TESIS

ANALISIS PENERAPAN ALGORITMA STEMMING NAZIEF & ADRIANI PADA PERBANDINGAN ALGORITMA WINNOWING DAN ALGORITMA RATCLIFF/OBERSHELP PADA PENDETEKSI KESAMAAN PRODUK DI MARKETPLACE

Disusun oleh:

Nama : **Wahyu Hidayat**
NIM : **19.51.1181**
Konsentrasi : **Business Intelligence**

PROGRAM STUDI S2 TEKNIK INFORMATIKA
PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA
YOGYAKARTA
2021

TESIS

ANALISIS PENERAPAN ALGORITMA STEMMING NAZIEF & ADRIANI PADA PERBANDINGAN ALGORITMA WINNOWING DAN ALGORITMA RATCLIFF/OBERSHELP PADA PENDETEKSI KESAMAAN PRODUK DI MARKETPLACE

ANALYSIS OF NAZIEF & ADRIANI STEMMING ALGORITHM ON COMPARISON OF WINNOWING ALGORITHM AND RATCLIFF/OBERSHELP ALGORITHM ON PRODUCT SIMILARITY DETECTION IN MARKETPLACE

Diajukan untuk memenuhi salah satu syarat memperoleh derajat Magister

Disusun oleh:

Nama : **Wahyu Hidayat**
NIM : **19.51.1181**
Konsentrasi : **Business Intelligence**

PROGRAM STUDI S2 TEKNIK INFORMATIKA
PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA
YOGYAKARTA
2021

HALAMAN PENGESAHAN

ANALISIS PENERAPAN ALGORITMA STEMMING NAZIEF & ADRIANI PADA PERBANDINGAN ALGORITMA WINNOWING DAN ALGORITMA RATCLIFF/OBERSHELP PADA PENDETEKSI KESAMAAN PRODUK DI MARKETPLACE

ANALYSIS OF NAZIEF & ADRIANI STEMMING ALGORITHM ON COMPARISON OF WINNOWING ALGORITHM AND RATCLIFF/OBERSHELP ALGORITHM ON PRODUCT SIMILARITY DETECTION IN MARKETPLACE

Dipersiapkan dan Disusun oleh

Wahyu Hidayat

19.51.1181

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada hari **Rabu, 6 Januari 2021**

Tesis ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Magister Komputer

Yogyakarta, 6 Januari 2021
Rektor

Prof. Dr. M. Suyanto, M.M.
NIK. 190302001

HALAMAN PERSETUJUAN

ANALISIS PENERAPAN ALGORITMA STEMMING NAZIEF & ADRIANI PADA PERBANDINGAN ALGORITMA WINNOWING DAN ALGORITMA RATCLIFF/OBERSHELP PADA PENDETEKSI KESAMAAN PRODUK DI MARKETPLACE

ANALYSIS OF NAZIEF & ADRIANI STEMMING ALGORITHM ON COMPARISON OF WINNOWING ALGORITHM AND RATCLIFF/OBERSHELP ALGORITHM ON PRODUCT SIMILARITY DETECTION IN MARKETPLACE

Dipersiapkan dan Disusun oleh

Wahyu Hidayat

19.51.1181

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada hari **Rabu, 6 Januari 2021**

Pembimbing Utama	**Anggota Tim Penguji**
Prof. Dr. Ema Utami, S.Si., M.Kom.	**Dr. Arief Setyanto, S.Si., M.T.**
NIK. 190302037	**NIK. 190302036**
	Dr. Andi Sunyoto, M.Kom.
Pembimbing Pendamping	**NIK. 190302052**
Anggit Dwi Hartanto, M.Kom.	**Prof. Dr. Ema Utami, S.Si., M.Kom.**
NIK. 190302163	**NIK. 190302037**

Tesis ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Magister Komputer

Yogyakarta, 6 Januari 2021
Direktur Program Pascasarjana

Dr. Kusrini, M.Kom.
NIK. 190302106

HALAMAN PERNYATAAN KEASLIAN TESIS

Yang bertandatangan di bawah ini,

Nama mahasiswa : Wahyu Hidayat
NIM : 19.51.1181
Konsentrasi : Business Intelligence

Menyatakan bahwa Tesis dengan judul berikut:
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace

Dosen Pembimbing Utama : Prof. Dr. Ema Utami, S.Si., M.Kom.
Dosen Pembimbing Pendamping : Anggit Dwi Hartanto, M.Kom.

1. Karya tulis ini adalah benar-benar ASLI dan BELUM PERNAH diajukan untuk mendapatkan gelar akademik, baik di Universitas AMIKOM Yogyakarta maupun di Perguruan Tinggi lainnya
2. Karya tulis ini merupakan gagasan, rumusan dan penelitian SAYA sendiri, tanpa bantuan pihak lain kecuali arahan dari Tim Dosen Pembimbing
3. Dalam karya tulis ini tidak terdapat karya atau pendapat orang lain, kecuali secara tertulis dengan jelas dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan disebutkan dalam Daftar Pustaka pada karya tulis ini
4. Perangkat lunak yang digunakan dalam penelitian ini sepenuhnya menjadi tanggung jawab SAYA, bukan tanggung jawab Universitas AMIKOM Yogyakarta
5. Pernyataan ini SAYA buat dengan sesungguhnya, apabila di kemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka SAYA bersedia menerima SANKSI AKADEMIK dengan pencabutan gelar yang sudah diperoleh, serta sanksi lainnya sesuai dengan norma yang berlaku di Perguruan Tinggi

Yogyakarta, 6 Januari 2021
Yang Menyatakan,

Wahyu Hidayat

HALAMAN PERSEMBAHAN

Alhamdulillah dengan rasa syukur yang mendalam dengan telah diselesaikannya tesis ini berkat segala nikmat dan kasih sayang yang telah diberikan oleh Allah Subhanahu Wa Ta'ala, maka penulis mempersembahkan tesis ini kepada semua pihak yang terlibat secara langsung ataupun tidak langsung dalam proses pembuatan tesis.

1. Orang tua dan saudara kakak-kakak, yang selalu mendoakan, memberikan semangat untuk menjalani perkuliahan.
2. Prof. Dr. M. Suyanto, M.M selaku Rekor Universitas AMIKOM Yogyakarta yang telah memberikan kesempatan kepada penulis untuk melanjutkan Studi jenjang Strata 2 Program Studi Magister Teknik Informatika di Universitas Amikom Yogyakarta.
3. Prof. Dr. Ema Utami, S.Si., M.Kom dan Bapak Anggit Dwi Hartanto, M.Kom yang telah membimbing penulis dari awal sampai akhir proses pembuatan tesis.
4. Agustin Dwi Cahyani yang selalu mengingatkan dan menyemangati untuk mengerjakan tesis .
5. Semua pihak yang tidak bisa disebutkan satu persatu yang sudah memberi semua ilmu pengetahuan, informasi dan segalanya sehingga penulis bisa menyelesaikan tesis ini.

HALAMAN MOTTO

"The only source of knowledge is experience." Albert Einstein

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Allah Subhanahu Wa Ta'ala atas nikmat dan karunia-Nya sehingga penulis dapat menyelesaikan tesis yang berjudul "Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace". Penulis menyadari tidak akan dapat menyelesaikan tesis ini dengan baik tanpa bimbingan, saran dan motivasi dari berbagai pihak. Peneliti mengucapkan terimakasih yang sebesar-besarnya kepada :

1. Prof. Dr. M. Suyanto, MM. selaku Rektor Universitas AMIKOM Yogyakarta.
2. Prof. Dr. Ema Utami, S.Si., M.Kom. dan Bapak Anggit Dwi Hartanto, M.Kom. selaku Dosen Pembimbing.
3. Bapak Dr. Arief Setyanto, S.Si., M.T., Bapak Dr. Andi Sunyoto, M.Kom. dan Prof. Dr. Ema Utami, S.Si., M.Kom. selaku Dosen Penguji.
4. Orang tua serta saudara selaku wali yang telah memberikan dukungan dan motivasi.

Semoga Allah Subhanahu Wa Ta'ala memberikan balasan yang lebih kepada pihak yang telah membantu penulis menyelesaikan tesis ini. Semoga tesis ini dapat bermanfaat bagi masyarakat.

Yogyakarta, 6 Januari 2021

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

INTISARI

Persaingan dalam penjualan produk sejenis menjadi meningkat karena semakin banyak pemasaran produk yang sama di marketplace. P-store.net merupakan marketplace yang menjadi platform jual beli produk fisik dan digital, produk yang dijual mendominasi produk karya sendiri. Identifikasi kesamaan produk diperlukan untuk mengetahui tingkat duplikasi produk pada persaingan produk sejenis. Perlu pengujian algoritma *similarity* dan penerapan pengaruh algoritma *stemming* pada pendeteksi kesamaan produk untuk mengetahui tingkat akurasi, efisiensi serta performa *stemming*.

Penentuan jumlah dataset menurut rumus Wibisono, dataset yang digunakan sebanyak 100 produk selanjutnya dibagai menjadi 50% untuk data dengan indikasi duplikat dan 50% untuk data dengan indikasi tidak duplikat. Algoritma Winnowing dan algoritma Ratcliff/Obershelp dibandingkan untuk mengetahui tingkat akurasi dan efisiensi dalam mendeteksi kesamaan antar produk. Algoritma stemming Nazief & Adriani diterapkan untuk tahap *stemming* pada *text preprocessing* sehingga dapat diketahui hasil performa dalam perbandingan algoritma *similarity*.

Algoritma *stemming* Nazief & Adriani menambah waktu pemrosesan dari *text preprocessing* hingga hasil nilai *similarity* ditemukan dengan total rata-rata selisih penambahan waktu algoritma Winnowing 7,45 detik dan algoritma Ratcliff/Obershelp 7,44 detik dan menyebabkan hasil penurunan nilai rata-rata *similarity* secara garis besar terhadap dua algoritma tersebut. Algoritma Ratcliff/Obershelp lebih efisien daripada algoritma Winnowing pada 4 pengujian dan 2 pengujian lainnya seimbang. Hasil akurasi terbaik indikasi produk tidak duplikat didapatkan algoritma Winnowing dengan *stemming* Nazief & Adriani pada nilai k = 7 dan tipe data deskripsi dengan nilai *similarity* 8,12% (plagiarisme ringan). Hasil akurasi terbaik indikasi produk duplikat diperoleh algoritma Ratcliff/Obershelp tanpa *stemming* pada tipe data judul dengan nilai *similarity* 86,91% (plagiarisme berat atau total).

Kata kunci: Algoritma *stemming* Nazief & Adriani, Algoritma Winnowing, Algoritma Ratcliff/Obershelp, Jaccard Similarity, *Similarity*

ABSTRACT

Competition in selling similar products is increasing due to more marketing of same product in marketplace. P-store.net is marketplace for buying and selling physical/digital products. Identification product similarities need to determine level product duplication in competition for similar products. It's necessary to test similarity algorithm and apply effect stemming algorithm detection product similarities to determine accuracy, efficiency, and performance of stemming.

Determining number of datasets according to Wibisono's formula, used dataset with 100 products, then divided into 50% for data with duplicate indications and 50% data with non-duplicate indications. Winnowing algorithm and Ratcliff/Obershelp algorithm were compared to determine level of accuracy and efficiency in detecting similarities. Nazief & Adriani stemming algorithm is applied for stemming stage so performance results in comparison of similarity algorithms can be seen.

