ISSN 2579 7247

PERANCANGAN APLIKASI DETEKSI PLAGIARISME KARYA ILMIAH MENGGUNAKAN ALGORITMA *WINNOWING*

Sunardi[1)], Anton Yudhana [2)], Iif Alfiatul Mukaromah[3)]

[1,2]Program Studi Teknik Elektrom Universitas Ahmad Dahlan
[3]Program Studi Magister Teknik Informatika Universitas Ahmad Dahlan
[1,2,3]Kampus 3 UAD Jl. Prof Soepomo, Janturan, Yogyakarta, 55164
E-mail : sunargm@gmail.com[1)], eyudhana@gmail.com[2)], iifam1604@gmail.com[3)]

ABSTRAK

Internet merupakan salah satu teknologi yang mengalami perkembangan sangat pesat. *Internet* membuat manusia lebih mudah dalam mendapatkan informasi tentang apa yang dicari, namun demikian sisi negatifnya banyak yang sekedar *copy-paste* terutama dalam pembuatan karya ilmiah. Hal ini disebabkan kurangnya minat baca dan terbatasnya waktu sehingga terdorong untuk melakukan tindakan plagiarisme. Plagiarisme merupakan penjiplakan yang dilakukan terhadap karya orang lain dan sudah menjadi hal yang biasa yang dilakukan oleh akademisi khususnya mahasiswa dalam menyelesaikan tugas akhir. Untuk meminimalisir praktik plagiarisme, diperlukan pendeteksian terhadap penulisan karya ilmiah. Pada penelitian ini dilakukan perancangan aplikasi deteksi plagiarisme menggunakan algoritma *winnowing.* Algoritma *winnowing* merupakan salah satu metode *document fingerprinting* yang digunakan untuk mendeteksi kemiripan antar teks dokumen dengan menggunakan teknik *hashing*. Algoritma yang digunakan untuk mencari nilai *hash* dalam *winnowing* adalah *rolling hash.* Penggunaan nilai *n-gram* yang tepat pada penelitian ini diharapkan akan menghasilkan tingkat keakuratan yang lebih tinggi. Aplikasi yang dibangun menggunakan algoritma *winnowing* diharapkan menghasilkan persentase tingkat kesamaan dengan akurasi yang lebih tinggi dengan mengurangi nilai *n-gram*, sedangkan rumus *rolling hash* yang sesuai diterapkan agar mengurangi waktu proses pendeteksian sehingga akan lebih efisien..

Kata Kunci: *Plagiarisme, algoritma winnowing, rolling hash, n-gram.*

1. PENDAHULUAN

Perguruan Tinggi adalah satuan pendidikan yang menyelenggarakan pendidikan tinggi dan dapat berbentuk akademi, politeknik, sekolah tinggi, institut, atau universitas. Perguruan tinggi berkewajiban menyelenggarakan pendidikan, penelitian, dan pengabdian kepada masyarakat sebagai sebuah tri darma perguruan tinggi yang sifatnya terpadu. Perguruan tinggi dapat menyelenggarakan program akademik, profesi, dan/atau vokasi.

Perguruan tinggi (mahasiswa dan dosen) diwajibkan mampu menghasilkan karya ilmiah atau penelitian yang benar-benar hasil karyanya tanpa unsur plagiarisme. Saat ini plagiarisme dengan atau tanpa disadari kadang menjadi hal yang biasa dalam sebuah perguruan tinggi terhadap karya ilmiah atau penelitian yang dilakukan seseorang mahasiswa maupun dosen. Tindakan tersebut sering terjadi terutama pada karya ilmiah atau penelitian yang dilakukan mahasiswa untuk menyelesaikan tugas akhir, hal ini disebabkan karena terbatasnya waktu untuk menyelesaikan karya ilmiah yang dirasakan sebagai beban dan tanggungjawab seseorang sehingga terdorong untuk *copy-paste* atas karya orang lain, rendahnya minat baca, kurangnya perhatian dosen pembimbing terhadap persoalan plagiarisme serta didukung kemajuan teknologi seperti *internet* yang memungkinkan dengan mudah seseorang untuk melakukan tindak plagiarime. Persoalan ini harus menjadi perhatian serius sehingga perlu diantisipasi dan diselesaikan dengan cara mencegah dan meminimalisir tindakan *copy-paste* atau disebut juga plagiarisme.

