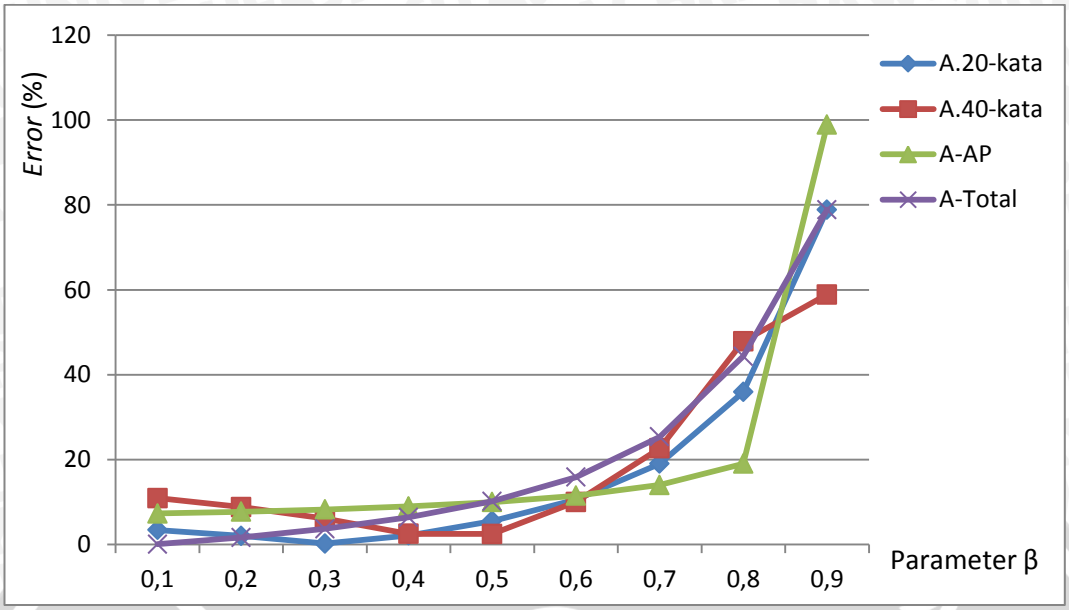
Pada dokumen A-AP terjadi kenaikan *error* saat nilai α semakin besar. Hal itu karena nilai *similarity* yang dihasilkan menggunakan α yang mendekati 0,1 memiliki selisih *similarity* yang terkecil dengan *similarity* sebenarnya, sedangkan saat α menjauhi 0,1 selisih *similarity* yang dihasilkan mengalami peningkatan. Jadi pada pengujian dokumen ini titik α yang dapat menyeimbangkan kombinasi algoritma terletak pada α = 0,1.

Pada dokumen A-Total terjadi kenaikan *error* saat nilai α semakin besar. Hal itu karena nilai *similarity* yang dihasilkan menggunakan α yang mendekati 0,1 memiliki selisih *similarity* yang terkecil dengan *similarity* sebenarnya, sedangkan saat α menjauhi 0,1 selisih *similarity* yang dihasilkan mengalami peningkatan. Jadi pada pengujian dokumen ini titik α yang dapat menyeimbangkan kombinasi algoritma terletak pada α = 0,1.

Berdasarkan analisa yang dilakukan parameter α yang terbaik adalah 0,1 karena dapat menyeimbangkan kombinasi algoritma VSM-Jaccard pada dua jenis dokumen uji A yaitu A-AP dan A-Total.

5.4.1.2 Analisa Hasil Uji Coba Parameter β

Pengujian ini dilakukan untuk untuk mengetahui nilai β yang sesuai sehingga dapat menyeimbangkan *similarity* yang dihasilkan oleh algoritma Jaccard Coefisien. Pengujian dilakukan pada dokumen A.20-kata, A.40-kata, A-AP dan A-Total. Nilai β yang diujikan sebanyak sembilan yaitu antara 0,1 sampai 0,9. Grafik pengujian parameter α tampak pada Gambar 5.2.



Gambar 5.2 Grafik Pengujian Parameter β
Sumber: [Pengujian]

Berdasarkan grafik di atas pada dokumen A.20 terjadi penurunan *error*, namun *error* mulai naik pada β = 0,3. Hal itu karena nilai *similarity* yang dihasilkan menggunakan β yang mendekati 0,3 memiliki selisih *similarity* yang kecil dengan *similarity* sebenarnya, sedangkan saat β menjauhi 0,3 selisih *similarity* yang dihasilkan mengalami peningkatan. Jadi pada pengujian dokumen ini titik α yang dapat menyeimbangkan kombinasi algoritma terletak pada β = 0,3.

Pada grafik dokumen A.40-kata terjadi penurunan *error*, namun *error* mulai naik pada β = 0,4. Hal itu karena nilai *similarity* yang dihasilkan menggunakan β yang mendekati 0,4 memiliki selisih *similarity* yang kecil dengan *similarity* sebenarnya, sedangkan saat β menjauhi 0,4 selisih *similarity* yang dihasilkan mengalami peningkatan. Jadi pada pengujian dokumen ini titik β yang dapat menyeimbangkan kombinasi algoritma terletak pada β = 0,4.