Nazief & Adriani stemming algorithm increases processing time from preprocessing text until similarity value is found with total average time difference of Winnowing algorithm's addition time of 7.45 seconds and Ratcliff/Obershelp algorithm of 7.44 seconds and causes result of decrease in average value similarity by line big against these two algorithms. Ratcliff/Obershelp algorithm is more efficient than Winnowing algorithm at 4 tests and 2 tests are balanced. Best accuracy results indicated that product wasn't duplicate, obtained by Winnowing algorithm with Nazief & Adriani at k = 7 and type description with similarity value of 8.12% (low plagiarism). Best accuracy of duplicate product indications was obtained by Ratcliff/Obershelp algorithm without stemming on type title with similarity value of 86.91% (heavy or total plagiarism).

Keyword: Nazief & Adriani Stemming Algorithm, Winnowing Algorithm, Ratcliff/Obershelp Algorithm, Jaccard Similarity, Similarity

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Masyarakat indonesia saat ini lebih menyukai bertransaksi atau belanja melalui internet, salah satu faktor yang menyebabkan ini adalah pertumbuhan *e-commerce* dan *marketplace* yang semakin pesat. Tingginya minat masyarakat untuk berbelanja melalui internet serta kemudahan yang di tawarkan oleh *e-commerce* ataupun *marketplace* tidak lepas dari perkembangan tekonologi yang cukup pesat (Simamora dan AK, 2019). Namun tak hanya itu yakni transaksi belanja *online* semakin disukai karena lebih menguntungkan pelakunya dari segi finansial. Salah satunya persaingan di *marketplace* yaitu persaingan dalam penjualan produk digital. Peminat produk digital pada kebutuhan masyarakat tentang teknologi contohnya pada bidang *software* aplikasi ataupun *website*, *e-course*, dan desain grafis semakin banyak dibutuhkan karena untuk memperluas bisnis sehingga dapat dimanfaatkan untuk mendapatkan penghasilan dari *online marketing*.

Persamaan produk yang ada di setiap toko di *marketplace* menyebabkan tidak meratanya pembelian produk sejenis, karena calon pembeli cenderung melakukan penyaringan produk tertentu sebelum mencari produk yang akan dicari, seperti penyaringan jumlah penjualan untuk mengetahui bahwa produk disalah satu toko mempunyai pelanggan yang banyak, dan penyaringan jumlah ulasan untuk mengetahui bahwa produk yang sudah dibeli oleh para pelanggan mempunyai reputasi yang baik dan ditandakan dengan satuan bintang antara satu sampai lima.

Persaingan pada produk yang sejenis harus diperhatikan oleh pemilik toko, persaingan ini dapat terjadi akibat perkembangan pada sektor inovasi teknologi, ekonomi (Gunawan, 2019). Sebelum menjual atau mempublikasi produk ke *marketplace* sebaiknya mengetahui terlebih dahulu berapa toko pesaing pada produk yang sejenis sehingga apabila sudah diketahui maka calon penjual akan mengetahui jumlah persaingan untuk memasarkan produk yang sama.

P-store.net merupakan *marketplace* yang menjadi platform jual beli dalam mengembangkan produk berupa produk fisik dan digital serta jasa virtual yang inovatif dan kompetitif. Produk yang dijual pada P-store.net mempunyai karakteristik yaitu produk yang dijual merupakan produk penunjang digital *marketing* seperti jasa penulisan artikel, pembuatan *software*, pembelian *tools*, *e-book*, desain grafis dan jasa periklanan. Berdasarkan produk yang dijual pada P-store.net diketahui bahwa produk yang dijual mendominasi dengan kriteria produk yang memiliki detail yang berbeda seperti dalam pembuatan logo walaupun ada beberapa produk dengan tipe yang memiliki kesamaan namun memiliki deskripsi dan ketentuan yang berbeda karena produk yang dijual tidak terkait dengan merek barang tertentu jadi deskripsi antara penjual satu dengan yang lainnya memungkinkan memiliki deskripsi yang berbeda dan memiliki hasil digital yang berbeda. Hal lainnya seperti pada pembuatan *software*, walaupun jenis *software* yang di jual belikan mempunyai tipe yang sama namun deskripsi lengkap software tersebut berkemungkinan memiliki deskripsi yang berbeda karena pembutan *software* dilakukan oleh berbeda penjual.

Penelitian tentang persaingan produk pada *e-commerce* telah banyak diteliti sebelumnya, seperti melakukan pengujian untuk mengetahui pengaruh *e-commerce* pada keunggulan kompetitif (Syahria, dkk, 2020). Sebagian besar dari data responden menggunakan data *e-commerce* terhadap produk kuliner yang dibuatnya dari daerah kota Makassar, Pare-pare, Bulukumba dan Bantaeng yang mengakibatkan meningkatkatnya omset dan daya saing pada produk sejenis, dengan menggunakan program SPSS mendapatkan nilai t dan variabel X1(*Online/ e-commerce*) lebih besar dari nilai t Tabel (2,00>1,66), serta hasil tingkat signifikansi 0,04, sehingga disimpulkan bahwa variabel X1 (*Online / e-commerce*) secara parsial berpengaruh terhadap daya saing kuliner tradisional di Sulawesi Selatan.

Pemasaran produk usaha mikro melalui *e-commerce* lokal dan nasional dengan melakukan analisis SWOT untuk menentukan urutan strategi yang paling prioritas dalam pengembangan usaha mikro di Kota Samarinda (Putranto, dkk, 2019). Terdapat beberapa ancaman salah satunya yaitu persaingan produk sejenis, dari analisis SWOT yang dilakukan yerdapat beberapa kesimpulan tentang strategi prioritas, diantaranya prioritas pertama dalam pengembangan usaha mikro di Kota Samarinda yaitu melakukan promosi produk, prioritas kedua yaitu melakukan pemasaran produk melalui *e-commerce*, prioritas ketiga mendirikan inkubator bisnis teknologi, prioritas keempat mendorong pihak swasta untuk berperan aktif, prioritas kelima yaitu melakukan penguatan regulasi daerah.

E-commerce menciptakan persaingan usaha yang kompetitif dimana terdapat banyak penjual, banyak pembeli, dan produk yang dijual sama, namun

pada kasus entry barrier berbeda karena keadaan tersebut tidak ada pesaing lain yang dapat menembus pasar pada produk yang sejenis, maka diperlukan kualitas baik dan harga yang kompetitif sehingga akan tercipta pasar kondusif (Hotana, 2018). *E-commerce* dapat menciptakan pasar kompetitif dan kondusif, memberikan peluang kepada pelaku usaha dengan mudah di internet, perlu adanya pengaturan Undang-Undang Nomor 5 Tahun 1999 mengenai peran Komisi Pengawas Persaingan Usaha dalam industri *e-commerce* untuk menghentikan monopoli dan persaingan usaha tidak sehat.

Pendeteksian kemiripan teks telah dilakukan sebelumnya untuk mengetahui tingkat persentase *similarity* terhadap teks yang dibandingkan, sehingga banyak penelitian yang telah dilakukan untuk mendeteksi kesamaan teks tersebut, penelitian tersebut meliputi dari pengambilan teks, pengolahan teks dan implementasi beberapa algoritma untuk membandingkan kemiripan teks atau dokumen. Penelitian tersebut antara lain :

Text mining merupakan penggalian informasi pada koleksi teks berskala besar serta mengidentifikasi pola dan hubungan yang menarik dalam data tekstual pada sumber data dari kumpulan dokumen yang tidak terstruktur daripada terstruktur (Feldman dan Sanger, 2007). Buku tersebut sebagai pengantar untuk penambangan teks, yang mencakup arsitektur umum sistem penambangan teks, dengan teknik utama yang digunakan oleh sistem tersebut dan menunjukkan dengan bukti dari contoh dunia nyata pada sistem FACT yang telah mereka kembangkan pada akhir 1990-an dengan pengintegrasian secara efektif ke dalam sistem penambangan teks. *Text preprocessing* adalah tahapan dari *text mining*, tahapan ini

bertujuan mengolah teks menjadi kata dasar untuk diproses ke tahap selanjutnya. Langkah dari *text preprocessing* antara lain *case folding* sebagai konversi teks ke dalam huruf kecil dan menghilangkan karakter selain huruf, *tokenizing* sebagai pemisah kata berdasarkan karakter spasi atau tabulasi, *filtering* tahap memilih kata-kata penting dan menghilangkan kata hubung, dan *stemming* tahap mengurai kata menjadi bentuk kata dasarnya (Muhammad, dkk, 2019).

Stemming merupakan tahapan dari *text preprocessing* yaitu pada tahapan proses untuk menemukan kata dasar pada sebuah kata. Terdapat beberapa algoritma *stemming* pada teks berbahasa Indonesia dan telah di lakukan penelitian sebelumnya tentang performa perbandingan dari beberapa algoritma tersebut. Menentukan tingkat ketepatan kata dasar (Novitasari, 2017) membandingkan dua *stemmers* Indonesia, yaitu Porter dan Arifin Setiono dengan 40 sampel dokumen dengan Bahasa Indonesia yang berisikan kata kerja, kata sifat, kata benda, dan kata keterangan yang diambil dari Kamus Besar Bahasa Indonesia (KBBI), menghasilkan *Exact Match* yaitu *stemmer* Porter sebesar 90,00%, *stemmer* Arifin Setiono95,00%. *Unchange* yaitu *stemmer* Porter 2,50%, *stemmer* Arifin Setiono 2,50%. *Spelling Exception stemmer* Porter 7,50%, *stemmer* Arifin Setiono 0%. *Overstemming stemmer* Porter 0%, *stemmer* Arifin Setiono 2,50%. Disimpulkan bahwa algoritma Arifin & Setiono memiliki presisi lebih tinggi dibandingkan dengan *stemming* Porter.

Penelitian lain yang membahas tentang pengujian algoritma *stemming* diteiliti oleh (Agusta, 2009) melakukan pencarian informasi pada dokumen teks dengan proses *Information Retrieval* (IR) atau pemisahan dokumen yang dianggap

relevan dari sekumpulan dokumen lainnya yang tersedia. *Stemming* berperan untuk meningkatkan performa IR dengan mentransformasi kata dalam sebuah dokumen teks menjadi kata dasarnya, pada penelitian tersebut melakukan pengujian dengan membandingkan efektivitas dan efisiensi dari dua *stemmer* Indonesia, yaitu Porter dan Nazief & Adriani dan menggunakan dataset yang dilakukan pada 30 dokumen teks Bahasa Indonesia dengan pengujian menggunakan ukuran dokumen yang bervariasi. Hasilnya dengan menguji pada 30 dokumen, algoritma Porter membutuhkan waktu yang lebih singkat yaitu dengan rata-rata waktu 0,37 detik dibandingkan dengan *stemming* menggunakan algoritma Nazief & Adriani yaitu dengan rata-rata waktu 18,72 detik. Namun algoritma Nazief & Adriani memiliki keakuratan lebih besar yaitu dengan presisi 84,27% dibandingkan dengan *stemming* algoritma Porter 70,56%.