Berdasarkan pemaparan singkat tentang beberapa hal di atas penelitian ini membuat suatu aplikasi deteksi plagiarisme untuk memproses file skala besar. Aplikasi yang diharapkan dapat menampilkan hasil dari algoritma *winnowing* dan dapat menghasilkan persentase tingkat kesamaan dengan peningkatan akurasi yang lebih tinggi dengan mengurangi nilai *n-gram* dan menggunakan rumus *rolling hash*, agar memberikan keakuratan

terhadap kemiripan dokumen dan bisa mengurangi waktu pada saat proses pendeteksian sehingga akan lebih efisien.

2. RUANG LINGKUP PENELITIAN

Permasalahan difokuskan pada:

1. Pendeteksian plagiarisme menggunakan algoritma *winnowing*
2. Metode yang digunakan adalah *n-gram* dan *rolling hash*
3. Pendeteksian plagiarisme mencocokkan dua dokumen
4. Data yang di uji bertipe text dengan ekstensi .txt.

3. BAHAN DAN METODE PENELITIAN

3.1 Plagiarisme

Plagarisme atau plagiat adalah penjiplakan atau pengambilan karangan, pendapat orang lain dan menjadikan seola-olah karangan sendiri (Sugono dkk:1997 dalam Salmuasi dan Sunyoto Andi: 2013). seperti yang dikatakan oleh Kramer Et Al (1995) dan Wray (2006) dalam Zainur (2012), menyatakan bahwa palgiarisme terjadi ketika seorang penulis mengambil karya intelektual seperti gagasan, pendapat, temuan, simpulan, data, kalimat dan kata-kata orang lain sehingga pembaca menganggap bahwa karya intelektual itu merupakan karya penulis tersebut.

Pendekatan deteksi plagiat terbagi menjadi *intrinsic* dan *external.* Pendekatan *external* terbagi lagi menjadi tiga, yaitu perbandingan teks lengkap, kesamaan kata kunci dan *fingerprinting.* Perbandingan teks lengkap diterapkan untuk membandingkan semua isi dokumen, kesamaan kata kunci bekerja dengan cara mengekstrak dan membandingkan kata kunci antardokumen, dan *fingerprinting* untuk mendeteksi kemiripan antardokumen dengan prinsip *hashing*.

Berdasarkan penelitian oleh B. Gipp dan N. Meuschke (2011) dalam Salmuasi dan Sunyoto Andi (2013) mengkatagorikan praktek plagiat berdasarkan cara yang digunakan, diantaraya:

1. *Copy & paste plagiarism,* menyalin setiap kata tanpa perubahan.
2. *Disguised plagiarism,* tergolong ke dalam praktek menutupi bagian yang disalin, teridentifikasi ke dalam empat teknik, yaitu *shake&paste, expansive plagiarism, contractive plagiarism,* dan *mosaic plagiarism.*
3. *Technical disguise,* teknik meringkas untuk menyembunyikan konten plagiat dari deteksi otomatis dengan memanfaatkan kelemahan dari metode analisis teks dasar, misal dengan mengganti huruf dengan huruf asing.
4. *Undue paraphrasing,* sengaja menuliskan ulang pemikiran asing dengan pemilihan kata dan gaya plagiator dengan menyembunyikan sumber asli
5. *Translated plagiarsm,* mengkonversi konten dari satu bahasa ke bahasa lain.
6. *Idea plagiarism,* menggunakan ide asing tanpa menyatakan sumber.

3.2 N-Gram

N-gram adalah rangkaian *token* dengan panjang *n* dalam konteks komputasi linguistik, token ini dapat berupa kata-kata, meskipun dapat berupa karakter atau himpunan bagian dari karakter (Wibowo & Hastuti: 2016). Nilai *n* hanya mengacu pada jumlah *token.* Metode *n-gram* ini digunakan untuk mengambil potongan-potongan karakter huruf sejumlah *n* dari sebuah kata yang secara kontinuitas dibaca dari teks sumber hingga akhir dari dokumen (Wibowo & Hastuti:2016). Gambar 1 adalah contoh *n-gram* dengan *n*=3.