Pada grafik dokumen A-AP terjadi kenaikan *error* saat nilai β semakin besar. Hal itu karena nilai *similarity* yang dihasilkan menggunakan β yang mendekati 0,1 memiliki selisih *similarity* yang terkecil dengan *similarity* sebenarnya, sedangkan saat β menjauhi 0,1 selisih *similarity* yang dihasilkan

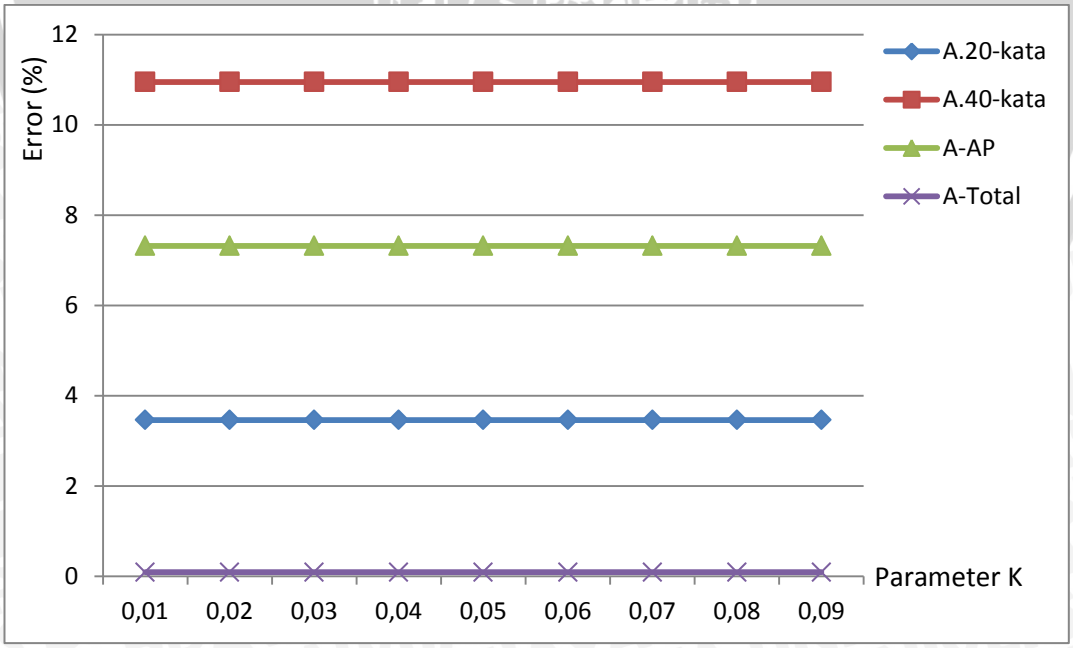
mengalami peningkatan. Jadi pada pengujian dokumen ini titik β yang dapat menyeimbangkan kombinasi algoritma terletak pada β = 0,1.

Pada grafik dokumen A-Total terjadi kenaikan *error* saat nilai β semakin besar. Hal itu karena nilai *similarity* yang dihasilkan menggunakan α yang mendekati 0,1 memiliki selisih *similarity* yang terkecil dengan *similarity* sebenarnya, sedangkan saat β menjauhi 0,1 selisih *similarity* yang dihasilkan mengalami peningkatan. Jadi pada pengujian dokumen ini titik β yang dapat menyeimbangkan kombinasi algoritma terletak pada β = 0,1.

Berdasarkan analisa yang dilakukan parameter β yang terbaik adalah 0,1 karena dapat menyeimbangkan kombinasi algoritma VSM-Jaccard pada dua jenis dokumen uji A yaitu A-AP dan A-Total.

5.4.1.3 Analisa Hasil Uji Coba Parameter K

Pengujian ini dilakukan untuk untuk mengetahui nilai K yang sesuai sehingga dapat menyeimbangkan kombinasi algoritma VSM dan Jaccard serta pengurangan oleh parameter α dan β. Pengujian dilakukan pada dokumen A.20-kata, A.40-kata, A-AP dan A-Total. Nilai parameter K yang diujikan antara 0,01-0,09. Grafik pengujian parameter Ktampak pada Gambar 5.3.

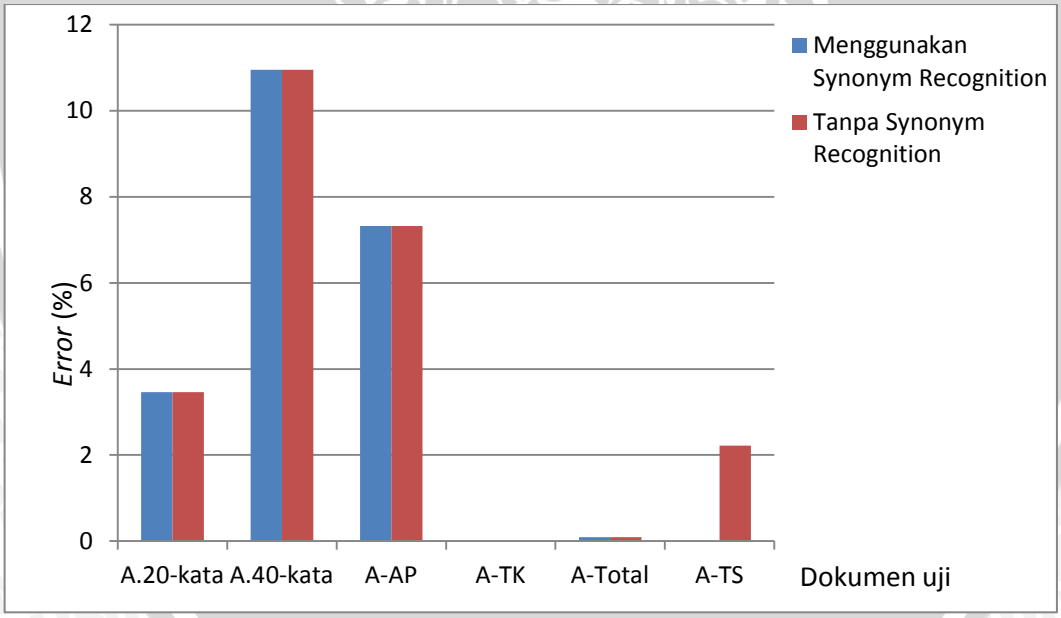


Gambar 5.3 Grafik Pengujian Parameter K
Sumber: [Pengujian]