Penelitian selanjutnya juga telah membahas tentang membandingkan performa kecepatan, akurasi pada algoritma Nazief & Adriani dengan algoritma Porter pada proses *stemming* pada teks ber-Bahasa Indonesia (Wahyudi, dkk, 2017). Penelitian tersebut dilakukan menggunakan sebaran data sebanyak 26 dokumen uji yang diperoleh dari situs berita (detik.com) pada tanggal 27 September 2015 sedangkan untuk kata yang disimpan pada penelitian ini terdapat 8.168 kata yang akan diujikan, 1.312 data kata *term*asuk kata yang muncul lebih dari satu kali, sedangkan 4.724 data kata *term*asuk dalam stopword dan 2.132 kata diujikan pada kedua *stemmer* yang diterapkan pada penelitian ini. Hasil dari penelitian ini dari sebanyak 2.132 kata yang diuji menyatakan bahwa algoritma Nazief & Adriani menghasilkan kebenaran kata dasar sebanyak 2.031 kata, sedangkan untuk

algoritma Porter 1.687 kata. Tingkat kesalahan algoritma *stemmer* Nazief & Adriani 5,00%, sedangkan algoritma Porter 21,00%. Efisiensi waktu proses, algoritma Porter lebih baik yaitu memiliki 12,38 detik sedangkan algoritma Nazief & Adriani memiliki 22,166 detik namun algoritma Nazief & Adriani memberikan hasil yang lebih baik untuk *stemming* pada dokumen berbahasa Indonesia dengan akurasi 95,26% sedangkan algoritma Porter 79,13%.

Pengujian algoritma *stemming* lainnya (Simarangkir, 2017) melakukan pengujian terhadap algoritma Nazief & Adriani, Arifin & Setiono, Vega, dan Tala. Proses *stemming* yaitu menghilangkan semua imbuhan (*affixes*) yang terdiri dari awalan (*prefixes*), sisipan (*infixes*), akhiran (*suffixes*) dan *confixes*. *Stemming* pada bahasa Indonesia, terdapat dua jenis metode *stemming* yaitu algoritma *stemming* yang berbasis kamus (*dictionary based*) dan algoritma *stemming* yang berbasis *non*-kamus (*purely rule based*). *Stemming* digunakan sebagai penguraian dari suatu kata menjadi bentuk kata dasarnya untuk meningkatkan kualitas informasi yang didapatkan. Dataset yang digunakan yaitu menggunakan dokumen sebanyak 100 dokumen yang sudah dibuat dengan format bertipe .txt dan memiliki total kata sebanyak 25.937 kata. Hasilnya pada rata-rata setiap pengujian yaitu algoritma Nazief & Adriani memiliki waktu proses 5,15 detik, akurasi 97,93%, *overstemming* 1,40% dan *understemming* 0,40%. Algoritma Arifin & Setiono memiliki waktu proses 15,20 detik, akurasi 92,09%, *overstemming* 6,10% dan *understemming* 1,56%. Algoritma Vega memiliki waktu proses 0,08 detik, akurasi 63,49%, *overstemming* 30,33% dan *understemming* 6,06%. Algoritma Tala memiliki waktu proses 0,22 detik, akurasi 78,27%, *overstemming* 19,09% dan *understemming*

1,70%, dari hasil pengujian yang didapatkan disimpulkan bahwa untuk algoritma yang memakai aturan imbuhan ditemukan algoritma terbaik yaitu algoritma Tala sedangkan untuk algoritma Nazief & Adriani merupakan algoritma terbaik.

Stemming merupakan proses mereduksi kata yang berimbuhan menjadi kata dasar, maka kata akan menjadi bentuk umum, sehingga akan menghasilkan dokumen yang memiliki relevan yang lebih baik (Rezalina, 2016). Penelitian tersebut bertujuan untuk mengetahui pengujian performansi yang berlandaskan kecepatan dan ketepatan dari masing-masing algoritma dari algoritma *stemming* Nazief & Adriani, Porter dan Arifin Setiono yang digunakan sebagai pencarian kata dasar pada abstrak sebuah jurnal dalam dokumen teks berbahasa Indonesia dengan dataset menggunakan 10 dokumen teks bahasa Indonesia dengan masukan dokumen yang bervariasi. Hasil dari penelitian tersebut menyatakan algoritma Nazief & Adriani memiliki hasil yang lebih unggul dalam hal kecepatan dan akurasi dibandingkan dengan dua algoritma lainnya pada pengujian 10 dokumen teks dengan perbandingan rata-rata akurasi antara algoritma Nazief & Adriani, algoritma Porter, algoritma Arifin & Setiono yaitu 87,00%:85,80%:86,80% sedangkan untuk kecepatan perbandingan rata-ratanya yaitu 16,75 detik:40,39 detik: 53,22 detik.

Terjadinya penjiplakan yang semakin marak terjadi karena seringnya kebiasaan peneliti yang malas dan ingin cepat dalam menyelesaikan tugas sehingga kemungkinan mencari referensi instan yang terdapat pada internet maupun perpustakaan. Perlu *software* untuk menyelesaikan persoalan tersebut menggunakan algoritma pencocokan *string* (Purba dan Situmorang, 2017). Karena

ada beberapa algoritma pada pencocokan *string* maka penelitian tersebut membandingkan algoritma Rabin-Karp terhadap algoritma Levenshtein Distance. Pengujian sistem yang dilakukan dengan 2 *Input*an kalimat untuk dibandingkan. Hasilnya algoritma Rabin-Karp pencocokkan multiple pattern lebih bagus jika dibandingkan dengan single pattern, hasil pengujian algoritma Rabin-Karp yaitu 79,85% sedangkan algoritma Levenshtein Distance memiliki tingkat plagiasi 66,67%.

Plagiat merupakan sebuah tindak pidana karena merupakan kegiatan mencuri hak cipta orang lain, pelaku plagiat sering disebut sebagai plagiator. Plagiarism merupakan hal yang hampir tidak dapat dihindari oleh karena itu, tindakan plagiarisme secara perlahan harus ditindaklanjuti (Wicaksono, 2012). Penelitian tersebut melakukan pendeteksian plagiat terhadap dokumen teks berbahasa Indonesia dengan menggunakan algoritma Rabin-Karp dengan proses menggunakan *hashing* yang dapat digunakan untuk menemukan sebuah *substring* dalam sebuah teks serta menggunakan algoritma Nazief & Adriani pada tahap *preprocessing* dengan data yang diuji bertipe .txt dengan tidak memperhatikan kesalahaan ejaan dan sinonim. Hasil dari penelitian tersebut yaitu besarnya akurasi dari sistem tergantung pada pemilihan nilai k-grams dan pemakaian *preprocessing* maka hasil nilai akurasi algoritma Rabin-Karp mencapai 90,00% untuk tipe plagiat aktif-pasif, *carbon copy*, ubah struktur kata dan tambah kata sedangkan karena sistem yang dibuat tidak menggunakan *synonym recognition* maka untuk tipe sinonim memiliki akurasi 72,76%.

Kemampuan mencari *similarity* dapat diterapkan pada pendeteksi plagiarisme dalam jurnal ataupun sebuah karya ilmiah (Sukmana, dkk, 2018). Penelitian tersebut bertujuan untuk membandingkan antara algoritma Rabin karp murni terhadap rabin karp dengan menggunakan algoritma *stemming* Nazief & Adriani pada tahap *text preprocessing*. Metode yang diterapkan adalah mnetode *fingerprinting* dengan algoritma Rabin Karp dan Jaccard *Similarity* serta pemakaian data yang diuji menggunakan 4 dokumen teks yang dimodifikasi. Hasil penelitian tersebut yaitu algoritma Rabin Karp yang menggunakan algoritma *Stemming* Nazief & Adriani dapat mempercepat waktu eksekusi dengan memiliki hasil *similarity* yang hampir sama yaitu persentase perbandingan antara algoritma Rabin Karp murni dan algoritma Rabin Karp dengan algoritma *Stemming* Nazief & Adriani pada percobaan dokumen 1 100%:100%, pada percobaan dokumen 2 74,87%:72,09%, pada percobaan dokumen 3 48,39%:45,69%, pada percobaan dokumen 4 23,60%:20,59%.

Google merupakan mesin pencari dan pengguna dapat melakukan pencarian dokumen untuk sumber informasi atau referensi makalah ilmiah. Namun semakin banyak kemudahan tentu ada masalah yang mengganggu yaitu salah satunya adalah plagiarisme. Penerapan algoritma untuk mendeteksi kemiripan antar teks (Leonardo dan Hansun, 2017) Pada dokumen terdapat *string* yang dirangkai menjadi satu kata atau kalimat, dengan menggunakan teknik *string* matching dapat dibandingkan antar dokumen jika ada indikasi plagiarisme atau tidak dengan algoritma *similarity* yang digunakan untuk pencocokan *string*. Membandingkan efektifitas dari algoritma Rabin-Karp dan Jaro-Winkler pada pendeteksi plagiarisme pada

dokumen teks degnan dataset yang digunakan ada beberapa variasi yaitu 30 dokumen .txt dengan ukuran <1000 KB, 30 dokumen .doc dan .docx dengan ukuran <1000 KB, 30 dokumen .pdf dengan ukuran <1000 KB, dan 30 dokumen lain dengan ukuran >1000 KB. Algoritma Rabin Karp lebih efektif daripada algoritma Jaro-Winkler Distance, pada beberapa percobaan algoritma Rabin Karp mendapatkan persentase kemiripan yang lebih tinggi daripada Jaro-Winkler Distance yaitu 51,00%:35,00%. Dalam waktu pemrosesan, algoritma Rabin-Karp memiliki hasil rata-rata 0,59 menit sedangkan algoritma Jaro-Winkler Distance memiliki hasil rata-rata 0,99 menit.

Metode *fingerprinting* merupakan metode yang digunakan untuk menelusuri karakter satu persatu pada deret karakter dan proses dari metode dokumen *fingerprinting* dengan menggunakan teknik *hashing* (Hermawan, 2015). Analisis performansi dilakukan untuk mendapatkan informasi dari algoritma yaitu bertujuan untuk mendapatkan informasi ketepatan, kecepatan, dan jumlah langkah. Penelitian tersebut membandingkan performa dari algoritma untuk mendeteksi kemiripan antar teks dengan menggunakan algoritma Winnowing dan algoritma Manber dengan dataset yang digunakan menggunakan dokumen Google.txt yang memiliki 5.633 karakter dibandingkan dengan dokumen Google 2.txt yang memiliki 4.862 karakter, dokumen Sejarah Google.txt yang memiliki 3.654 karakter dibandingkan dengan dokumen Google 2.txt yang memiliki 3.148 karakter. Hasil dari penelitian tersebut dalam hal kecepatan Manber lebih baik dengan nilai proses 0,10 detik dari Winnowing 0,33 detik namun dalam hal ketepatan Winnowing lebih baik dari Manber karena dari perhitungan manual

menghasilkan nilai kemiripan 86,15% pada pengujian algoritma winnowing mendapatkan persentase nilai pengujiannya 86,12% sedangkan untuk algoritma manber memiliki nilai pengujiannya 85,71%.

