Sistem Deteksi Plagiarisme Dokumen Karya Ilmiah dengan Algoritma Pencocokan Pola

Anton Yudhana[1], Abdul Djalil Djayali[*2], Sunardi[3]
[1,3]Teknik Elekro, Universitas Ahmad Dahlan
[2]Magister Teknologi Informasi Universitas Ahmad Dahlan
e-mail: [*2]**jlcnate.zit@gmail.com**

Abstrak

Plagiarisme secara konvensional dipandang sebagai pelanggaran ilmiah yang serius, pencurian terhadap ide-ide di pasar intelektual kompetitif. Kasus plagiarisme memiliki jumlah yang sangat besar di dalam sebuah lembaga, termasuk penulis yang tidak diketahui dan atribusi kepenulisan untuk para elit birokrasi. Ada banyak usaha yang dilakukan didalam sebuah lembaga untuk mengurangi stigma plagiarisme yang semakin kompetitif. Perkembangan internet yang intensif sekarang ini pun menyakibatkan ketersediaan akan jumlah informasi yang dapat diperoleh. Hal ini memungkinkan seseorang melakukan plagiarisme dari sebuah karya. Jadi menurut kamus besar Bahasa Indonesia (KBBI) plagiarisme berarti memetik atau menulis sebagian besar atau teks yang dimiliki oleh orang lain, menjiplak atau meniru tulisan dari karya orang lain, mencuri skripsi orang lain dan mengakui sebagai milik sendiri, mengutip karangan orang lain tanpa izin dari penulis. Penelitian ini bertujuan untuk merancang aplikasi untuk mendeteksi plagiarisme menggunakan algoritma Rabin-Karp. Metode penelitian yang digunakan adalah studi pustaka dan dalam pengembangan sistem menggunakan metode prototipe. Aplikasi deteksi plagiarisme dihasilkan untuk sistem pelaporan berbasis web. Aplikasi yang dibangun berhasil mendeteksi kalimat yang sama pada judul, abtraksi dan kata kuci serta antara file terbanding dengan file yang ada di dalam repositori.

Kata kunci—*Plagiarisme, rabin-karp, similaritas, hashing.*

1. PENDAHULUAN

Plagiarisme ternyata tidak hanya menjangkiti negara yang sedang berkembang seperti Indonesia. Beberapa kasus terakhir bahkan dijumpai di negara maju seperti Amerika Serikat [21]. Bedanya, negara maju menetapkan sanksi yang tidak main-main dengan plagiarisme, di saat Indonesia masih terkesan malu-malu untuk menjatuhkan sanksi tegas dikarenakan sebagian besar karya ilmiah belum dilindungi Undang-Undang Hak atas Kekayaan Intelektual (HaKI) maka plagiarisme digolongkan sebagai kejahatan akademik yang termasuk sebagai pelanggaran etika dan sulit untuk dipidanakan. Sebagai langkah awal untuk mencegah kasus serupa diperlukan cara mendeteksi kemungkinan terjadinya penjiplakan seperti di lingkungan perguruan tinggi yaitu utamanya pada hasil tugas akhir calon sarjana S1 maupun tesis calon sarjana S2 dan disertasi calon sarjana S3 yang rawan penjiplakan [20]. Penulis akan memaparkan hasil analisis pendekatan atau metode yang ada untuk mendeteksi plagiarisme dokumen karya ilmiah. Pendekatan atau metode yang dipaparkan adalah dengan menggunakan string matching algoritma Rabin-Karp.

Untuk meminimalisasi praktik plagiarisme, diperlukan pendeteksian terhadap

penulisan. Untuk mengatasi praktik plagiarisme, tidaklah cukup hanya mengingatkan kepada mahasiswa bahwa tindakan plagiarisme tidak baik dilakukan. Pendeteksian praktik plagiarisme merupakan solusi yang sebaiknya dilakukan sehingga tindakan curang tersebut dapat diminimalisasi. Namun, proses pendeteksian secara manual sulit untuk dilakukan karena jumlah penulisan yang banyak. Sehingga diperlukan sistem untuk mendeteksi plagiarisme. Metode untuk mendeteksi plagiarisme dapat di klasifikasikan menjadi tiga metode yaitu metode perbandingan teks lengkap, metode dokumen fingerprinting dan metode kesamaan kata kunci.