Berdasarkan grafik di atas tampak pada parameter K tidak menunjukkan penurunan atau kenaikan prosentasi *error* secara signifikan. Parameter ini relatif konstan karena parameter K tidak menurunkan *similarity* VSM atau Jaccard namun sebagai *degree* atau pembagi untuk mendapatkan hasil yang tidak terlalu kecil sehingga nilai yang digunakan saat uji coba antara 0,01-0,09. Karena nilai K sebagai pembagi maka nilai yang digunakan tidak boleh 0. Pada semau jenis dokumen yang diujikan yaitu A.20-kata, A.40-kata, A-AP, A-Total semua grafik *error* yang dihasilkan cenderung konstan meskipun nilai K berubah.

5.4.2 Analisa Hasil Uji Coba Pengaruh *Synonym Recognition*

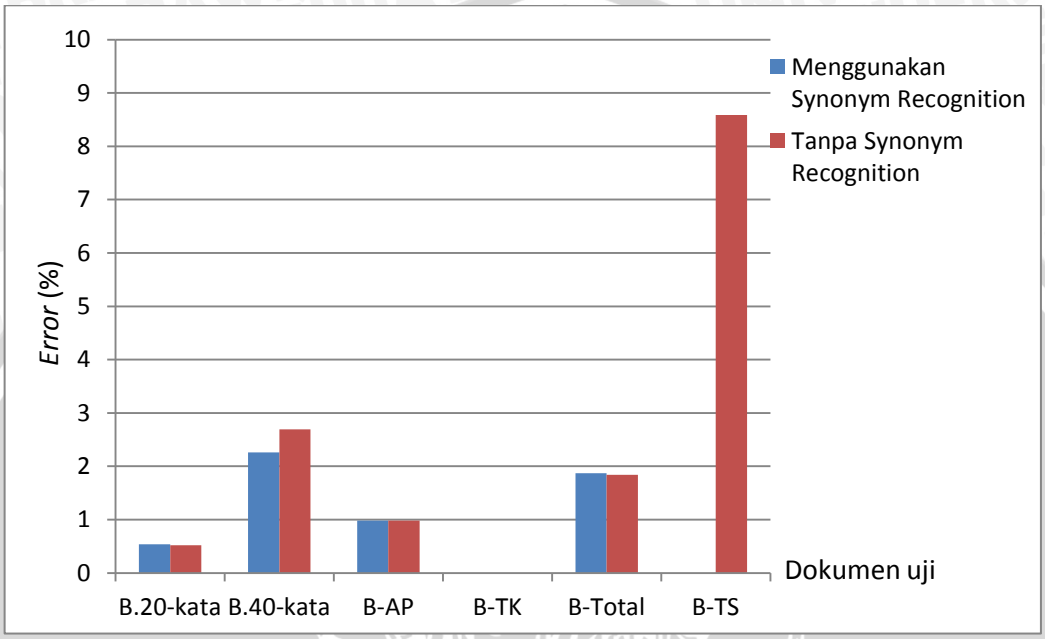
Pengujian ini dilakukan dengan membandingkan deteksi plagiarisme menggunakan *synonym recognition* dan tanpa *synonym recognition*. Grafik pengaruh *synonym recognition* pada prosentse *error* dijelaskan pada Gambar 5.4 sampai 5.6



Gambar 5.4 Grafik Pengaruh *Synonym Recognition* pada Deteksi Plagiarisme Dokumen A
Sumber: [Pengujian]

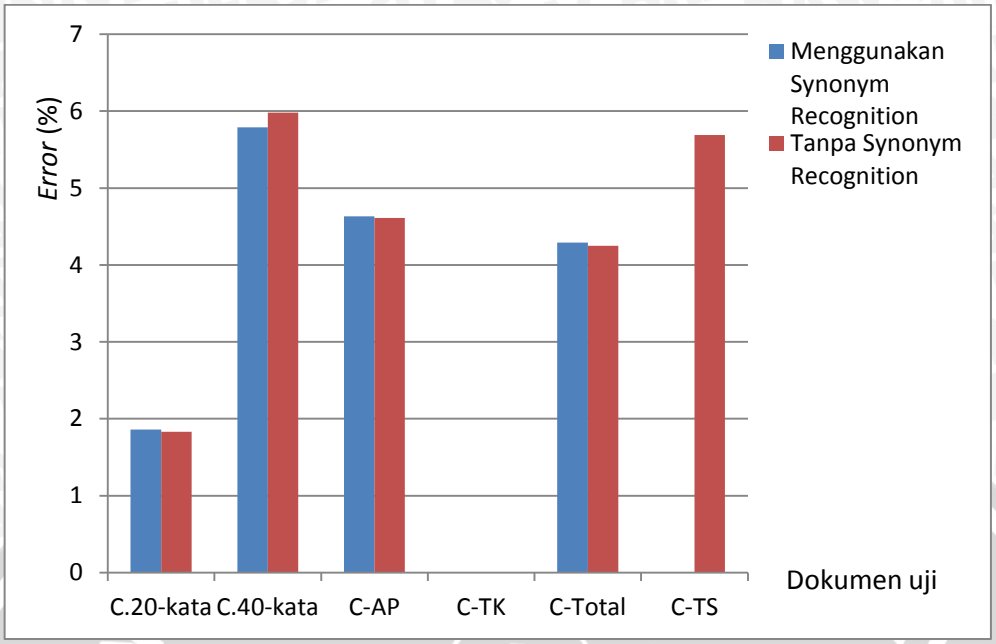
Pada grafik di atas dapat dilihat bahwa hasil *error* yang dihasilkan menggunakan *synonym recognition* lebih baik terutama pada dokumen A-TS yaitu dokumen uji yang telah mengalami perubahan dengan kata sinonimnya. Pada dokumen A-TS *error* yang dihasilkan tanpa *synonym recognition* sebesar 2,22%,