Saat ini segala jenis informasi dapat ditemukan dengan cara yang mudah melalui internet, namun informasi ini sering disalahgunakan oleh berbagai pihak yang mengarah pada plagiarisme. Plagiarisme dalam penulisan ilmiah dapat menurunkan kredibilitas publik terhadap kualitas penelitian di tempat institusi pada pelaku plagiarisme (Wibowo, dkk, 2013). Penelitian tersebut melakukan implementasi dalam mendeteksi kesamaan dokumen pada tugas akhir fakultas informatika berjumlah 112 judul dengan memparafrasekan 32 judul dokumen lain yang selanjutnya di proses menggunakan metode *fingerprint* metode n-gram dibandingkan dengan algoritma Winnowing yang memakai Dice Coefficient untuk menghitung koefisien kemiripannya. Hasil penelitian tersebut yaitu nilai akurasi algoritma *Fingerprint* lebih tinggi 92,80% dibandingkan dengan algoritma Winnowing 91,80%, namun algoritma Winnowing memiliki hasil kinerja yang lebih baik dan memiliki hasil yang lebih stabil pada spesifik kata kunci dengan menghasilkan nilai spesifik pada Distinct *Term* dan *Term* Frequency yaitu 34,70% dan 37,10% sedangkan metode *fingerprint* menghasilkan nilai spesifik pada Distinct *Term* dan *Term* Frequency yaitu 31,70% dan 33,60%.

Pendeteksi plagiarisme telah dikembangkan pada kasus tertentu sepertin untuk data teks berupa essay, artikel, jurnal, dan penelitian. Terdapat metode *fingerprinting* untuk mendeteksi kemiripan dokumen, pada metode tersebut terdapat beberapa algoritma diantaranya algoritma Rabin Karp, algoritma Manber

dan algoitma Winnowing. Pendekatan algoritma untuk mendeteksi plagiarisme atau kemiripan teks dokumen (Alamsyah, 2017) Membandingkan metode *fingerprinting* yang terdapat di algoritma Rabin Karp dengan algoritma Winnowing dalam pendekatan pendeteksi plagiarisme dengan pembandingan data yang ada pada database judul-judul skripsi mahasiswa yang telah lulus dibandingkan dengan data pengajuan judul skripsi mahasiswa. Hasilnya pada ujicoba ke 8 algoritma Winnowing dengan nilai n-gram = 9 dan *window* = 3 , menghasilkan proses waktu 0,03 detik dengan tingkat kemiripan terkecil yaitu 32,60% dibandingkan dengan algoritma Rabin-Karp pada ujicoba yang sama yaitu memerlukan waktu 0,03 detik dengan tingkat kemiripan 54,78% dengan hasil tersebut disimpulkan tingkat presentase algoritma Winnowing lebih baik.

Penelitian pendeteksi kesamaan teks untuk memberikan solusi yang dapat mengatasi masalah dengan sistem pendeteksian yang bertujuan untuk mendetksi kemiripan judul atau pun teks skripsi dengan menggunakan algoritma Winnowing dan algoritma Rabin- Karp (Sibarani, dkk, 2019). Algoritma winnowing sebagai mendeteksi adanya keberadaan kesamaan kata pada dua judul yang dibandingkan, sedangkan algoritma Rabin-Karp untuk pencarian *string* yang berjumlah banyak dengan menggunakan data *Input*an judul skripsi satu dibandingkan dengan *Input*an judul skripsi lainnya. Hasil dari penelitian tersebut yaitu merancang aplikasi yang dapat membandingkan judul skripsi dengan menggunakan algoritma Winnowing dan algoritma Rabin Karp serta menghasilkan tingkat akurasi kemiripan judul mendekati 75,00%.

Algoritma pencocokan *string* didasarkan pada skor kesamaan antara dua *string* dari masukan *string* dan di cocokkan pada *string* yang telah ada pada database (Ilyankou, 2014). Pemeriksaan ejaan merupakan fitur penting dari perangkat lunak seperti penggunaannya pada pengolah teks, *email*, kamus dan mesin pencari. Namun, pemeriksa ejaan membuat pengguna lebih cepat dan lebih percaya diri saat menulis, dengan pemeriksaan ejaan pengguna menggunakan kata-kata yang tidak dikenal secara lebih intensif, tanpa takut melakukan kesalahan. Peneltiian tersebut membandingkan dua algoritma pencocokan *string* pada pemeriksaan ejaan yaitu membandingkan antara algoritma Jaro-Winkler dan algoritma Ratcliff/Obershelp dalam aplikasi spell check dengan menggunakan dataset dari daftar kata kamus yang diambil dari FreeBSD sebanyak 236.000 kata dan kamus Mieliestronk sebanyak 58.000 kata. Hasil dari penelitian tersebut yaitu menghasilkan kesimpulan bahwa algoritma Ratcliff/Obershelp 4% lebih efisien daripada algoritma jarak Jaro-Winkler. Algoritma Ratcliff/Obershelp bekerja sangat baik ketika ada salah ketik pada karakter pertama atau terakhir dalam *string*. Algoritma Ratcliff/Obershelp menunjukkan hasil 4,00% -18,60% lebih baik saat memproses daftar 53 kata yang salah eja.

Kegiatan plagiat terus terjadi dan mengakibatkan kurangnya seseorang untuk dapat menghargai karya orang lain maka perlu dilakukan tahapan pencegahan terhadap tindakan plagiat salah satunya dengan *software* antiplagiarisme kesamaan teks (Yusuf, dkk, 2019). Penelitian tersebut menggunakan algoritma untuk mendeteksi plagiarisme pada portal berita *online* dengan membandingkan antara algoritma Rabin-Karp dan algoritma Ratcliff/Obershelp sebagai menghitung

persentase nilai *similarity* dokumen teks pada portal berita. Penelitian ini menggunakan data crawling sebanyak 50 halaman berita dari portal berita *online* dan setelah melakukan tahap cleaning diperoleh data bersih sebanyak 6.090 kata dengan 46.233 huruf yang selanjutnya dibagi kedalam 10 data uji. Hasilnya algoritma Ratcliff/Obershelp mendekati persentase data masukan dibandingkan algoritma Rabin-Karp, selisih nilai *similarity* rata-rata dua algoritma tersebut 3,00%. Tata letak kalimat mempengaruhi nilai *similarity* algoritma Rabin-Karp, sedangkan algoritma Ratcliff/Obershelp sama sekali tidak dipengaruhi. Algoritma Rabin-Karp memiliki tingkat konsistensi yang rendah dibandingkan algoritma Ratcliff/Obershelp. Pada pengujian 10 data algoritma Ratcliff/Obershelp memiliki nilai rata-rata 1,33 *milisecond* dengan 2 nilai kemiripan (94 dan 95) sedangkan algoritma Rabin-Karp memiliki nilai rata-rata 4,2 *milisecond* dan menghasilkan 5 nilai kemiripan yang berbeda (93, 94, 95, 96 dan 97) dengan k-gram 2, 4, 6, 8 dan 10 pada pengujian yang sama.

Sehingga berdasarkan studi literatur yang telah dilakukan terhadap penelitian sebelumnya maka penulis memakai p-store.net sebagai studi kasus pada penelitian ini karena pada *marketplace* tersebut mempunyai karakteristik produk yang akan dijadikan sebagai dataset memungkinkan memiliki deskripsi yang berbeda pada setiap produk yang ditawarkan walaupun pada jenis produk yang sama sehingga nantinya dapat di teliti lebih lanjut tentang presentase tingkat kemiripan pada produk sejenis untuk mengetahui dalam persaingan produk tersebut terdapat duplikasi produk atau produk yang sejenis namun tidak duplikasi. Serta peneliti menggunakan algoritma *stemming* Nazief & Adriani karena pada beberapa

penelitian yang telah dilakukan dihasilkan bahwa algoritma *stemming* tersebut unggul daripada algoritma *stemming* lainnya pada teks berbahasa Indonesia, sedangkan pada algoritma pencocokan *string* atau pendeteksi *similarity* peneliti menggunakan algoritma Winnowing dengan Jaccard *Similarity* dan menggunakan algoritma Ratcliff/Obershelp karena pada beberapa pengujian dalam penelitian sebelumnya menghasilkan bahwa kedua algoritma tersebut unggul di masing-masing pengujian dan belum pernah dibandingkan sebelumnya sehingga nantinya dapat diketahui perbandingan dari kedua algoritma *similarity* tersebut. Hasil akhir dari penelitian adalah melihat apakah ada pengaruh performa dari penerapan algoritma *stemming* Nazief & Adriani terhadap algoritma Winnowing dan algoritma Ratcliff/Obershelp dalam mendeteksi kesamaan produk, serta melakukan perbandingan hasil akurasi dari kedua algoritma tersebut dalam mendeteksi kesamaan produk.

1.2. Rumusan Masalah

Berdasarkan latar belakang masalah yang telah dijelaskan di atas maka rumusan masalah dalam penelitian ini sebagai berikut :

a. Apakah penerapan algoritma *stemming* Nazief & Adriani mempengaruhi performa dalam penerapannya pada masing-masing algoritma yaitu algoritma Winnowing dan algoritma Ratcliff/Obershelp dan dalam perbandingan pada algoritma Winnowing dan algoritma Ratcliff/Obershelp dalam mendeteksi tingkat persentase kesamaan produk ?
b. Berapa tingkat akurasi dan efisiensi dalam perbandingan algoritma Winnowing dan algoritma Ratcliff/Obershelp dalam mendeteksi tingkat persentase

kesamaan produk tanpa menggunakan algoritma *stemming* dan menggunakan algoritma *stemming* Nazief & Adriani?

1.3. Batasan Masalah

Pembatasan masalah dalam penelitian digunakan untuk menghindari adanya penyimpangan atau pelebaran pokok masalah agar nantinya penelitian lebih terarah sehingga tujuan penelitian akan tercapai. Beberapa batasan masalah dalam penelitian ini sebagai berikut:

a. Data yang dipakai pada penelitian ini hanya berupa teks.
b. Batasan *text mining* dalam penelitian ini hanya berfokus pada teks berbahasa Indonesia.
c. Penelitian ini tidak memperhatikan kesalahaan ejaan dan sinonim.
d. Penelitian ini tidak memperhatikan penanganan negasi, sarkasme, hiperbola, dan pendeteksian emosi.
e. Pengambilan *text mining* hanya pada judul dan deskripsi pada produk dari *Marketplace* PT. Trijaya Digital Grup (TRIDI) yaitu di p-store.net.
f. Pengambilan dan pengolahan data serta pemrosesan algoritma menggunakan PHP dan MySQL.
g. Pada tahap *stemming* pada *preprocessing* menggunakan algoritma *stemming* Nazief & Adriani.
h. Pendeteksian tingkat persentase kesamaan produk menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp.
i. Penentuan jumlah dataset yaitu dengan pengambilan data sampel dengan jumlah total populasi tidak diketahui menurut rumus Wibisono.

j. Data set yang digunakan adalah 100 produk pada jenis produk digital yang dikumpulkan dari *text mining* pada p-store.net.

k. Pendeteksi persentase kesamaan produk dilakukan pada pembandingan kesamaan dari judul produk terhadap judul produk yang lain serta deskripsi produk terhadap deskripsi produk yang lain serta membandingkan judul+deskripsi produk.

l. Mengetahui tingkat performa pada penerapan algoritma *stemming* Nazief & Adriani dilakukan berdasarkan hasil persentase masing-masing algoritma yaitu pada algoritma Winnowing dan algoritma Ratcliff/Obershelp dan pada perbandingan algoritma Winnowing dan algoritma Ratcliff/Obershelp tanpa menggunakan algoritma *stemming* dan hasil persentase perbandingan algoritma Winnowing dan algoritma Ratcliff/Obershelp menggunakan algoritma *stemming* Nazief & Adriani.