"Membela dia"
"memebeladia"

Gambar 1. Pembentukan *n-gram*

3.3 Rolling Hash

Rolling hash merupakan teknik yang digunakan untuk mendapatkan nlai *hash* dari rangkaian *grams* yang telah

terbentuk dari metode *n-gram* yang fungsinya untuk mempercepat komputasi nilai *hash* dari rangkaian *grams* selanjutnya yang telah terbentuk (Ridho: 2013). *Rolling hash* merupakan salah satu metode *hashing* yang memberikan kemampuan untuk menghitung nilai tanpa mengulangi seluruh *string* (Wibowo & Hastuti: 2016). Berikut rumus *rolling hash.*

$$H(C_1..C_l) = C_1 . b^{(l-1)} + C_2 . b^{(l-2)} + ... + C_{(l-1)} . b + C_l \quad (1)$$

Dimana:

$H(C_1..C_l)$ = nilai *hash*

C_l = nilai ASCII karakter ke -1 pada *string*

l = panjang *string*

b = nilai basis *hash*

Maka:

$H(\text{"mem"}) = (109 \times 2^{(3-1)}) + (101 \times 2^{(2-1)}) + (109 \times 2^{(2-1)})$

$= 436 + 202 + 109$

$= 747$

Pada perhitungan nilai *hash* kedua dan seterusnya tidak perlu dilakukan perhitungan lagi dari iterasi pertama namun dengan melakukan perhitungan rumus 2.

$$H(c_2....c_{l+1}) = (H(c_1....c_l) - c_1 * b^{(l-1)}) * b + c^{(l-1)} \quad (2)$$

$H(\text{“emb”}) = (747 - 109 \times 2^{(3-1)}) \times 2 + 98 \times 2^{(1-1)}$

$= (747 - 436) \times 2 + 98$

$= 311 \times 2 + 98$

$= 720$

[747, 720, 733, 702, 717, 726, 693, 707]

Gambar 2. Nilai *hash* dari setiap *grams*

3.4 Algoritma *Winnowing*

Winnowing adalah algoritma yang digunakan untuk melakukan proses pengecekan kesamaaan kata (dokumen *fingerprinting*) untuk mendeteksi plagiarisme (Mozgovoy: 2007, dalam Ulfa, dkk: 2016). Secara teknis *winnowing* adalah ekstensi dari implementasi algoritma *rabin-karp fingerprint* dengan penambahan metode *window* (Wibowo & Hastuti: 2016). Berikut penjelasan metode dari algoritma *winnowing.*

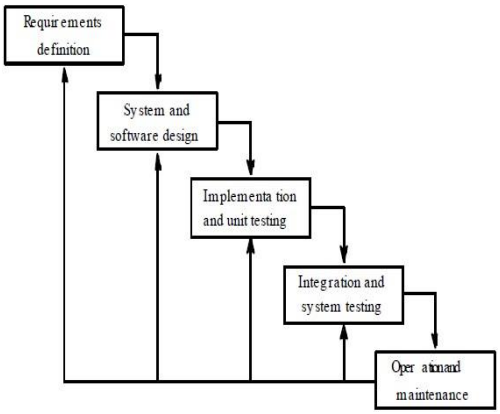
Preprocessing, proses awal dari metode dimana *input file* akan melalui berbagai macam proses manipulasi text. Metode *preprocessing* meliputi *case folding, filtering, stemming,* dan *tokenizing. Case folding* adalah proses manipulasi *case-sensitive,* semua *input* teks data akan diubah menjadi huruf kecil/*lower-case.* Sedangkan *filtering* atau sering dikenal dengan istilah *stopword removal* adalah proses penghapusan kata yang tidak relevan dalam teks. *Stemming* ialah proses pemisahan kata menjadi kata dasar. *Tokenizing* adalah proses pemisahan kata berdasarkan susunan kata (Wibowo, A.T., dkk: 2013 dalam Wibowo & Hastuti: 2016). Hasil dari proses ini akan menghasilkan dokumen teks yang relevan untuk diproses dan dicari kecocokannya. Pada *preprocess* file teks akan dibentuk menjadi rangkaian *substring* senilai *n* atau *n-gram.*

Setelah rangkaian *string* dibentuk maka akan diproses menjadi rangkaian *hash*. *Hashing* adalah proses untuk mengubah karakter *string* menjadi bilangan *integer* yang disebut nilai *hash*. Proses pengubahan menjadi nilai *hash* menggunakan fungsi *rolling hash*.