sedangkan saat menggunakan *synonym recognition error* yang dihasilkan 0%. Untuk dokumen uji lain pada dokumen asli A, hasil tanpa *synonym recognition* dan mengggunakan *synonym recognition* bernilai sama. Hal ini karena perubahan sinonim tidak ditemukan pada dokumen tersebut.



Gambar 5.5 Grafik Pengaruh *Synonym Recognition* pada Deteksi Plagiarisme Dokumen B
Sumber: [Pengujian]

Pada grafik di atas dapat dilihat bahwa hasil *error* yang dihasilkan menggunakan *synonym recognition* relatif lebih baik terutama pada dokumen A-TS yaitu dokumen uji yang telah mengalami perubahan dengan kata sinonimnya. Pada dokumen A-TS *error* yang dihasilkan tanpa *synonym recognition* sebesar 8,59%, sedangkan saat menggunakan *synonym recognition error* yang dihasilkan 0%. Untuk dokumen B.40-kata hasil mengggunakan *synonym recognition* juga lebih baik. Sedangkan pada B.20-kata dan B-Total hasil tanpa *synonym recognition* sedikit lebih baik, hal ini disebabkan karena kata hasil perubahan oleh *synonym recognition* terdapat pada dokumen tersebut yang menyebabkan bertambahnya frekuensi kata tersebut.

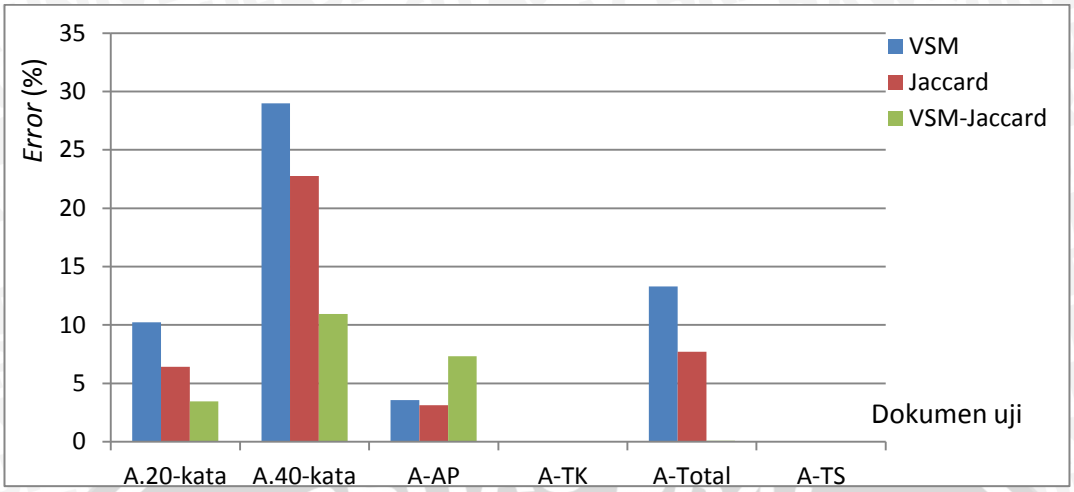


Gambar 5.6 Grafik Pengaruh *Synonym Recognition* pada Deteksi Plagiarisme Dokumen C
Sumber: [Pengujian]

Pada grafik di atas dapat dilihat bahwa hasil *error* yang dihasilkan menggunakan *synonym recognition* relatif lebih baik terutama pada dokumen A-TS yaitu dokumen uji yang telah mengalami perubahan dengan kata sinonimnya. Pada dokumen A-TS *error* yang dihasilkan tanpa *synonym recognition* sebesar 5,69%, sedangkan saat menggunakan *synonym recognition error* yang dihasilkan 0%. Untuk dokumen C.40-kata hasil mengggunakan *synonym recognition* juga lebih baik. Sedangkan pada C-AP dan C-Total tanpa *synonym recognition* sedikit lebih baik, hal ini disebabkan karena kata hasil perubahan oleh *synonym recognition* terdapat pada dokumen tersebut yang menyebabkan bertambahnya frekuensi kata tersebut.