1.4. Tujuan Penelitian

Adapun tujuan penulis dalam melakukan penelitian ini adalah sebagai berikut :

a. Mengetahui performa penerapan algoritma *stemming* Nazief & Adriani pada tahap *stemming* pada proses *text preprocessing* yang di implementasikan pada masing-masing algoritma yaitu pada algoritma Winnowing dan algoritma Ratcliff/Obershelp serta dalam perbandingan perbandingan algoritma Winnowing dan algoritma Ratcliff/Obershelp.

b. Membandingkan akurasi dan efisiensi dari algoritma Winnowing dan algoritma Ratcliff/Obershelp pada pendeteksi kesamaan produk di *marketplace*.

1.5. Manfaat Penelitian

Adapun manfaat penelitian adalah sebagai berikut :

a. Bagi penjual (*seller*) di p-store dapat menginformasikan persentase persaingan produk yang sama yang akan dijual pada *marketplace* p-store.
b. Bagi pembeli di p-store dapat menginformasikan daftar duplikasi produk terhadap toko lainnya.
c. Bagi peneliti dapat mengetahui akurasi dan efisiensi dari perbandingan algoritma Winnowing dan algoritma Ratcliff/Obershelp serta mengetahui performa algoritma *stemming* Nazief & Adriani dalam penerapan pada masing-masing algoritma *similarity* dan pada perbandingan kedua algoritma tersebut.

BAB II

TINJAUAN PUSTAKA

2.1. Tinjauan Pustaka

Terdapat penelitian tentang pembahasan persaingan produk pada pemasaran salah satunya yaitu tentang ekstraksi data dengan web crawler untuk menyajikan segmentasi aktivitas peletakan produk di online shop berdasarkan jenis produk yang di pasarkan pada markeplace sehingga diketahui strategi untuk memulai bisnis dan pelaku bisnis untuk mengetahui segmentasi pasar (Surahman, dkk, 2020). Kelebihan dari penelitian ini terdapat detail segmentasi seperti segmentasi yang diinginkan sesuai dengan kategori *behavioral*, demografi, *geographics* dan *psychographics*, namun kelemahan pada penelitian ini yaitu tidak dijelaskan secara spesifik dalam melakukan pengujian ekstraksi pada penentuan jenis produk yang akan diteliti. Hasil pengujian prototype pada ekstraksi data produk marketplace dari pengolahan 250 responden yaitu pada kriteria kualitas informasi mendapatkan skor aktual 80,00% untuk kriteria kualitas sistem mendapatkan skor aktual 79,00% untuk kualitas layanan mendapatkan skor aktual 77,00% untuk kriteria penggunaan mendapatkan skor aktual 79,00% untuk kriteria kepuasan pengguna mendapatkan skor aktual 78,00%. Maka dari keseluruhan untuk kesuksesan informasi segmentasi pasar yaitu sebesar 79,00%.

Penelitian lain terkait dengan kajian daya saing OTT *e-commerce* untuk memetakan kondisi dan kemampuan *e-commerce* Indonesia dalam menghadapi tantangan dan persaingan Global. Asosiasi *e-commerce* Indonesia menghimpun

anggota berbagai sektor bisnis *online* yaitu: 7 Bank, 18 *Classified Ads*, 6 *Daily Deals*, 1 direktori, 29 infrastruktur, 20 logistik, 54 *marketplace*, 104 *online* retail, 21 *payment gateway* dan 18 *travel*. Dalam *e-commerce*, terdapat produk digital dan produk berwujud, pada ekonomi digital semua data serta transaksi yang ada termasuk dalam bentuk informasi digital yang akan dapat diekstraksi. Persaingan yang ada pada *e-commerce* tidak hanya persaingan harga namun keberagaman dan kualitas produk menjadi daya tarik tersendiri bagi konsumen (Abdullah, 2018). Kelebihan pada penelitian ini menggunakan dasar hukum yang lengkap serta pembahasan data secara global dan besar dengan sumber terkait. Saran pada penelitian ini untuk lebih spesifik dalam menunjukkan data misalnya pada *e-commerce* dengan produk tertentu sehingga akan diketahui persaingan yang tepat. Hasil kaijan tersebut tentang perlunya penerapan dari rekomendasi laporan BEPS dengan memperbaiki peraturan pajak, dan belajar dari pengalaman dari Negara lain tentang perusahaan *e-commerce* multinasional dan penyedia layanan OTT.

Melakukan implementasi dalam proses bisnis transaksi *online* menggunakan Opencart pada Pastbrik Malang menggunakan dataset observasi, wawancara dan dokumentasi menghasilkan perubahan penjualan konvensional/offline karena kurang efektif dan efisien dengan membuat *e-commerce* pada Pastbrik Malang untuk mendapatkan hasil yang maksimal serta dapat saling bersaing dengan toko sejenis (Maulana, dkk, 2015). Mengoptimasi *e-commerce* produk UMKM hingga pasar global dengan teknik Search Engine Optimisazion / SEO (Diartono, dkk, 2015), penelitian tersebut berkaitan dengan persaingan produk sejenis dapat terjadi antara produk lokal dan luar maka peneliti

melakukan implementasi pada pembuatan *e-commerce* untuk produk umkm seperti batik, kerajinan tangan, dan pariwisata pada kota kendal. Kelebihan penelitian tersebut yaitu peneliti melakukan penerapan beberapa teknik SEO seperti penerapan meta *description*, penempatan tag *title* dan memberikan *backlink*. Kelemahan penelitian tersebut tidak dijelaskan dengan detail tentang data yang diolah pada penelitian, hasil dari penelitian tersebut terkait dengan hasil statistik pada Alexarank menunjukkan *rank* website yang diuji memiliki *rank* 40,867 di Indonesia.

Menghadapi persaingan produk impor dengan mengetahui strategi inovasi dan imitasi pada penjualan produk sejenis (Hannisa, dkk. 2016). Penelitian tersebut menentukan inovasi pada perubahan strategi pemasaran produk. Hasil perhitungan regresi linear berganda terhadap 80 data kuisoner dari wawancara mendalam terhadap beberapa pemasaran. Kelebihan penelitian tersebut menggunakan data kekuatan bersaing dengan membandingkan beberapa produk yang sejenis, namun kekurangan penelitian tersebut tidak dijelaskan secara detail tentang poin-poin data hasil wawancara dan kuisoner yang diproses pada penelitian. Hasil pada penelitian tersebut memperoleh beberapa hasil koefisien yaitu pada variabel intercept 7,35 variabel imitasi 0,02 variabel inovasi 0,42 menunjukkan bahwa terdapat pengaruh strategi imitasi terhadap daya saing, serta inovasi terhadap daya saing.

Berdasarkan penelitian sebelumnya terkait dengan analisis tentang persaingan produk, sehingga penulis tertarik untuk melakukan penelitian terkait dengan ide tentang menganalisa persaingan produk pada *marketplace*, selanjutnya untuk proses dalam pengolahan data dan analisa data, penulis menggunakan metode

yang berbeda namun juga berdasarkan kajian dari peneliti sebelumnya tentang penggunaan metode tersebut. Berikut beberapa kajian tentang penggunaan metode tersebut mencakup dengan tahap *text preprocessing*, algoritma Winnowing, Jaccard *Similarity* dan algoritma Ratcliff/Obershelp.

Penelitian yang akan dilakukan oleh penulis dalam tahap proses *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani maka dari itu penulis meninjau hasil peneliti sebelumnya yang menggunakan algoritma steming tersebut. Tujuan dari tahap *text preprocessing* yaitu untuk mentransformasikan *Input* data menjadi lebih mudah untuk memproses pada langkah selanjutnya. Algoritma Nazief & Adriani digunakan pada tahap *stemming* proses *text preprocessing*. Algoritma ini menerapkan aturan dasar morfologi Indonesia, memeriksa pengikatan yang diizinkan dan pengikatan tidak diizinkan, dan menggunakan kamus kata dasar Bahasa Indonesia untuk membandingkan kata dasar (Amalia, dkk, 2019). Penelitian yang dilakukan oleh Amalia untuk membuat konsep evaluasi esai otomatis dengan perbandingan kesamaan jawaban mahasiswa dengan kunci jawaban dari dosen. Tahap pengujiannya menggunakan *Input*an jawaban mahasiswa dengan *Input*an jawaban dosen selanjutnya di proses pada tahap *preprocessing* dan dihitung tingkat kemiripannya menggunakan Cosine *Similarity*. Kelebihan pada penelitian ini menggunakan deteksi sinonim berdasarkan kamus. Sedangkan kekurangan pada penelitian ini tidak dijelaskan jumlah detail pengujian setiap jawaban, dalam tahap penjelasan hanya menghitung percontohan 1 dokumen yang diteliti, namun tidak disajikan hasil detail dari perhitungan semua dokumen dan hanya menampilkan hasil rata-rata akurasi.

Hasilnya dari 60 jawaban yang diuji dengan skenario yang ditentukan, menghasilkan akurasi sistem sebesar 83,30%.

Pendeteksian kemiripan teks dengan jumlah besar dan banyak menjadi sulit dilakukan oleh manusia (Imbar, dkk, 2014), pendeteksian *similarity* teks dapat dilakukan untuk mencegah plagiarisme. Algoritma Winnowing digunakan untuk mendeteksi kemiripan antar dokumen (Fauzi dan Wibawa, 2018), algoritma Winnowing menggunakan metode *fingerprinting* yaitu melakukan teknik *hash* yang terbentuk dari perhitungan ASCII setiap karakter selanjutnya melakukan pembentukan *window* dan pemilihan *fingerprint hash* terkecil dari tiap *window*. Langkah algoritma Winnowing yaitu : membuang karakter tidak relevan, membentuk n-gram, menghitung Rolling Hash, pembentukan *window*, pemilihan *fingerprint hash* terkecil dari tiap *window*, hitung persamaan dengan Jaccard Coeficient. Kelebihan dalam penelitian ini yaitu menggunakan variasi n-gram pada pemilihan panjang nilai *n*. Kelemahan penelitian ini tidak menggunakan dataset yang bervariasi dalam jumlah karakter dan jumlah katanya, karena pada penelitian tersebut menggunakan 4 dokumen yang hampir sama dalam jumlah karakter dan jumlah katanya. Hasil penelitian yang dilakukan oleh Fauzi dengan menguji 6 pembanding dalam deteksi kesamaan memperoleh hasil 100%, 7,95%, 7,48%, 5,06%, 4,31%, dan 4,08%.

Penerapan algoritma Winnowing dan Jaccard's Coefficient juga telah di lakukan oleh peneliti sebelumnya, seperti penelitian tentang pendeteksi *similarity* lintas bahasa dari bahasa Indonesia terhadap bahasa Inggris (Putri Ratna, dkk, 2019) menggunakan algoritma Winnowing untuk perhitungan *fingerprints*. Pengujian

penelitian yang dilakukan oleh Putri Ratna menggunakan perbandingan 2 *core* processor, 4 *core* processor, dan 8 *core* processor dengan masing-masing diberikan *Input*an 10, 100, dan 1.000 dokumen referensi. Kelebihan penelitian ini membandingkan performa menggunakan beberapa macam processor untuk mengetahui efisiensi serta melakukan perbandingan kemiripan teks antar bahasa yang berbeda. Kelemahan dari penelitian ini tidak menggunakan algoritma *stemming* untuk proses merubah kata dasar. Hasilnya jumlah *core processor* yang lebih tinggi dapat memproses jumlah data yang lebih besar dengan nilai kecepatan yang lebih baik, di dapatkan nilai kecepatan 3,52 detik pada pengujian paper ke 3 dengan 22 paragraf dan 1.000 referensi dokumen pada pengujian dengan *processor* 8 *core*, sehingga tingkat efisiensi untuk varietas tersebut 50,91%.