Setelah proses rangkaian *hash* terbentuk dilanjutkan dengan proses *winnowing* yaitu proses pembentukan *window* dari rangkaian *hash*. *Window* adalah proses pembentukan *substring* dari nilai *hash* sepanjang *w-gram.* Dari proses *winnowing* akan menghasilkan *fingerprint* yang nanti akan digunakan untuk pencocokan plagiasi.

3.5 Metode *Waterfall* (Air Terjun)

Sistem pendeteksian plagiat ini menggunakan prosedur pengembangan model *Waterfall*. Disebut dengan *waterfall* karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan. Sebagai contoh tahap desain harus menunggu selesainya tahap sebelumnya yaitu tahap *requirement.* Secara umum tahapan pada model *waterfall* dapat dilihat pada Gambar 3.



Gambar 3. *Waterfall*

Gambar 3 adalah tahapan dari model *waterfall* menurut Sommerville (2011:30-31) dalam Rosmiati (2015), tahapan utama dari *waterfall* model langsung mencerminkan aktifitas pengembangan dasar. Berikut adalah tahap-tahap dari *waterfall* yang dilakukan:

1. *Requirements definition.* Proses pencarian kebutuhan diintensifkan dan difokuskan pada *software.* Pada tahap ini dilakukan pengumpulan data dengan cara mencari referensi-referensi terkait yang dibutuhkan untuk penelitian. Referensi berupa buku, jurnal, tulisan penelitian dan juga artikel dari internet yang memiliki kaitan dengan algoritma *winnowing* atau yang berkaitan dengan pendeteksian plagiarisme.

2. *System and Software Design.* Proses ini digunakan untuk mengubah kebutuhan-kebutuhan di atas menjadi representasi ke dalam bentuk "*blueprint*" *software* sebelum *coding* dimulai. Pada tahapan kedua menganalisis aplikasi meliputi analisis aplikasi pendeteksi plagiat, analisis algoritma *winnowing* (*Preprocessing, n-gram*, menghitung nilai *hash*, pembentukan *window*, *Fingerprint* dan *similarity).*

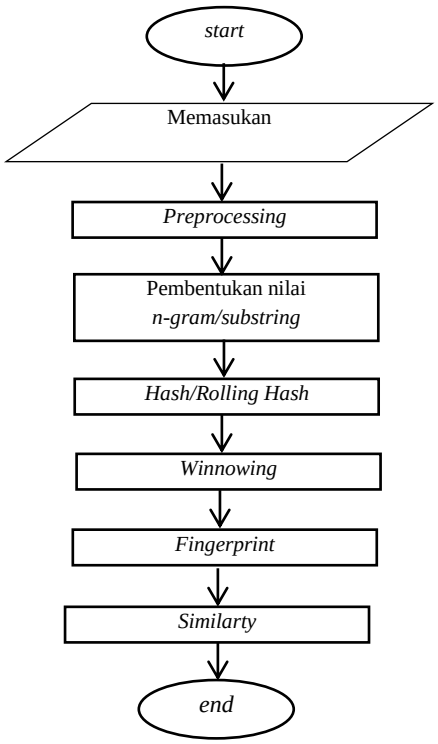
3. *Implementation and Unit Testing.* Untuk dapat dimengerti oleh mesin, maka desain tadi harus diubah bentuknya menjadi bentuk yang dapat dimengerti oleh mesin, yaitu ke dalam bahasa pemrograman melalui proses *coding.* Tahap ini merupakan implementasi dari tahap desain yang secara teknis nantinya dikerjakan oleh *programmer.* Perancangan aplikasi meliputi perancangan *database*, perancangan struktur menu, dan perancangan *interface.*

4. *Integration and Sytem Testing.* Sesuatu yang dibuat haruslah diujicobakan. Maka pada tahap ini menerapkan algoritma *winnowing* dengan mengoperasikan aplikasi yang telah dirancang.