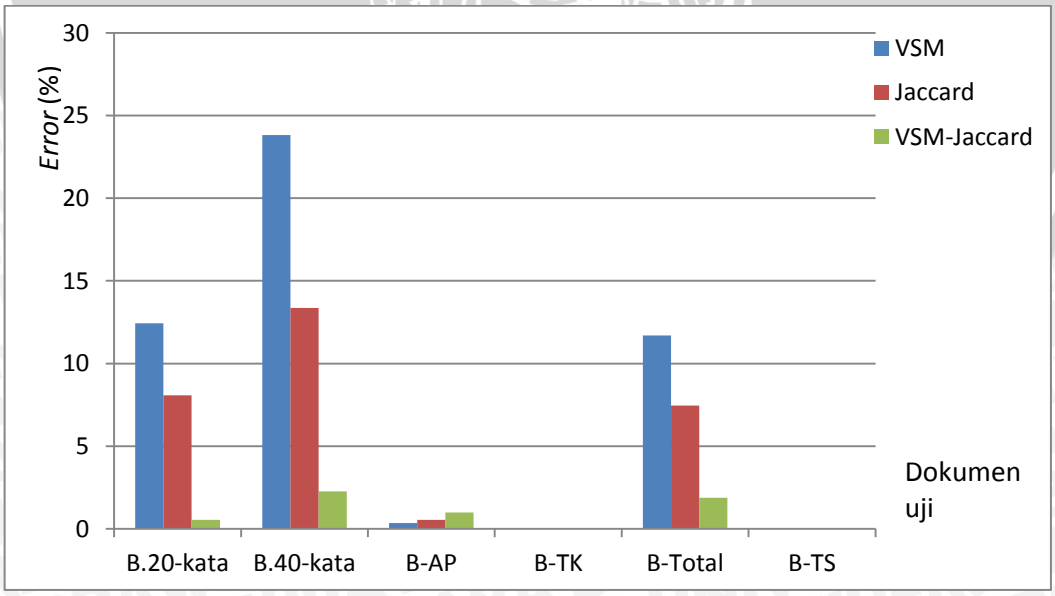
5.4.3 Analisa Hasil Uji Coba Menggunakan Algoritma VSM, Jaccard, VSM-Jaccard

Pada pengujian ini dilakukan perbandingan tingkat *error* ketiga algoritma yaitu VSM, Jaccard coefisien dan kombinasi VSM-Jaccard. Grafik perbandingan algoritma VSM, Jaccard dan VM-Jaccard dijelaskan pada Gambar 5.7 sampai 5.9



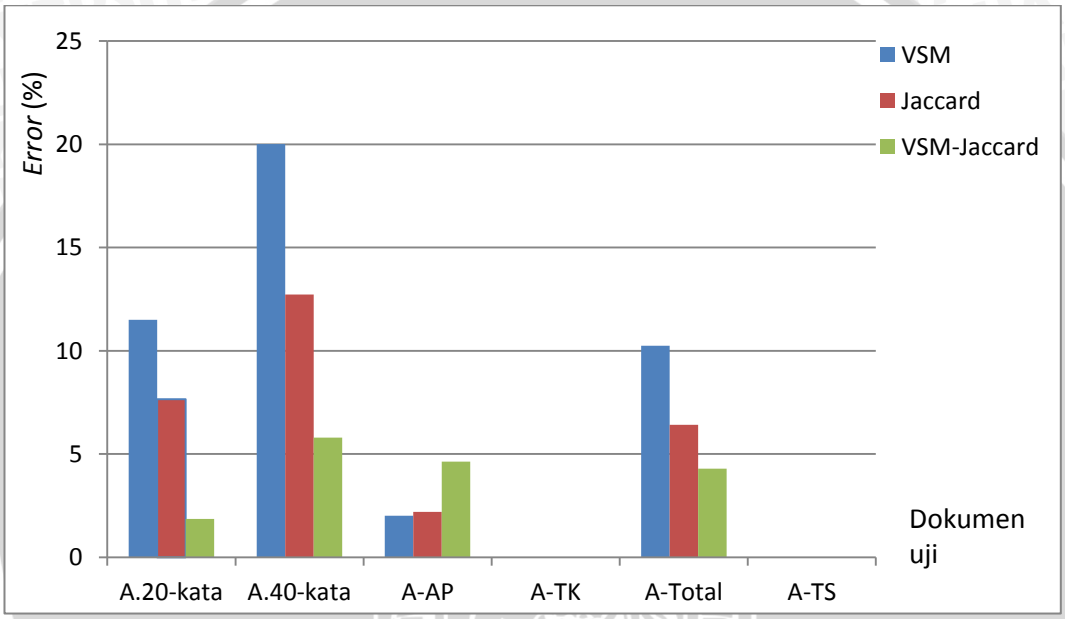
Gambar 5.7 Grafik Perbandingan *Error* Menggunakan Algoritma VSM, Jaccard dan VSM-Jaccard pada Deteksi Plagiarisme Dokumen A
Sumber: [Pengujian]

Pada grafik di atas dapat dilihat bahwa hasil kombinasi VSM dan Jaccard mampu memperkecil *error* pada dokumen A-20-kata, A.40-kata dan A-Total. Hal tersebut dikarenakan nilai α dan β dapat menurunkan *similarity* pada algoritma VSM dan Jaccard sehingga dengan nilai α dan β yang sesuai, kombinasi algoritma VSM-Jaccard dapat memperkecil nilai *error*. Sedangkan untuk dokumen A-AP terjadi kenaikan *error* hal ini disebabkan karena proses *stemming* yang belum sepenuhnya sempurna untuk merubah ke kata dasar.



Gambar 5.8 Grafik Perbandingan *Error* Menggunakan Algoritma VSM, Jaccard dan VSM-Jaccard pada Deteksi Plagiarisme dokumen B
Sumber: [Pengujian]

Pada grafik di atas dapat dilihat bahwa hasil kombinasi VSM dan Jaccard mampu memperkecil *error* pada dokumen B-20-kata, B.40-kata dan B-Total. Hal tersebut dikarenakan nilai α dan β dapat menurunkan *similarity* pada algoritma VSM dan Jaccard sehingga dengan nilai α dan β yang sesuai kombinasi algoritma VSM-Jaccard dapat memperkecil nilai *error*. Sedangkan untuk dokumen B-AP terjadi kenaikan *error* hal ini disebabkan karena proses *stemming* yang belum sepenuhnya sempurna untuk merubah ke kata dasar.