Plagiarisme merupakan tindakan mencuri karena mengambil ide, gagasan, dan pikiran orang lain tanpa izin, atau tanpa menyantumkan pemilik aslinya, dan bahkan mengakui karya tersebut miliknya (Sunardi, dkk, 2018). Penelitian tersebut melakukan penerapan algoritma untuk mendeteksi plagiarisme yaitu dengan menggunakan algoritma Winnowing sebagai pendeteksian kemiripan dokumen yang akan diidentifikasi apakah dokumen tersebut adanya suatu tindak plagiariasme dan Jaccard *Similarity* sebagai pembanding antar dua dokumen dengan menghitung kemiripan dari dokumen yang dibandingkan. Penelitian tersebut menggunakan data *Input*an kalimat satu dan kalimat dua sebagai pembanding, pada beberapa percobaan yaitu melakukan pembeda pada *Input*an nilai n-gram dan w-gram. Semakin kecil nilai n-gram dan w-gram maka potongan kata yang ada akan sering ditemukan ketika dicocokkan, dan sebaliknya jika nilainya terlalu besar akan

semakin jarang data itu di temukan. Kelebihan pada penelitian ini membandingkan nilai pada *Input*an n-gram, w-gram dengan nilai bilangan prima. Kelemahan penelitian yaitu peneliti menggunakan contoh pengujian data sebanyak satu kali, tidak dijelaskan dataset yang digunakan darimana saja serta pada tahap *text preprocessing* tidak ada tahap *stemming* untuk mengubah *string* menjadi kata dasar. Hasilnya dengan melakuan pengujian dengan pembeda nilai n-gram dan w-gram diperoleh hasil nilai dengan n-gram 3 dan nilai dengan w-gram 3 mempunyai hasil nilai *similarity* 35,71%, nilai dengan n-gram 5 dan nilai dengan w-gram 3 mempunyai hasil nilai *similarity* 41,67%, nilai dengan n-gram 4 dan nilai dengan w-gram 5 mempunyai hasil nilai *similarity* 45,45%, nilai dengan n-gram 4 dan nilai dengan w-gram 4 mempunyai hasil nilai *similarity* 41,67%, nilai dengan n-gram 6 dan nilai dengan w-gram 6 mempunyai hasil nilai *similarity* 18,18%.

Algoritma Ratcliff/Obershelp juga dapat memutuskan seberapa mirip dua pola satu dimensi dengan mencari nilai yang nantinya di gunakan sebagai faktor persentase (Jacob, dkk, 2019). Konsep pencocokan dari algoritma ini, dengan menemukan *substring* terpanjang yang memiliki kesamaan berdasarkan *string* satu dan *string* dua yang di sebut sebagai anchor, kemudian bagian yang tersisa dari *string* sebelah kiri dan kanan dari anchor harus diperiksa sebagai *string*-*string* yang baru, proses tersebut akan di ulangi sampai seluruh karakter dari *string* satu dan *string* dua di analisa. Pengujian penelitian yang dilakukan oleh Jacob menggunakan tugas dari mahasiswa yang berjumlah 5 dokumen dan di uji dengan artikel pada wikipedia dengan banyak karakter pada dokumen 1 sebanyak 871, dokumen 2 sebanyak 2.405, dokumen 3 sebanyak 4.325, dokumen 4 sebanyak 7.633, dokumen

5 sebanyak 7.633. Kelebihan pada penelitian ini yaitu membandingkan beberapa *Input*an dokumen dengan jumlah kata yang berbeda sehingga dapat mengetahui performa penggunaan algoritma pada teks sekala kecil ataupun besar. Saran untuk menambahkan algoritma *stemming* dan menguji algoritma sejenis untuk mengetahui hasil perbandingan akurasi. Setelah dilakukan pengujian terhadap 5 dokumen dengan artikel pada wikipedia dihasilkan terhadap dokumen 1 dengan prediksi nilai 0% dan memperoleh nilai 14,00%, sedangkan untuk dokumen 2 dengan prediksi nilai 25,00% dan memperoleh nilai pengujian 35,00%, sedangkan untuk dokumen 3 dengan prediksi nilai 50,00% dan memperoleh nilai pengujian 67,00%, sedangkan untuk dokumen 4 dengan prediksi nilai pengujian 75,00% dan memperoleh nilai pengujian 72,00%, dan untuk dokumen 5 dengan prediksi nilai 100% dan memperoleh nilai pengujian 54,00%.

Plagiat dapat dianggap tindakan pidana karena merupakan tindakan mencuri hak cipta milik orang lain dan pelakunya sering disebut sebagai plagiator. Karena adanya alat bantu komputer sehingga memberikan kemudahan kepada mahasiswa untuk menjiplak. Plagiat sangat berdampak buruk bagi mahasiswa, mahasiswa menjadi malas dalam mengerjakan tugas kuliah dan hanya berharap kepada orang lain sehingga dapat mengurangi kreatifitas sehingga dosen juga direpotkan dalam menganalisis tugas yang dikumpulkan mahasiswa dengan membandingkan dokumen secara manual. Cara tersebut kurang efektif dan efisien mengingat jumlah mahasiswa yang tidak sedikit sehingga memerlukan waktu lama. Salah satu algoritma untuk mencari kesamaan dari *string* yang dibandingkan yaitu menggunakan algoritma Ratcliff/Obershelp pada pencarian kesamaan berdasarkan

string dari dokumen yang berbeda dan juga dalam penilaian tingkat kesamaan dokumen teks (Joane, dkk, 2017). Data uji yang digunakan pada penelitian tersebut menggunakan 6 dokumen tugas mahasiswa yang dibandingkan dengan 11 dokumen, pada tahap pengujian dilakukan uji pada dokumen 1 terhadap 11 dokumen. Kelebihan pada penelitian ini yaitu melakukan pengujian dengan merotasi kalimat pada dokumen yang di uji sehingga diketahui perbedaan hasil pada perbandingan dokumen yang sama namun di ubah urutan kalimatnya. Kelemahan pada penelitian ini pada tahap *stemming* tidak menggunakan algoritma *stemming* untuk menambah performa pencarian kata dasar. Hasilnya banyaknya karakter didalam dokumen mempengaruhi waktu eksekusi dari proses pengidentifikasian kesamaan dokumen, dan posisi kata ataupun kalimat yang sama akan dapat mempengaruhi nilai *similarity* dari pendeteksian, pada prediksi dan hasil nilai perbandingan kesamaan dokumen 1 terhadap 11 dokumen memiliki rata-rata perbandingan antara prediksi dan nilai yaitu 50,00%:54,90%.

2.2. Keaslian Penelitian

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
1	Kemudahan Aplikasi dan Keragaman Produk dalam Membentuk Keputusan Pembelian Generasi Milenial Berbelanja secara *Online*	Awy Apriani Ningrum Simamora, Marlya Fatira AK, Jurnal Maneksi, 2019	Meneliti mengenai dampak yang dapat diberikan berdasarkan ketersediaan dan kemudahan pada penggunaan aplikasi dan keragaman pada produk yang dijual pada market *online* yang dihadapkan pada keputusan pembelian generasi milenial pada berbelanja *online*.	Karakteristik mayoritas konsumen kegiatan proses melakukan belanja *online* merupakan jenis kelamin wanita, yang memiliki rentang usia 20 tahun. Keragaman pada produk *online* tersebut berpengaruh positif terhadap keputusan pada pembelian oleh konsumen dalam berbelanja *online*, hal tersebut ditampilkan oleh hasil koefisien regresi dengan nilai 0,253 dan 0,417, dan nilai signifikansi uji f 0,000, nilai koefisien determinasi R2 = 0,475.	Karena pemakaian data hanya pada lingkup usia tertentu membuat hasil dari analisis yang dilakukan tidak memiliki banyak varian yang didapatkan, dan karena data tersebut didapatkan menggunakan kuisioner dengan jumlah yang terbatas tidak bisa mewakili pengguna aplikasi belanja *online* secara keseluruhan.	Penulis menggunakan algoritma yang berbeda dan tujuan penelitian tidak sama namun penulis mengambil intisari dari ilmu persaingan produk yang ada pada belanja *online*.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Pencliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
2	Pengaruh *Electronic Word Of Mouth* Terhadap *Purchase Intention* pada Produk Aloevera Nature Republic di Surabaya Melalui Mediasi *Brand Image* dan *Brand Trust*	Leo Hadi Gunawan, Universitas Katolik Widya Mandala Surabaya, 2019	Penelitian tersebut melakukan pengujian serta menganalisis mengenai pengaruh electronic word of mouth pada sebuah brand image, pengaruh electronic word of mouth pada brand trust, serta pengaruh brand image terhadap purchase intention pada produk Aloevera Nature Republic di Surabaya	Electronic word of mouth berpengaruh pada brand image terhadap produk Aloevera Nature Republic, Electronic word of mouth berpengaruh pada brand trust, Brand image berpengaruh pada purchase intention terhadap produk Aloevera Nature Republic di Surabaya. Hal tersebut didasari pada kesan baik tentang kesan baik, kesan baik , konsumen memiliki rencana untuk membeli produk, serta kepuasan konsumen terhadap produk Aloevera Nature Republic.	Koresponden yang digunakan kurang lengkap hal itu membuat ambigu penelitian terutama pada jangkauan seluruh masyarakat surabaya karena koresponden yang didapat kurang banyak dan tidak bisa mewakili jangkauan luas.	Penulis menggunakan algoritma yang berbeda dan tujuan penelitian tidak sama namun penulis mengambil intisari dari ilmu persaingan produk yang ada pada belanja *online*.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Pencliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
3	*Effect of Online Marketing (E-commerce) Activities to the Advantages of Competing Traditional Culinary Products in South Sulawesi*	Syahrial, Lily Dianafitry Hasan, Muhammad Musawantoro, Faisal Akbar Zainal, Journal of Tourism, Hospitality, Travel and Busines Event, 2020	Melakukan pengujian untuk mengetahui pengaruh *e-commerce* pada keunggulan kompetitif.	Menggunakan program SPSS mendapatkan nilai *t* serta var X1(*Online/Ecommerce*) yang berkedapatan lebih besar dari nilai t Tabel (2,00>1,66), dengan tingkat nilai yang signifikansinya sebesar 0,04, sehingga dapat menarik disimpulkan bahwa var X1 (*Online / Ecommerce*) secara parsial berpengaruh pada daya saing kuliner tradisional di Sulawesi Selatan.	Saran untuk menambahkan dataset pada pengujian.	Penulis menggunakan penelitian sebagai dasar terhadap persaingan produk, penelitian berbeda dalam penggunaan algoritma dan metodenya.
4	Strategi Pengembangan Usaha Mikro di Kota Samarinda	Fajar Febrian Putranto, Zhikry Fitrian, Bramantyo Adi Nugroho, Eka Nor Santi, Noor Wahyuningsih, Puput Wahyu	Melakukan analisis SWOT untuk menentukan urutan strategi yang paling prioritas dalam pengembangan usaha mikro di Kota Samarinda.	Terdapat beberapa kesimpulan tentang strategi prioritas, diantaranya prioritas pertama dalam pengembangan usaha mikro di Kota Samarinda yaitu	Saran perlu adanya pembahasan lebih spesifik mengenai strategi pengembangan usaha mikro berdasarkan jenis usaha.	Penulis menggunakan penelitian sebagai dasar terhadap persaingan produk, penelitian berbeda dalam penggunaan algoritma dan metodenya.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
		Budiman, Adi Hendro Purnomo, Riset Inossa, 2019		melakukan promosi produk, prioritas kedua yaitu melakukan pemasaran produk melalui *e-commerce*, prioritas ketiga mendirikan inkubator bisnis teknologi, prioritas keempat mendorong pihak swasta untuk berperan aktif, prioritas kelima yaitu melakukan penguatan regulasi daerah.		
5	Industri *E-commerce* dalam Menciptakan Pasar yang Kompetitif Berdasarkan Hukum Persaingan Usaha	Melisa Setiawan Hotana, Jurnal Hukum Bisnis Bonum Commune, 2018	Melakukan amandemen pada Undang-Undang Nomor 5 Tahun 1999 Mengenai Larangan Tentang Praktek Monopoli Serta Persaingan Usaha Tidak Sehat.	*E-commerce* menciptakan pasar kompetitif dan kondusif, memberikan peluang kepada pelaku usaha dengan mudah di internet, perlu adanya pengaturan dalam UU Nomor 5 Tahun 1999 mengenai peranan pada	Tidak dijelaskan dengan detail tentang data pada penelitian.	Penulis menggunakan penelitian sebagai dasar terhadap persaingan produk, penelitian berbeda dalam penggunaan algoritma dan metodenya