5. *Operation and Maintenance.* Pemeliharaan suatu *software* diperlukan, termasuk di dalamnya adalah pengembangan, karena *software* yang dibuat tidak selamanya hanya seperti itu. Ketika dijalankan mungkin saja masih ada error kecil yang tidak ditemukan sebelumnya, atau ada penambahan *fitur-fitur* yang belum ada pada *software* tersebut. Pengembangan diperlukan ketika adanya perubahan dari eksternal perusahaan seperti ketika ada pergantian sistem operasi, atau perangkat lainnya.

4. RANCANGAN SISTEM/APLIKASI

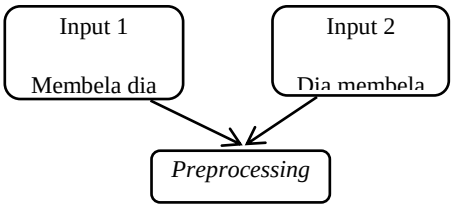
Sistem deteksi plagiat secara umum dirancang untuk dapat mendeteksi kemiripan isi pada dokumen teks, yang dimungkinkan kemiripan ini adalah hasil plagiat. *Input* sistem diperoleh dari *file*/dokumen (dalam hal ini *file* berupa *plain text*) yang di-*upload* oleh *user*. Dokumen yang di-*upload* otomatis akan tersimpan dalam *database* sistem. Gambar 4 menampilkan:



Gambar 4. *Flowchart* Alur Sistem

1. *Input* dokumen

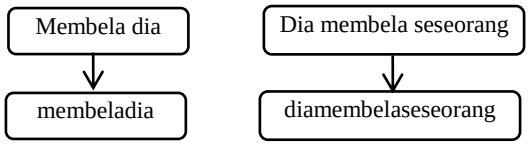
Pada awal antar muka, sistem ini akan memberikan pilihan kepada *user* untuk memasukkan teks yang akan diuji kesamaannya seperti pada Gambar 5.



Gambar 5. *Input* Dokumen

2. *Preprocessing*

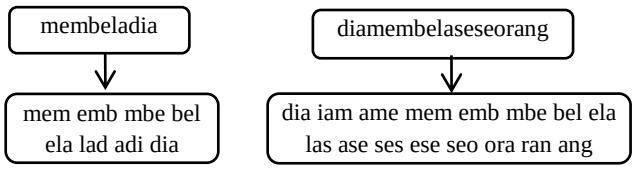
Pertama, sebelum file dokumen dicocokkan, dokumen harus melalui langkah *preprocessing* meliputi *case folding*, dan *filtering*. Pada tahap ini yaitu dengan menghilangkan karakter yang tidak relevan pada dokumen teks seperti tanda baca atau membuang semua huruf yang bukan kelompok (a-z dan 0-9), tanda spasi dan mengubah huruf besar menjadi kecil. Pada Gambar 6 merupakan hasil dari *preprocessing.*



Gambar 6. Tahapan *Preprocessing*

3. Pembentukan nilai *n-gram*

Hasil dari *preprocessing* akan meghasilkan data murni yang akan digunakan pada proses *n-gram* membentuk *substring.* Tahap ini merupakan pembentukan *n-gram* yang akan digunakan untuk menghitung *hash.* Penggunaan *n-gram* yang tepat akan menghasilkan nilai *similarity* yang tinggi. Pada Gambar 7 menggunakan *n-gram* 3 untuk menghasilkan *substring* dari rangkaian *string.*



Gambar 7. Pembentukan Nilai *n-gram*

4. Penghitungan *hash/rolling hash*

Dari rangkaian *string* yang telah terbentuk menjadi *subtring* akan diproses dengan *rolling hash* dan akan menghasilkan rangkaian nilai *hash* dari rangkaian *n-gram* yang telah terbentuk. *Substring* pertama menggunakan rumus (1) untuk menghasilkan nilai *hash*, pada perhitungan nilai *hash* kedua dan seterusnya menggunakan rumus (2) sehingga proses *winnowing* jadi lebig cepat.