Gambar 5.9 Grafik Perbandingan *Error* Menggunakan Algoritma VSM, Jaccard dan VSM-Jaccard pada Deteksi Plagiarisme Dokumen C
Sumber: [Pengujian]

Pada grafik di atas dapat dilihat bahwa hasil kombinasi VSM dan Jaccard mampu memperkecil *error* pada dokumen C-20-kata, C.40-kata dan C-Total. Hal tersebut dikarenakan nilai α dan β dapat menurunkan *similarity* pada algoritma VSM dan Jaccard sehingga dengan nilai α dan β yang sesuai kombinasi algoritma VSM-Jaccard dapat memperkecil nilai *error*. Sedangkan untuk dokumen C-AP terjadi kenaikan *error* hal ini disebabkan karena proses *stemming* yang belum sepenuhnya sempurna untuk merubah ke kata dasar.

BAB VI
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan hasil pengujian dan pembahasan yang telah dilakukan dapat diambil kesimpulan sebagai berikut :

1. Kombinasi algoritma VSM (*cosine similarity*) dan Jaccard Coefisien dapat meningkatkan akurasi deteksi plagiarisme, hal tersebut dibuktikan dengan tingkat rata-rata *error* yang dihasilkan lebih baik daripada hanya menggunakan VSM atau Jaccard Coefisien saja. Deteksi plagiarisme menggunakan VSM (*cosine similarity*) rata-rata prosentase *error* yang dihasilkan 8,17%, Jaccard Coefisien sebesar 5,42% sedangkan kombinasi VSM-Jaccard sebesar 2,33%.
2. Kombinasi VSM-Jaccard mampu menurunkan prosentase *error* karena dengan parameter α dan β yang sesuai, algoritma ini mampu menurunkan *similarity* VSM dan Jaccard yang memiliki *error* berlebihan.
3. Penggunaan *synonym recognition* dapat menurunkan prosentase *error* pada deteksi plagiarisme, hal ini disebabkan karena *synonym recognition* mampu mengenali perubahan kata yang telah diubah ke dalam bentuk sinonimnya. Rata-rata prosentase *error* yang dihasilkan menggunakan *synonym recognition* sebesar 2,33%, sedangkan tanpa *synomym recognition* sebesar 3,27%.

6.2 Saran

Untuk penelitian lebih lanjut disarankan ditambahkan lagi variasi data pengujian. Pada penentuan parameter α dan β disarankan melakukan pengujian bilangan desimal dua angka dibelakang koma untuk didapatkan nilai α dan β yang lebih baik.

DAFTAR PUSTAKA

[ADI-13] Aditya, Christian Sri Kusuma. 2013. *Perancangan Sistem Deteksi Plagiarisme Dokumen Teks Menggunakan Algoritma Damerau Laveshtein Distance*. Universitas Brawijaya Malang.

[ANO-14] Anonim. Kasus Plagiarism Gelar Profesor akan dicopot Bobroknya Dunia Pendidikan di Indonesia Mulai Terkuak. [Online]. http://www.nahimunkar.com/kasus-plagiarism-gelar-profesor-akan-dicopot-bobroknya-dunia-pendidikan-di-indonesia-mulai-terkuak [27 Oktober 2014]

[ANZ-11] Anzelmi, Daniele. 2011. *Plagiarism Detection Based SCAM Algorithm, Proceedings of the International MultiConference of Engineers and Computer Scientist 2011, Vol.1*.

[ARI-06] Arifin, Agus Zainal dan Setiono, Ari Novan. 2003. Klasifikasi Dokumen Berita Kejadian Berbahasa Indonesia dengan Algoritma Single Pass Clustering. Institut Teknologi Sepuluh Nopember (ITS) Surabaya.

[EKB-12] Ekbal, Asif. Saha, Sriparna. Choudhary, Gaurav. 2012. *Plagiarism Detection in Text using Vector Space Model*. Department of Computer Science and Engineering Indian Institute of Technology Patna.

[GRM-11] Grman, Ján and Ravas, Rudolf. 2011. *Improved Implementation for Finding Text Similarities in Large Collections of Data*. Slovak Republic.

[KBB-97] Kamus Besar Bahasa Indonesia, 1997: 775

[KUS-09] Kustanto, Cynthia and Liem, Inggriani. 2009. *Automatic Source Code Plagiarism Detection*. School of Electrical Engineering and Informatics Bandung Institute of Technology.

[MAU-06] Maurer, Hermann. Kappe, Frank. Zaka, Bilal. 2006. *Plagiarism - A Survey*. Institute for Information Systems and Computer Media Graz University of Technology, Austria.

[MOO-05] Mooney, Raymond J. and Bunescu, Razvan. 2005. *Mining Knowledge from Text Using Information Extraction. SigKDD Explorations on Text Mining and Natural Language Processing*.

[PMP-10] Peraturan Mentri Pendidikan Nasional Nomor 17 tahun 2010: Pencegahan dan Penaggulangan Plagiat di Perguruan Tinggi

[PRA-11] Pratama, Mudafiq Riyan. Cahyono, Eko Budi dan Marthasari, Gita Indah. 2011. *Aplikasi Pendeteksi Duplikasi Dokumen Teks Bahasa Indonesia Menggunakan Algoritma Winnowing Dengan Metode K-Gram dan Synonym recognition.* Universitas Muhammadiyah Malang.