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				Komisi Pengawas Persaingan Usaha dalam industri *e-commerce* untuk menghentikan monopoli serta persaingan usaha yang tidak sehat.		
6	*The Text mining Handbook: Advanced Approaches to Analyzing Unstructured Data*	Ronen Feldman, James Sanger, Cambridge University Press, 2007	Buku tersebut membahas pengantar menyeluruh untuk *text mining*, yang mencakup arsitektur umum sistem penambangan teks, serta dengan teknik utama yang digunakan.	Dijelaskan pada beberapa chapter dengan membahas teknik dan konsep dalam melakukan *text mining* serta teknik dalam melakukan *text preprocessing*. Membahas proses penambangan data, pemrosesan bahasa alami, pembelajaran mesin, dan pengambilan informasi.	Saran membandingkan algoritma *stemming* pada pembahasan *text preprocessing*.	Penulis menggunakan intisari dari penelitian ini, yaitu mengambil konsep *text mining* dan *text preprocessing*. Penulis memiliki tujuan dan pemakaian algoritma yang berbeda untuk memproses hasil *text mining* tersebut.
7	*Sentiment Analysis of Positive and Negative of YouTube*	Muhammad, Abbi Nizar, Saiful Bukhori, Priza Pandunata,	Menganalisis sentimen pada komentar di Youtube untuk mengetahui	Hasil klasifikasi algoritma Naïve Bayes dan Support Vector Machine (SVM)	Dari penelitian ini menggunakan dataset dari hasil pengolahan algoritma *stemming*	Penulis mengambil intisari dari penelitian sebelumnya yaitu tentang tahapan *text preprocessing*, pembeda

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	Comments Using Naïve Bayes – Support Vector Machine (NBSVM) *Classifier*	International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE), 2019	klasifikasi komentar tersebut positif atau negatif.	memiliki hasil yang optimal ketika pelatihan data tersebut memiliki jumlah data yang bervariasi, kombinasi algoritma Naïve Bayes dan SVM menghasilkan tingkat akurasi yang lebih baik serta kinerja yang lebih baik dengan penggunaan data skala 7:3, yaitu 70,00% data pelatihan dan 30,00% data pengujian sehingga nilai tes kinerja tertinggi, yaitu presisi 91,00%, recall 83,00% dan skor f1 87,00%.	Tala, saran untuk membandingkan dataset dari pemrosesan *stemming* lainnya untuk mengetahui perbedaan akurasi kombinasi Naïve Bayes dan Support Vector Machine.	penelitian ini yaitu perbedaan algoritma yang dipakai dan perbedaan tujuan penelitian.
8	Perbandingan Algoritma *Stemming* Porter Dengan Arifin Setiono untuk Menentukan	Dian Novitasari, Jurnal *String*, 2016	Bertujuan untuk membandingkan dua *stemmers* Indonesia, yaitu Porter dan Arifin Setiono	Dengan 40 sampel dokumen Bahasa Indonesia, menghasilkan *Exact Match* yaitu *stemmer* Porter sebesar 90,00%, *stemmer* Arifin	Perlunya penelitian yang lebih detail untuk mengetahui hasil tingkat efektifitas dari masing-masing algoritma tersebut	Penulis mengambil intisari dari penelitian sebelumnya yaitu dalam pemilihan algoritma *stemmer* yang paling optimal. Perbedaan terletak pada tujuan penelitian serta metode

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	Tingkat Ketepatan Kata Dasar			Setiono95.00%. *Unchange* yaitu *stemmer* Porter 2,50%, *stemmer* Arifin Setiono 2,50%. *Spelling Exception stemmer* Porter 7,50%, *stemmer* Arifin Setiono 0%. Over*stemming stemmer* Porter 0%, *stemmer* Arifin Setiono 2,50%. Disimpulkan bahwa algoritma Arifin & Setiono memiliki presisi lebih tinggi dibandingkan dengan *stemming* Porter.	seperti pemakaian waktu pemrosesan disetiap algoritma *stemmer* yang digunakan.	yang digunakan pada penelitian.
9	Perbandingan Algoritma *Stemming* Porter dengan Algoritma Nazief & Adriani untuk *Stemming* Dokumen Teks Bahasa Indonesia	Ledy Agusta, Konferensi Nasional Sistem dan Informatika, 2009	Membandingkan efektivitas dan efisiensi dari dua *stemmer* Indonesia, yaitu algoritma Porter serta algoritma Nazief & Adriani.	Menguji pada 30 dokumen, algoritma Porter memerlukan waktu lebih singkat yaitu dengan rata-rata waktu 0,37 detik dibandingkan dengan *stemming* menggunakan	Pada landasan teori dijelaskan tentang detail aturan disetiap algoritma, namun pada pembahasan tidak dijelaskan secara detail pada data yang dipakai	Penulis mengambil intisari dari penelitian sebelumnya yaitu dalam pemilihan algoritma *stemmer* yang paling optimal. Perbedaan terletak pada tujuan penelitian serta metode

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				algoritma Nazief & Adriani yaitu dengan rata-rata waktu 18,72 detik. Namun algoritma Nazief & Adriani menghasilkan keakuratan lebih baik yaitu dengan presisi 84,27% dibandingkan dengan *stemming* algoritma Porter 70,56%.	tentang penggunaan aturan tersebut.	yang digunakan pada penelitian.
10	Implementasi dan Analisis Algoritma *Stemming* Nazief & Adriani dan Porter Pada Dokumen Berbahasa Indonesia	Dwi Wahyudi, Teguh Susyanto, Didik Nugroho, Jurnal Ilmiah SINUS	Membandingkan antara algoritma Nazief & Adriani dan algoritma Porter untuk pada proses tahapan *stemming* pada teks ber-Bahasa Indonesia, sehingga nantinya diketahui algoritma yang lebih cepat, lebih akurat dan mengetahui algoritma yang lebih	Menggunakan 2.132 kata yang digunakan untuk pengujian menyatakan bahwa algoritma Nazief & Adriani menghasilkan kebenaran kata dasar sebanyak 2.031 kata, sedangkan hasil untuk algoritma Porter 1.687 kata. Tingkat kesalahan algoritma *stemmer* Nazief & Adriani	Karena pada algoritma yang dibandingkan menggunakan kamus kata dasar maka untuk mengurangi tingkat kesalahan pada saat proses *stemming*, sebaiknya menggunakan kamus kata dasar yang lengkap.	Penulis mengambil intisari dari penelitian sebelumnya yaitu dalam pemilihan algoritma *stemmer* yang paling optimal. Perbedaan terletak pada tujuan penelitian serta metode yang digunakan pada penelitian.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
			banyak melakukan kesalahan *stemming*.	5,00%, sedangkan algoritma Porter 21,00%. Efisiensi waktu proses untuk algoritma Porter menghasilkan lebih baik yaitu memiliki 12,38 detik sedangkan algoritma Nazief & Adriani memiliki 22,166 detik namun algoritma Nazief & Adriani menghasilkan nilai yang lebih baik untuk *stemming* pada dokumen berbahasa Indonesia dengan akurasi 95,26% sedangkan algoritma Porter 79,13%.		
11	Studi Perbandingan Algoritma - Algoritma *Stemming* Untuk	Manase Sahat H Simarangkir, Jurnal Inkofar, 2017	Melakukan pengujian untuk membandingkan algoritma Nazief & Adriani, Arifin &	Rata-rata setiap pengujian yaitu algoritma Nazief & Adriani memiliki waktu proses 5,15 detik,	Peneliti hanya menggunakan dataset dari kata Kamus Besar Bahasa Indonesia schingga	Penulis mengambil intisari dari penelitian sebelumnya yaitu dalam pemilihan algoritma *stemmer* yang paling optimal. Perbedaan