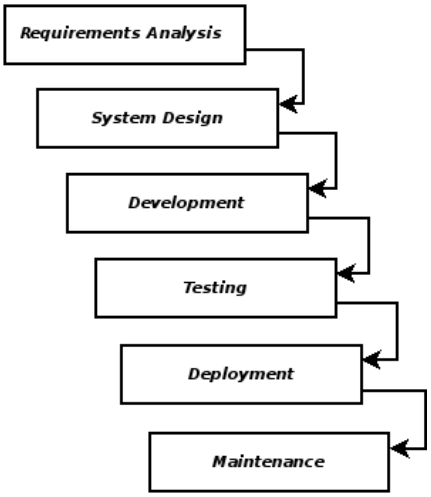
2. METODE PENELITIAN

2.1 Pengumpulan Data

Penelitian yang dilakukan dalam penelitian ini termasuk dalam bidang rekayasa software. Metode pengumpulan data dilakukan bersama dengan pengembangan sistem sekuensial linier atau yang sering disebut juga dengan siklus kehidupan klasik atau model air terjun (waterfall model) memberikan sebuah pendekatan pengambangan sistem yang sistematik dan sekuansial, dimulai pada Studi literatur, analisis dan perancangan, implementasi, uji coba sistem, dan evaluasi dan analisa sistem. [17]

2.2 Pengembangan Perangkat Lunak

Metodologi yang digunakan dalam pengembangan perangkat lunak adalah dengan menggunakan paradigma Waterfall. Model ini merupakan sebuah pendekatan terhadap pengembangan perangkat lunak yang sistematik, dengan beberapa tahapan, yaitu: Requirements Analysis, Design, Development, Testing dan Maintenance. [8]



Gambar 1 Paradigma Waterfall
(*SoftwareDevelopment Life Cycle* (SDLC))

Dari gambar 2 diatas dapat dijelaskan bahwa:

1) Requirements Analysis, merupakan tahapan dimana System Engineering menganalisis segala hal yang ada pada pembuatan proyek atau pengembangan perangkat lunak yang bertujuan untuk memahami sistem yang ada, mengidentifikasi masalah dan mencari solusinya.

2) Design, tahapan ini merupakan tahap penerjemah dari keperluan atau data yang telah dianalisis ke dalam bentuk yang mudah dimengerti oleh pemakai (user).
3) Development, yaitu menerjemahkan data yang dirancang ke dalam bahasa pemrograman yang telah ditentukan.
4) Testing, merupakan uji coba terhadap sistem atau perangkat lunak setelah selesai dibuat.
5) Deployment, yaitu merapakan dan uji coba sistem pada komputer yang dijadikan server.
6) Maintenance, yaitu penerapan sistem secara keseluruhan disertai pemeliharaan jika terjadi perubahan struktur, baik dari segi software maupun hardware.

2.3 Penentuan Nilai Similaritas

Ada tiga macam teknik yang dibangun untuk menentukan nilai similarity (kemiripan) dokumen. Antara lain: [19]

1) Distance-based similarity measure, yaitu mengukur tingkat kesamaan dua buah objek dari segi jarak geometris dari variabel-variabel yang tercangkup di dalam kedua objek tersebut. Metode Distance-based Similarity ini meliputi Minkowski Distance, Manhattan/City Block Distance, Euclidean Distance, Jaccard Distance, Dice's Coefficient, Cosine Similarity, Levenshtein Distance, Hamming Distance, dan Soundex Distance.
2) Feature-based similarity measure, yaitu melakukan perhitungan tingkat kemiripan dengan merepresentasikan objek ke dalam bentuk fitur-fitur yang ingin diperbandingkan. Feature-based similarity banyak digunakan dalam melakukan pengklasifikasian atau pattern matching untuk gambar dan teks.
3) Probabilistic-based similarity measure, yaitu menghitung tingkat kemiripan dua objek dengan merepresentasikan dua set objek yang dibandingkan dalam bentuk probability. Meliputi Kullback Leibler Distance dan Posterior Probability.