[RID-03] Ridhatillah, Ardini. 2003. *Dealing with Plagiarism in the Information System Research Community: A Look at F actors that Drive Plagiarism and Ways to Address Them, MIS Quarterly; Vol. 27, No. 4, p. 511-532*

[SAN-14] Santoso, Hadi. 2014. Deteksi Plagiarisme Dokumen Teks Menggunakan Monge Elka Distance. Universitas Brawijaya Malang.

[SOL-14] Soleman, Sidik and Purwarianti, Ayu. 2014. *Experiments on the Indonesian Plagiarism Detection using Latent Semantic Analysis*. School of Electrical Engineering and Informatics Bandung Institute of Technology.

[STE-06] Stein, B., Meyer, S. zu Eissen. 2006. *Near Similarity Search and Plagiarism Analysis, 29th Annual Conference of the German Classification Society (GfKl)*, Magdeburg, ISDN 1431-8814,pp. 430 – 437.

[TAL-03] Talla, Fadillah Z.2003. *A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia*. Universiteit van Amsterdam.

[TES-08] Tesaurus Bahasa Indonesi Pusat Bahasa Indonesia. 2008. Departemen Pendidikan Indonesia. Dendy Sugono.

[WAN-13] Wang, Shuai. Qi, Haoliang, Kong, Leilei, Du, Cuixia. 2013. *Combination of VSM and Jaccard Coefficient for External Plagiarism Detection*. China.

LAMPIRAN

Lampiran 1: Data Asli
Data Asli A

Penderita asma yang kekurangan vitamin D akan mengalami pemburukan penyakit asma.

Defisiensi vitamin ini diduga akan menghalangi reaksi terhadap pengobatan steroid yang sering dipakai dalam obat pelega dan pengontrol asma.

Dalam risetnya, para peneliti di National Jewish Health (NJH) di Denver mengukur tingkat vitamin D dari 50 penderita asma dan menilai fungsi paru-paru, hyper-responsiveness saluran udara, yang lazim terjadi di dalam pengerutan saluran udara, dan reaksi terhadap pengobatan steroid.

Ternyata, pasien yang mengalami defisiensi vitamin D memperlihatkan hasil yang buruk dalam pemeriksaan fungsi paru-paru dan hyper-responsiveness saluran udara.

Lebih dari itu, pasien yang kadar vitamin D dalam tubuhnya di bawah 30 nanogram/mL kadar hyper-responsiveness saluran udara hampir dua kali lipat, dibandingkan dengan mereka yang memiliki lebih banyak vitamin D di dalam darah mereka.

Tingkat vitamin D yang rendah juga berkaitan dengan reaksi yang lebih buruk terhadap pengobatan steroid dan peningkatan produksi sitokin pro-peradangan, TNF-alpha.

Hal itu meningkatkan kemungkinan bahwa tingkat vitamin D yang rendah berkaitan dengan peningkatan peradangan saluran udara, kata para peneliti itu.

Selain itu, kadar vitamin D meramalkan seberapa baik seseorang akan bereaksi terhadap pengobatan steroid bagi asma.

"Itu mungkin terjadi karena vitamin D bertindak sebagai pengubah reaksi steroid dengan cara yang sejalan dengan orang yang menderita asma," kata Dr E Rand Sutherland, dari divisi paru-paru dan perawatan medis kritis di NJH.

Peserta paling parah memiliki tingkat vitamin D paling rendah, kata mereka.

Asma berkaitan dengan kegemukan, dan kekurangan vitamin D mungkin menjadi faktor yang menghubungkan kedua kondisi tersebut, kata Sutherland.

"Memulihkan tingkat normal vitamin D pada orang yang menderita asma mungkin membantu meningkatkan asma mereka," kata Sutherland.

Namun, tidak diketahui apakah asupan vitamin D akan membantu penderita asma, katanya.

Asupan vitamin D buat orang dewasa yang disarankan ialah 300 IU sampai 600 IU, tergantung pada usia, kata US National Institutes of Health.

Data Asli B

Kegiatan olahraga berfungsi membakar kalori agar kalori berlebihan tidak menjadi lemak dalam tubuh.

Latihan teratur dengan intensitas sedang sampai berat berperan penting dalam mencegah berbagai penyakit dan membantu memerangi pengaruh merugikan dari kelebihan berat badan.

Olahraga aerobik, minimal 30 menit setiap hari dan dilakukan secara teratur, telah terbukti dapat membakar lemak dan mengencangkan otot-otot.

Itu sebabnya latihan aerobik harus jadi bagian dari strategi untuk memangkas kelebihan berat badan.

Kendati begitu, apabila kita menambah latihan aerobik dengan latihan membangun otot, seperti mengangkat beban, hasilnya akan lebih baik.

"Jaringan otot memerlukan kalori lebih banyak," kata Janet Walberg Ranin, PhD, profesor di Exercise Science Program di Virginia Polytechnic Institute, Amerika Serikat.

Jika kita meningkatkan massa otot sambil mengurangi lemak, berarti kita meningkatkan kemampuan pembakaran.

"Sampai petang hari, ketika Anda duduk dalam rapat atau membaca, otot-otot Anda yang baru terus bekerja membakar kalori," kata Ranin.

Latihan daya tahan tidak hanya sebatas mengangkat barbel.

Jika Anda ingin betul-betul melatih kelompok otot yang lain, cara yang terbaik adalah pergi ke pusat kebugaran dan meminta pelatih menunjukkan latihan-latihan yang tepat.