$$H(C_1..C_l) = C_1 . b^{(l-1)} + C_2 . b^{(l-2)} + ... + C_{(l-1)} . b + C_l \quad (1)$$

$$H("mem") = (109x\ 2^{(3-1)}) + (101x\ 2^{(2-1)}) + (109\ x\ 2^{(2-1)})$$

$$= 436 + 202 + 109$$

$$= 747$$

$$H(c_2....c_{l+1}) = (H(c_1....c_l) - c_1 * b^{(l-1)}) * b + c^{(l-1)} \quad (2)$$

$$H(\text{“emb”}) = (747 - 109\ x\ 2^{(3-1)})\ x\ 2 + 98\ x\ 2^{(1-1)}$$

$$= (747 - 436)\ x\ 2 + 98$$

$$= 311\ x\ 2 + 98$$

$$= 720$$

[747, 720, 733, 702, 717, 726, 693, 707]

Gambar 8. Nilai *hash* dari setiap *grams*

5. Proses *winowwing/*pembentukan *window*

Nilai-nilai *hash* yang telah terbentuk, selanjutnya dibentuk dalam beberapa *window* dengan ukuran *w*. *Window* merupakan pembagian atau pengelompokan beberapa nilai *hash* dengan ukuran yang ditentukan. Pada Gambar 9 merupakan *hash* yang telah terbentuk dari *string* “membeladia” yaitu:

[747, 720, 733, 702, 717, 726, 693, 707]

ISSN 2579 7247

$$\begin{bmatrix} 747\ 720\ 733 \\ 720\ 733\ 702 \\ 733\ 702\ 717 \\ 702\ 717\ 726 \\ 717\ 726\ 693 \\ 726\ 693\ 707 \end{bmatrix}$$

Gambar 9. Pembentukan *window/w-gram*

6. *Fingerprint*

Dari *window* yang telah dibentuk dilakukan pemilihan nilai *hash* terkecil pada tiap *window* untuk dijadikan *fingerprint* tiap dokumen. Pada Gambar 10 *fingerprint* dengan menggunakan *window/w-gram* 3 dari rangkaian nilai *hash* inilah akan dipilih nilai *hash* terkecil, jika terdapat 2 atau lebih nilai yang sama, nilai *hash* terkecil yang paling kanan yang akan di pilih.

[720 702 693]

Gambar 10. *Fingerprint*

7. *similarity*

Similarity adalah persentase tingkat kemiripan antar dokumen. Pada tahap ini proses pencocokkan dari dua dokumen yang telah melalui proses *winnowing* dan telah dipilih *fingerprint* terkecil dari dua dokumen teks.

Dokumen teks 1: [720 702 693]

Dokumen teks 2: [707 720 702 717 719 735 740 711]

$$(1, 2) = \frac{1 \cap 2}{1 \cup 2} \text{ X } 100\% \qquad (3)$$

$$= \frac{[720\ 702]}{[707\ 720\ 702\ 717\ 719\ 735\ 740\ 711]} \text{ X } 100$$

= 22,22%

5. IMPLEMENTASI

Dari percobaan yang dilakukan diperoleh data pengujian. Data yang digunakan adalah kalimat asli dan kalimat uji. Hasil dari percobaan ditampilkan pada Tabel 1 yang menampilkan hasil uji kalimat dengan nilai *n-gram* yang berbeda.

Tabel 1. Uji Kalimat dengan *n-gram* dan *w-gram* yang Berbeda

No	Dokumen	*n-gram*	*w-gram*	*Similarity* (%)	Waktu (detik)
1	Membela dia Dia membela seseorang	3	3	22,22	0,007
2	Membela dia Dia membela seseorang	5	5	00,00	0,006
3	Membela dia Dia membela seseorang	7	3	00,00	0,005

Bagian pertama menggunakan *n-gram* 3 dan *w-gram* 3 menghasilkan nilai *similarty* 22,2% dengan waktu 0,007 detik. Pada bagian kedua dengan *n-gram* 5 dan *w-gram* 5 menghasilkan nilai *similarty* yaitu 00,0% dengan waktu 0,006 detik. Pada bagian terakhir menggunakan *n-gram* 7 dan *w-gram* 3 menghasilkan nilai *similarty* yang sama dengan bagian kedua yaitu 00,0% dan waktu yang dihasilkan dalam proses yaitu 0,005 detik.