Algoritma Rabin-Karp didasarkan pada fakta jika dua buah string sama maka harga hash value-nya pasti sama. Akan tetapi ada dua masalah yang timbul dari hal ini, masalah pertama yaitu ada begitu banyak string yang berbeda, permasalahan ini dapat dipecahkan dengan meng-assign beberapa string dengan hash value yang sama [13]. Masalah yang kedua belum tentu string yang mempunyai hash value yang sama cocok untuk mengatasinya maka untuk setiap string yang di-assign dilakukan pencocokan string secara Brute-Force [1]. Kunci agar algoritma Rabin-Karp efisien, terdapat pada pemilihan hash value-nya. Salah satu cara yang terkenal dan efektif adalah memperlakukan setiap substring sebagai suatu bilangan dengan basis tertentu. Terdapat empat kategori proses perbandingan yaitu: [13]

1) Dari kanan ke kiri
2) Dari kiri ke kanan
3) Dalam order spesifik
4) Dalam order apapun

Berdasarkan keempat kategori tersebut algoritma Rabin-Karp termasuk dalam kategori dari kiri ke kanan. Algoritma Rabin-Karp menerapkan fungsi hash yang menyediakan metode sederhana untuk mencegah kompleksitas waktu O(m2). Fungsi hash setidaknya harus menyediakan empat properti yaitu:

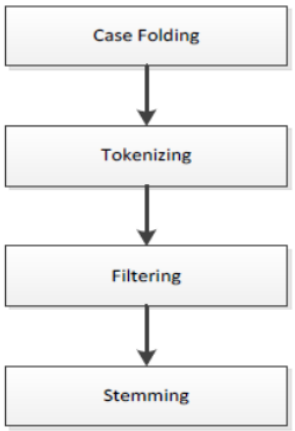
1) Mampu melakukan komputasi secara efisien
2) Diskriminasi string yang tinggi

3) Fungsi hash (s[i+1 ... i+m] = s[i ... i+m-1] - s[i] + s[i+m]) harus mudah mengkomputasi dari:
   a) Hash (s[i ... i+m-1])
   b) Hash (s[i])
   c) Hash (s[i+m])
4) Algoritma Rabin-Karp menandai langkah-langkah berikut ini:
   a) Menerapkan fungsi hash
   b) Fase preproses dalam kompleksitas waktu O(m) dan waktu yang konstan
   c) Fase pencarian dalam kompleksitas waktu O(m)
5) O(n+m) mengestimasi waktu aktif

Algoritma Rabin-Karp adalah algoritma pencocokan string yang menggunakan fungsi hash sebagai perbandingan antara string yang dicari (m) dengan substring pada teks (n). Apabila hash value keduanya sama maka akan dilakukan perbandingan sekali lagi terhadap karakter-karakternya. Apabila hasil keduanya tidak sama, maka substring akan bergeser ke kanan. Pergeseran dilakukan sebanyak (n-m) kali. Perhitungan nilai hash yang efisien pada saat pergeseran akan mempengaruhi performa dari algoritma ini.[6]

2.4 *Preprocessing*

Pada preprocessing, *Rabin-Karp* menggunakan teori bilangan dalam perhitungan *hash key*, antara lain menggunakan aturan *Horner* dan aturan modulus. Preprocessing bertujuan agar teks dapat diubah menjadi lebih terstruktur dan menghilangkan noise pada dokumen. Proses preprocessing tersebut meliputi *case folding*, *tokenizing*, *filtering*, dan *stemming*. [17] Gambar 2 berikut menunjukkan tahapan-tahapan pada *text preprocessing*.