Di gym juga biasanya terdapat berbagai alat untuk menggunakan otot leher, lengan, dada, atau kaki.

Jika sudah tahu, Anda bisa mengerjakan latihan serupa di rumah.
Pilihlah latihan teratur yang cukup intens, sering, dan dalam jangka waktu yang cukup dengan jenis latihan yang tepat secara bertahap.
Semakin sering Anda mengerjakannya, makin cepat peningkatan massa otot Anda.
Dengan demikian, makin banyak pula lemak yang terbakar.

Data Asli C

PENYAKIT kanker sudah menjadi momok, terutama bagi perempuan.
Upaya pencegahan penyakit itu pun kini makin diperhatikan.
Perilaku hidup merupakan salah satu penyebab utama timbulnya kanker, di luar faktor keturunan, lingkungan dan pemanfaatan pelayanan kesehatan.
Antara lain lewat pola makan sehari-hari.
Tak percaya? Baca indikasi para ahli berikut.
"80 - 90% dari berbagai bentuk kanker berkaitan erat dengan makanan yang sehari-hari kita konsumsi".
Nah, kalau Anda tak mau peduli, kanker payudara, kanker tenggorokan, kanker lambung, kanker kolon (usus besar) dan rektum, kanker hati hingga kanker paru-paru dan kandung kencing, siap menghampiri Anda.
Artinya, kantong Anda pun ikut-ikutan kering.
Tapi tak usah khawatir.
Ada beberapa langkah untuk mencegah kedatangan kanker.
Perlu perhatian dan penerapan perilaku hidup sehat.
Berikut ini langkah-langkah yang bisa Anda coba.
Pertama, makanlah sayur-sayuran, buah-buahan, biji-bijian seperti tempe, tahu, dan makanan yang mengandung banyak serat.
Sayur-sayuran dan buah-buahan yang banyak mengandung vitamin A, betakaroten, vitamin C, vitamin E, serta zat gizi seperti selenium (Se), asam folat, niasin (vitamin B3), vitamin D, seng (zinc), kalsium (Ca), dan magnesium (Mg), dikenal sebagai zat antikarsinogen, yaitu zat yang bersifat protektif (melindungi dari timbulnya kanker).
Sedangkan serat yang merupakan bagian dari pangan nabati berperan penting

dalam pemeliharaan kesehatan tubuh dan dapat mencegah timbulnya kanker kolon. Sebaiknya, Anda mengonsumsi sayuran hijau dan buah-buahan paling tidak satu atau dua kali.

Lampiran 2: Kamus kata dasar, Kamus *Stopword*, Kamus Sinonim
Kamus Kata Dasar

badan	kembali	masuk	rutin
bank	kenapa	menit	saat
bantu	kencang	mereka	salah
bantu	kencing	milik	saran
beli	kendati	minimum	sebab
berat	kering	minta	sebab
berat	kerja	mungkin	sedang
beri	ketik	nilai	sederhana
besar	ketika	pangan	sehat
betul	khawatir	pangkas	seperti
buruk	kira	peduli	serat
cegah	konsumsi	pemain	seri
cemas	kritis	pengaruh	serikat
cepat	kuningan	penting	setiabudi
control	kurang	penyakit	sila
demi	kurang	peran	susah
dewasa	kurang	perang	tahan
elit	lancar	percaya	tahap
gemuk	langkah	pergi	tambah
guna	lanjut	periksa	telah
halang	latih	perlu	teliti
ingin	layanan	perlu	tenggokan
jalan	lengan	pertama	tersebut
jalan	makan	petang	tingkat
jos	makan	prediksi	tingkat

kadang	makin	puas	tolong
kait	manfaat	pulih	tujuan
kandung	manfaat	pulih	ubah
kelompok	mantab	reaksi	usah
kemang	masalah	restoran	vitamin

Kamus *Stopword*

ada	mbak
agar	melulu
akan	mu
anda	ni
atau	nih
beberapa	nya
berikut	pada
dahulu	pasti
dalam	pernah
dan	pun
dari	saja
deh	sangat
dengan	saya
di	segera
dong	sekali
eh	sekarang
harus	selalu
ini	semua
itu	sering
jadi	serius
jika	setiap
kah	sih
kali	sudah
kalian	supaya
kan	tadi

karena	tahu
ke	tahun
kemarin	tapi
kita	telah
kok	terhadap
ku	terlalu
lagi	terus
lah	tu
lama	tuh
lho	untuk
loh	ya
mah	yah
mas	yang
masih	

Kamus Sinonim

abad=era	asumsi=hipotesis
abah=arah	asusila=amoral
abang=kakak	atur=rutin
abnormal=ajaib	awak=badan
absah=asli	awal=mula
acara=agenda	awas=waspada
adab=adat	bagai=seperti
adegan=babak	bakteri=kuman
adu=kompetisi	bakti=hormat
afiliasi=aliansi	bangkit=bangun
agama=akidah	bantu=tolong
agresif=nafsu	baru=anyar
ahli=andal	basi=busuk
aib=hina	bebas=leluasa
ajal=maut	bedah=operasi
ajang=arena	benar=betul

akhlak=etika	bungkam=diam
akomodasi=mudah	cemas=khawatir
akrab=dekat	fungsi=tujuan
aksara=abjad	halangi=cegah
aksesoris=hiasan	ketika=saat
alim=pandai	langkah=tahap
almarhum=mendiang	minimal=minimum
almarhumah=mendiang	petang=malam
amal=jasa	pulih=sehat
ambisi=hasrat	ramal=prediksi
angka=bilangan	usah=perlu
angker=seram	usia=umur