Semakin rendah nilai *n-gram* maka semakin tinggi nilai *similarity*-nya dan semakin tinggi nilai *n-gram*, maka semakin kecil *similarity*-nya atau semakin kecil tingkat keakuratannya. Semakin kecil nilai *n-gram* maka semakin kecil karakter yang akan dicocokan dan semakin sering karakter tersebut akan ditemukan dalam teks

Proses waktu yang dihasilkan dari nilai *n-gram* yang rendah dan tinggi akan menghasilkan proses yang berbeda, semakin rendah nilai *n-gram* maka waktu yang diperlukan dalam memproses lebih rendah dibandingkan menggunakan *n-gram* yang lebih tinggi. Hal ini dikarenakan proses pemilihan *substring* dan perhitungan *hash* lebih sedikit, tiap-tiap *substring* memuat konten karakter lebih banyak sehingga jumlah dari banyaknya *substring* akan berkurang.

6. KESIMPULAN

Pengujian awal yang telah dilakukan didapatkan semakin rendah nilai *n-gram* dan *w-gram* maka akan semakin tinggi nilai *similarity*-nya dan proses yang diperlukan semakin lambat. Semakin besar niliai *n-gram* dan *w-gram* maka semakin rendah nilai akurasinya atau *similarity*-nya dan proses yang dihasilkan lebih cepat.

Algoritma *winnowing* baik digunakan untuk mendeteksi karya ilmiah dengan mengurangi nilai *n-gram* dan jauh lebih cepat prosesnya karena *winnowing* menggunakan rumus *rolling hash* sehingga tidak perlu dilakukan perhitungan lagi dari iterasi pertama.

7. SARAN

Ada beberapa hal yang perlu diperhatikan dalam melakukan penelitian lebih lanjut terkait sistem rekomendasi ini, yaitu:

a. Algoritma *winnowing* kedepannya bisa mencocokkan banyak dokumen.

b. Penelitian lanjutan diharapkan dapat menggunakan beberapa metode terhadap algoritma *winnowing* agar tingkat akurasi lebih baik lagi.

c. Perlu ujicoba menggunakan *n-gram* dan *w-gram* yang tepat agar dapat meningkatkan akurasi dari deteksi plagiarisme.

d. Perlu evaluasi dan pembanding dengan metode yang lain.

8. DAFTAR PUSTAKA

Ridho, Muhammad. 2013. Rancang Bangun Aplikasi Pendeteksi Penjiplakan Dokumen Menggunakan Algoritma *Biword Winnowing*. *Jurusan Teknik Informatika Universitas Islam Negeri SLTAN Syarif Kasim Pekanbaru Riau.*

Rosmiati, Mia. 2015. Analisis Dan Perancangan *E-Service* untuk Pelanggan pada Jaya Besama Konveksi. *Indonesian Journal on Software Engineering* (*IJSE), Volume. 1, No. 1*.

Salmuasih & Sunyoto, Andi. 2013. Implementasi Algoritma *Rabin Karp* untuk Pendeteksian Plagiat Dokumen Teks Menggunakan Konsep *Similarity*. *Seminar Nasional Aplikasi Teknologi Informasi (SNATI),* Hal 1907 - 5022.

Ulfa, N.F., dkk. 2016. Pendeteksian Tingkat Similaritas Dokumen Berbasis *Web* Menggunakan Algoritma *Winnowing*. *Konferensi Nasional Teknologi Informasi dan Komunikasi (KNASTIK),* hal 2338-7718.

Wibowo, R.K., & Khafiizah. 2016, Penerapan Algoritma *Winnowing* untuk Mendeteksi Kemiripan Teks pada Tugas Akhir Mahasiswa. *Jurnal Techno.COM, Vol. 15, No. 4,* Hal 301-311.

Zainur, Muhammad. 2012. Plagiarisme diKalangan Mahasiswa dalam Membuat Tugas-tugas Perkuliahan pada Fakultas Tarbiyah IAIN Imam Bonjol Padang. *Jurnal Al-Ta'lim, Jilid 1, Nomor 1,* Hal 55-65.