Gambar 2 Tahapan *Text Preprocessing*

2. 4.1 *Case Folding*

Case folding merupakan proses pertama dari rangkaian preprocessing dokumen. Dalam proses ini akan dilakukan perubahan pada kata-kata dalam dokumen menjadi huruf kecil (a sampai z). [6]

2. 4.2 *Tokenizing*

Tokenizing merupakan tahapan dimana dilakukannya pemotongan terhadap string input berdasarkan atas delimiter yang telah ditentukan. Karakter selain huruf akan dianggap sebagai delimiter dan akan dihilangkan atau dihapus untuk proses mendapat kata-kata penyusun teks.

Dari proses ini akan dihasilkan kata-kata penyusun string atau teks atau yang sering disebut token atau term. [6]

2. 4.3 *Filtering*

Filtering merupakan tahap pengambilan kata-kata penting dari hasil *tokenizing string*. *Filtering* dilakukan dengan membuang kata-kata yang telah terdaftar ke dalam stop-word atau stop-list. Stop-word adalah kata-kata yang sering muncul dalam teks dalam jumlah besar dan dianggap tidak memiliki makna penting. [17]

2. 4.4 *Stemming*

Stemming merupakan proses yang dilakukan untuk mendapatkan kata dasar dari suatu kata. Stemming Nazief-Adriani merupakan suatu algoritma stemming yang dibuat oleh Bobby Nazief dan Mirna Adriani. [15]

2. 5 Pengukuran Nilai *Similarity*

Mengukur similarity (kemiripan) dan jarak antara dua entitas informasi merupakan syarat utama pada penemuan informasi. Tahap pertama, membagi kata menjadi k-grams. Kedua, mengelompokkan hasil term dari k-grams yang sama. Kemudian untuk menghitung similarity dari kumpulan kata tersebut maka digunakan rumus *Dice's Similarity Coefficient* untuk pasangan kata yang digunakan. [10]

2. 6 Persentasi Nilai Similarity

Untuk menentukan nilai similarity antara dokumen yang diuji ada 5 jenis penilaian presentase similarity. [13]

1) 0% : hasil uji 0% berarti kedua dokumen tersebut benar-benar berbeda baik dari segi isi dan kalimat secara keseluruhan.
2) < 15% : hasil uji kurang dari 15% berarti kedua dokumen tersebut hanya mempunyai sedikit kesamaan.
3) 15 - 50% : hasil uji 15-50% berarti menandakan dokumen tersebut termasuk plagiat tingkat sedang.
4) > 50% : hasil uji lebih dari 50% berarti dapat dikatakan bahwa dokumen tersebut mendeteksi plagiarisme.
5) 100% : hasil uji 100% menandakan bahwa dokumen tersebut adalah plagiat karena dari awal sampai akhir mempunyai isi yang sama persis.

3. HASIL DAN PEMBAHASAN

Aplikasi ini merupakan aplikasi berbasis web sehingga dalam pelaksanaannya memerlukan web server dengan sistem operasi GNU/Linux, selain itu dibutuhkan pula Python yang digunakan sebagai module dalam extraksi file PDF dengan PDFMiner, Ruby sebagai library dari K-gram, PHP engine sebagai pengolahan dan MySQL yang digunakan sebagai database sistem. Pada saat tampilan awal aplikasi dipilih salah satu metode pendeteksian, yaitu pedeteksian dengan menggunakan judul, pada tabel 1 berikut menunjukkan detail dari isi perbandingan dari file :

Tabel 1 Text Pengujian

Text 1	Berisi Text1
Text 2	Isi dari Text 2

Proses pertama, proses persiapan dilakukan proses tokenizing, filtering dan stemming hasil proses ditunjukkan pada tabel 2 berikut:

Tabel 2 Hasil *Tokenizing*, *Filtering* dan *Stemming*

Text 1	berisitext1
Text 2	isitext2

Proses kedua, proses parsing K-gram dengan panjang K = 4. Hasil parsing K-gram. Hasil dari proses ini ditunjukkan pada tabel 3 berikut:

Tabel 3 Hasil *Parsing* K-gram = 4

No.	Parsing Teks 1	Parsing Teks 2
1	beri	isit
2	eris	site
3	risi	itex
4	isit	text
5	site	ext2
6	itex	
7	text	
8	ext1	

Berikut adalah perhitungan hashing dengan merubah char menjadi desimal berdasarkan ASCII dengan K-gram = 4 dan Modulo = 101. Hasil dari perhitungan hashing ini ditunujukkan pada tabel 4.

- Pattern = ‘beri’
- Hashing = 98 * 103 + 101 * 102 + 114 * 101 + 105 * 100 = 109345 mod 101 = 63
- Remainder = 109345/101 = 1082.623762 = 109345
- Dan seterusnya.

Tabel 4 Hasil Perhitungan dengan K-gram dan Modulo

No.	Teks 1			Teks 2		
	Parsing	Hashmod	Remainder	Parsing	Hashmod	Remainder
1	beri	63	109345	isit	1	117666
2	eris	41	113565	site	6	126761
3	risi	10	125755	itex	65	117730
4	isit	1	117666	text	55	127416
5	site	6	126761	ext2	80	114210
6	itex	65	117730			
7	text	55	127416			
8	ext1	79	114209			

Proses ketiga yang ditunjukkan pada tabel 5 berikut adalah hasil perhitungan nilai-nilai yang ada pada tabel 4 yang dicocokkan menggunakan string matching dengan mengambil nilai match yes.

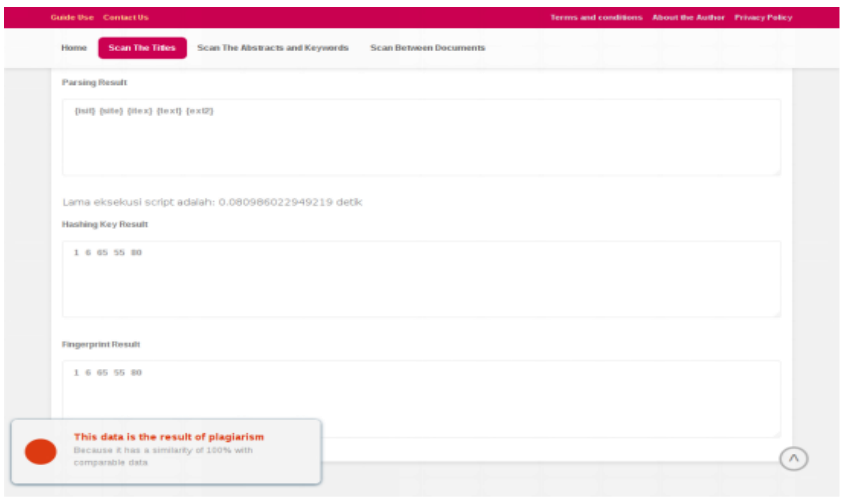
Tabel 4 Hasil Perhitungan dengan K-Gram dan Modulo

No.	Teks 1			Teks 2			Match
	Parsing	Hashmod	Remainder	Parsing	Hashmod	Remainder	
1	beri	63	109345	isit	1	117666	
2	eris	41	113565	site	6	126761	
3	risi	10	125755	itex	65	117730	Yes
4	isit	1	117666	text	55	127416	Yes
5	site	6	126761	ext2	80	114210	
6	itex	65	117730				
7	text	55	127416				
8	ext1	79	114209				

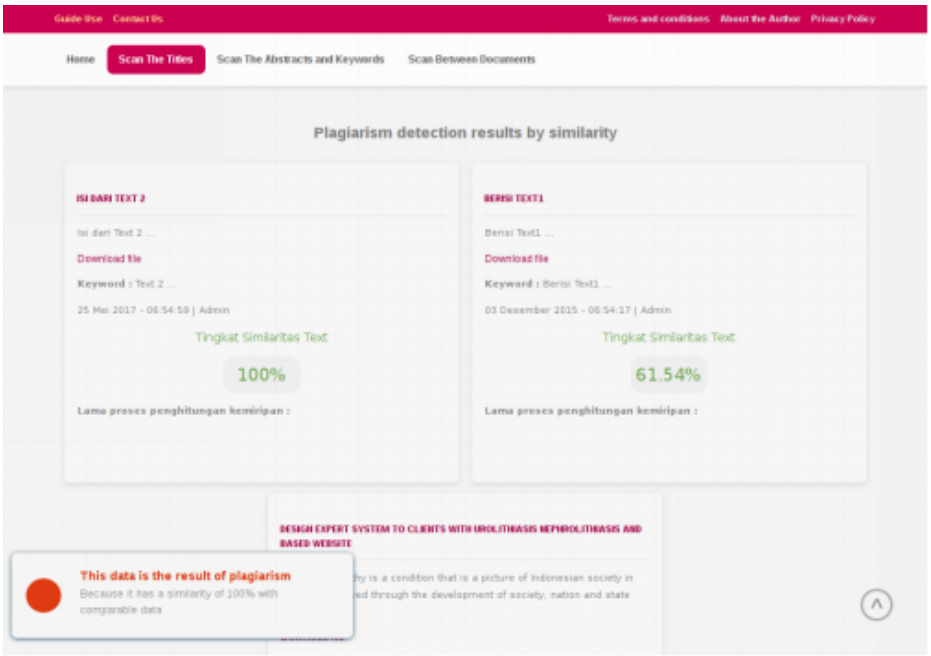
Proses keempat, untuk mendapatkan informasi tingkat similarity dilakukan pembobotan menggunakan Dice's Similarity Coefficient. [16]

P Similarity = ((4*2)/(8+5))*100%
= (8/13)*100%
= 61.53846154% = 61.54%

Dapat disimpulkan perbandingan Teks 1 dan Teks 2 memiliki kemiripan 61.54% dan dapat dikatakan bahwa dokumen tersebut mendeteksi plagiarisme. Dengan waktu yang dibutuhkan dalam membandingkan text1 dan text2 adalah 0.08 detik. Pengujian terhadap sistem menghasilkan output seperti pada gambar 3 dan gambar 4 berikut ini :



Gambar 3 Hasil Parsing, Hashing Key dan Fingerprint



Gambar 4 Notifikasi dan hasil persentase antara dua file text

4. KESIMPULAN

Berdasarkan rangkaian pengujian dan penilaian yang telah dilakukan maka dapat disimpulkan bahwa:

1) Sistem dapat memberikan sebuah nilai kebenaran dari data karya ilmiah dengan menggunakan parsing k-gram dan hashing dalam menemukan kecocokan kata atau kalimat yang sama pada jawaban dan kunci jawaban.
2) Algoritma Rabin-Karp modifikasi waktu proses similaritas (running time) dengan baik.
3) Sistem melakukan pengecekan terhadap judul karya ilmiah, abstraksi atau dokumen terbanding dengan dokumen pembanding yang terdapat pada database dengan akurat.
4) Sistem pengecekan pada tingkat similaritas dokumen dengan algoritma Rabin-Karb ini memberikan hasil berupa persentase similaritas.

5. SARAN

Penulis perharap, semoga peneliti selanjutnya dapat mengembangkan sistem yang penggunaan waktunya lebih efisien dan optimal, tentu saja dengan keakuratan yang maksimal. Tidak hanya dokumen PDF saja namun dokumen dengan format lainnya. Mungkin dapat menggunakan OCR atau penggabungan dengan pengolahan citra sehingga gambar maupun tabel yang terdapat pada sebuah dokumen uji dapat di ukur pula similaritasnya.

DAFTAR PUSTAKA

[1] Abdeen, Ali., Rawan. (2011). An Algorithm for String Searching Based on Brute-Force Algorithm, International Journal of Computer Science and Network Security, Vol.11 No.7.

[2] Andres, Christopher, Saloko. (2006). Penelaan Algoritma Rabin-Karb dan Perbandingan Peosesnya dengan Algoritma Knut-Morris-Pratt, Departemen Teknik Informatika, Institut Teknologi Bandung.

[3] Anzelmi, Daniele., et. al. (2011). Plagiarism Detection Based SCAM Algorithm, Proceedings of the International Multi Conference of Engineers and Computer Scientist 2011, Vol.1.

[4] Deddy Winarsono. (2012), Daniel O Siahaan, Umi Yuhana, Sistem Penilaian Otomatis Kemiripan Kalimat Menggunakan Syntatics-Semantic Similarity Pada Sistem E-Learning, Jurnal Ilmiah KURSOR Menuju Solusi Teknologi Informasi, Vol. 5, No. 2.

[5] Dreher, Heinz. (2007). Automatic Conceptual Analysis for Plagiarism Detection, Issues in Informing Science and Information Technology Volume 4.

[6] Firdaus, Bagus. (2008). Deteksi Plagiat Dokumen Menggunakan Algoritma Rabin- Karp, Makalah If2251 Strategi Algoritmik Tahun 2008

[7] Gipp, Bela, et. al. (2011), Citation Pattern Matching Algorithms for Citation-based Plagiarism Detection: Greedy Citation Tiling, Citation Chunking and Longest Common Citation Sequence, scholar.google.com, diakses 07 November 2015.

[8] Iwan Binanto. (2014). Analisa Metode Classic Life Cycle (waterfall) Untuk Pengembangan Perangkat Lunak Multimedia. Universitas Sanata Dharma, Yogyakarta, Indonesia.

[9] Jain, Shivani., Rao, Nersimha, A.L., Agarwal, Pankaj. (2007). A Relative Study of Pattern Matching Algorithms, Journal of Computing Technologies, Vol.2 Issue 1.

[10] Kosinov, Serhiy. 2001. Evaluation of n-grams conflation approach in text based information retrieval. Unpublished journal. Computing Science Department, University of Alberta, Canada.

[11] Lukashenko, Romans., et. al. (2007). Computer-Based Plagiarism Detection Methods and Tools: An Overview, International Conference on Computer Systems and Technologies – CompSysTech'07.

[12] Martin, Brian, (1994), Plagiarism: a misplaced emphasis, Journal of Information Ethics, Vol. 3, No. 2.

[13] Mutiara, Benny. A, Agustina., Sinta. (2008). Anti Plagiarism Application with Algorithm Karp-Rabin at Thesis in Gunadharma University, Gunadharma University, Jakarta.

[14] Nanda Zanniba Harisma. (2008). Implementasi Sistem Penilaian Esai Otomatis Metode LSA Dengan Tiga Bobot Kata Kunci, Universitas Indonesia.

[15] Pramudita. (2014). Penerapan Algoritma Stemming Nazief & Adriani dan Similarity pada Penerimaan Judul Thesis, Jurnal DASI Vol. 14, No. 4.

[16] Rizqi Bayu Aji P, ZK. Abdurrahman Baizal, Yaunar Firdaus. (2011). Automatic Essay Grading System Menggunakan Metode Latent Semantic Analysis, Seminar Nasional Aplikasi Teknologi Informasi (SNATI).

[17] Sahriar Hamza, M. Sarosa, Purnomo Budi Santoso. (2013). Sistem Koreksi Soal Essay Otomatis Dengan Menggunakan Metode Rabin Karp, Jurnal EECCIS Vol. 7, No. 2.

[18] Oktavianti. (2012). Cegah Plagiarisme, Dosen Darmajaya Bentuk Sistem Pendeteksi Plagiarisme Multi Bahasa, Informatics & Business Institute Darmajaya Lampung, diambil dari : http://www.darmajaya.ac.id/id/cegah-plagiarisme-dosen-darmajaya-bentuk-sistem-pendeteksi-plagiarisme-multi-bahasa/

[19] Zaka, Bilal ; Maurer, Hermann (2007), "Plagiarism - A Problem And How To Fight It", Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2007, AACE, pp. 4451–4458

[20] Dian. (2012). Aplikasi pendeteksian plagiat pada karya ilmiah menggunakan algoritma Rabin-Karb. Laporan Penelitian Pengembangan Fakultas dan Keilmuan. Universitas Negeri Gorontalo.

[21] Rahma. (2011). Plagiarisme di Dunia Akademik. Makalah Ilmu Sosial dan Budaya. Universitas Negeri Padang