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	Dokumen Teks Berbahasa Indonesia		Setiono, Vega, dan Tala	akurasi 97,93%, over*stemming* 1,40% dan under*stemming* 0,40%. Algoritma Arifin & Setiono memiliki waktu proses 15,20 detik, akurasi 92,09%, over*stemming* 6,10% dan under*stemming* 1,56%. Algoritma Vega memiliki waktu proses 0,08 detik, akurasi 63,49%, over*stemming* 30,33% dan under*stemming* 6,06%. Algoritma Tala memiliki waktu proses 0,22 detik, akurasi 78,27%, over*stemming* 19,09% dan under*stemming* 1,70%, dari hasil pengujian yang didapatkan disimpulkan bahwa untuk algoritma yang	data yang diolah berupa kata per kata, peneliti tidak melakukan pengujian pada sebuah kalimat atau kumpulan kata yang besar yang tidak terstruktur.	terletak pada tujuan penelitian serta metode yang digunakan pada penelitian.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				menggunakan aturan imbuhan yaitu algoritma Tala sedangkan untuk algoritma Nazief & Adriani merupakan algoritma terbaik.		
12	Perbandingan Algoritma *Stemming* Nazief & Adriani, Porter dan Arifin Setiono untuk Dokumen Teks Bahasa Indonesia	Oppie Rezalina, Journal of Undergraduate Thesis, Universitas Muhammadiyah Jember, 2016	Melakukan perbandingan algoritma Nazief & Adriani, algoritma Porter serta algoritma Arifin Setiono untuk proses *stemming* teks berbahasa Indonesia.	Algoritma Nazief & Adriani lebih unggul dalam hal kecepatan sedangkan untuk akurasi dibandingkan dengan dua algoritma lainnya pada pengujian 10 dokumen teks dengan perbandingan rata-rata akurasi pada setiap algoritma yaitu algoritma Nazief & Adriani, algoritma Porter, algoritma Arifin & Setiono yaitu 87,00%:85,80%:86,80% sedangkan untuk kecepatan perbandingan	Tidak dijelaskan secara detail tentang pemakaian data uji yang digunakan, sehingga tidak diketahui performa algoritma tersebut berdasarkan dokumen yang kecil atau dokumen yang besar.	Penulis mengambil intisari dari penelitian sebelumnya yaitu dalam pemilihan algoritma *stemmer* yang paling optimal. Perbedaan terletak pada tujuan penelitian serta metode yang digunakan pada penelitian.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				rata-ratanya yaitu 16,75 detik:40,39 detik: 53,22 detik.		
13	Analisis Perbandingan Algoritma Rabin-Karp Dan Levenshtein Distance Dalam Menghitung Kemiripan Teks	Andry Hery Purba, Zakarias Situmorang, Jurnal Teknik Informatika Unika St. Thomas (JTIUST), 2017	Menghitung tingkat *similarity* berdasarkan penerapan algoritma Rabin-Karp serta Levenshtein Distance.	Algoritma Rabin-Karp pencocokkan multiple pattern lebih bagus jika dibandingkan dengan single pattern, hasil pengujian algoritma Rabin-Karp yaitu 79,85% sedangkan algoritma Levenshtein Distance memiliki tingkat plagiasi 66,67%.	Penulis tidak menggunakan dataset yang sama saat melakukan pengujian kedua algoritma schingga perbandingan datanya tidak sesuai.	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya penulis menggunakan perbandingan algoritma yang berbeda.
14	Analisis Dan Implementasi Algoritma Rabin-Karp Dan Algoritma *Stemming* Nazief & Adriani Pada Sistem Pendeteksi Plagiat Dokumen Teks Berbahasa Indonesia	Yosef Agung Wicaksono, Suyanto, Telkom University, 2012	Melakukan pendeteksian plagiat terhadap dokumen teks berbahasa Indonesia.	Besarnya akurasi dari sistem tergantung pada pemilihan nilai k-grams dan pemakaian *preprocessing* maka hasil nilai akurasi algoritma Rabin-Karp mencapai 90,00% untuk tipe plagiat aktif-pasif, *carbon copy*, ubah struktur kata dan	Tidak dijelaskan secara detail hasil dari perbandingan penerapan algoritma *Stemming* Nazief & Adriani atau tidak memakai *stemming* pada data yang diuji schingga tidak diketahui perbedaan tingkat akurasinya.	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya penulis menggunakan algoritma perbandingan yang berbeda dan tujuan penelitian tidak sama.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				tambah kata sedangkan karena sistem yang dibuat tidak menggunakan *synonym recognition* maka untuk tipe sinonim memiliki akurasi 72,76%.		
15	Perbandingan Pengunaan *Stemming* pada Deteksi Kemiripan Dokumen Menggunakan Metode Rabin Karp dan Jaccard *Similarity*	Adji Sukmana, Kusrini, Andi Sunyoto,Seminar Nasional Teknologi Informasi dan Multimedia, 2018	Membandingkan dan menganalisa pemakaian algoritma *Stemming* Nazief & Adriani pada penerapan algoritma Rabin-Karp untuk mendeteksi *similarity* dokumen.	Algoritma Rabin Karp pada tahap *stemming* menggunakan algoritma *Stemming* Nazief & Adriani menghasilkan kesimpulan yaitu mempercepat waktu eksekusi serta hasil *similarity* yang hampir sama yaitu persentase perbandingan antara algoritma Rabin Karp murni dan algoritma Rabin Karp dengan algoritma *Stemming* Nazief & Adriani pada percobaan dokumen 1 100%:100%, pada	Tidak diperlihatkan lebih detail tentang perhitungan algoritma Rabin-Karp dan perhitungan Jaccard *Similarity* pada pembahasan hanya menampilkan hasil setelah perhitungan.	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya penulis menggunakan algoritma yang berbeda.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				percobaan dokumen 2 74,87%:72,09%, pada percobaan dokumen 3 48,39%:45,69%, pada percobaan dokumen 4 23,60%:20,59%.		
16	*Text Documents Plagiarism Detection using* Rabin-Karp *and* Jaro-Winkler Distance *Algorithms*	Brinardi Leonardo, Seng Hansun, Indonesian Journal of Electrical Engineering and Computer Science, 2017	Membandingkan efektifitas dari perbandingan algoritma Rabin-Karp dan Jaro-Winkler pada pendeteksi kemiripan pada dokumen teks.	Algoritma Rabin Karp lebih efektif daripada algoritma Jaro-Winkler Distance, pada beberapa percobaan algoritma Rabin Karp mendapatkan persentase kemiripan yang lebih tinggi daripada Jaro-Winkler Distance yaitu 51,00%:35,00%. Untuk hasil pada pengujian waktu pemrosesan, algoritma Rabin-Karp memiliki nilai rata-rata 0,59 menit sedangkan algoritma Jaro-Winkler Distance memiliki nilai rata-rata 0,99 menit	Tidak ada proses *text preprocessing* sehingga teks tidak dibersihkan terlebih dahulu.	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya berkaitan dengan metode yang digunakan untuk mendeteksi *similarity* serta pada penelitian sebelumnya tidak menggunakan algoritma *stemming*.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
17	Analisis Performansi Algoritma Winnowing dan Algoritma Manber untuk Deteksi Kesamaan Dokumen Teks Berbahasa Indonesia	Bambang Imam Hermawan, Universitas Komputer Indonesia, 2015	Melakukan perbandingan kecepatan, ketepatan algoritma Winnowing terhadap algoritma Manber untuk mendeteksi kesamaan antar dokumen.	Kecepatan Manber lebih baik dengan nilai proses 0,10 detik dari Winnowing 0,33 detik namun pada pengujian ketepatan Winnowing lebih baik dari Manber karena dari perhitungan manual menghasilkan nilai kemiripan 86,15% pada pengujian algoritma winnowing mendapatkan persentase nilai pengujiannya 86,12% sedangkan untuk algoritma manber memiliki nilai pengujiannya 85,71%.	Belum dapat menunjukan posisi dari karakter yang terbagi menjadi beberapa gram yang terdeteksi sama sehingga dibutuhkan pengembangan dalam tahap implementasi.	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya penulis tidak menggunakan algoritma Manber namun digantikan oleh algoritma Ratcliff/Obershelp serta penciiti sebelumnya tidak menggunakan algoritma *stemming*.
18	*Comparison Between Fingerprint and* Winnowing *Algorithm to Detect Plagiarism Fraud on* Bahasa	Agung Toto Wibowo, Kadek W. Sudarmadi, Ari M. Barmawi, International Conference of Information and	Mendeteksi kesamaan dokumen pada tugas akhir fakultas informatika	Algoritma *Fingerprint* lebih tinggi 92,80% dibandingkan dengan algoritma Winnowing 91,80%, namun algoritma Winnowing memiliki hasil kinerja	Pada Diagram dijelaskan ada proses mod p (bilangan prima) namun pada hasil dan pembahasan tidak dijelaskan nilai bilangan prima yang	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya penulis menggunakan algoritma

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	Indonesia *Documents*	Communication Technology (ICoICT), 2013		yang lebih baik serta memiliki hasil yang lebih stabil pada spesifik kata kunci dengan menghasilkan nilai spesifik pada Distinct *Term* dan *Term* Frequency yaitu 34,70% dan 37,10% sedangkan metode *fingerprint* menghasilkan nilai spesifik pada Distinct *Term* dan *Term* Frequency yaitu 31,70% dan 33,60%.	dipakai pada penelitian.	yang berbeda dan tujuan penelitian tidak sama.
19	Perbandingan Algoritma Winnowing dengan Algoritma Rabin Karp untuk Mendeteksi Plagiarisme pada Kemiripan Teks Judul Skripsi	Nur Alamsyah, Technologia, 2017	Membandingkan akurasi dan efektifitas algoritma Rabin Karp dengan algoritma Winnowing dalam pendekatan pendeteksi plagiarisme.	Pada ujicoba ke 8 algoritma Winnowing dengan nilai n-gram = 9 dan *window* = 3 , menghasilkan proses waktu 0,03 detik dengan tingkat kemiripan terkecil yaitu 32,60% dibandingkan dengan algoritma	Hanya menggunakan data uji berupa teks dari judul skripsi saja, dari itu maka pengujian hanya dilakukan pada teks berukuran kecil sehingga tidak ada perbandingan jika dalam kasusnya	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya penulis membandingkan algoritma yang berbeda serta peneliti sebelumnya tidak

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				Rabin-Karp pada ujicoba yang sama yaitu memerlukan waktu 0,03 detik dengan tingkat kemiripan 54,78% dengan hasil tersebut disimpulkan tingkat presentase algoritma Winnowing lebih baik.	menggunakan teks berukuran besar.	menggunakan algoritma *stemming*.
20	Analisa Perbandingan Sistem Pendeteksian Kemiripan Judul Skripsi Menggunakan Algoritma Winnowing Dan Algoritma Rabin Karp	Lelawati Sibarani , Magdalena , Abdi Dharma, Riset dan E-Jurnal Manajemen Informatika Komputer, 2019	Mendeteksi plagiarisme atau kesamaan pada judul skripsi.	Merancang sebuah aplikasi / sistem yang mampu memproses atau membandingkan dua judul skripsi menggunakan Algoritma Winnowing serta dengan Algoritma Rabin Karp serta menghasilkan tingkat akurasi kemiripan judul mendekati 75,00%.	Pada tahap hasil dan pembahasan penelitian hanya menggunakan 2 data uji untuk di terapkan pada perhitungan algoritma, sedangkan tidak dijelaskan / ditampilkan jumlah dataset yang diuji.	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya penulis menggunakan algoritma perbandingan yang berbeda dan tujuan penelitian tidak sama.
21	*Comparison of* Jaro-Winkler *and* Ratcliff/Obershelp	Ilya Ilyankou	Membandingkan efektifitas dua algoritma pencocokan *string* Jaro-Winkler	Algoritma Ratcliff/Obershelp 4% lebih efisien daripada algoritma jarak Jaro-	Peneliti menggunakan data percobaan kesalahan eja *string* untuk menguji tingkat	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	algorithms in spell check		dan Ratcliff/Obershelp dalam aplikasi pemeriksaan ejaan.	Winkler. Algoritma Ratcliff/Obershelp bekerja sangat baik ketika ada salah ketik pada karakter pertama atau terakhir dalam *string*. Algoritma Ratcliff/Obershelp menunjukkan hasil 4,00% -18,60% lebih baik saat memproses daftar 53 kata yang salah eja.	pengujian dalam kesalahan pengetikan kata namun peneliti tidak menggunakan percobaan rangkaian kata untuk menguji kedua algoritma tersebut.	akan dilaksanakan, diantaranya penulis membandingkan algoritma yang berbeda serta peneliti sebelumnya tidak menggunakan algoritma *stemming*.
22	Analisis Perbandingan Algoritma Rabin-Karp dan Ratcliff/Obershelp untuk Menghitung Kesamaan Teks dalam Bahasa Indonesia	Bustami Yusuf, Sari Vivianie, Jiwa Malem Marsya, Zuhra Sofyan, Seminar Nasional APTIKOM (SEMNASTIK), 2019	Mendeteksi plagiarisme pada portal berita *online* dan membandingkan algoritma Rabin-Karp dan Ratcliff/Obershelp.	Algoritma Ratcliff/Obershelp mendekati persentase data masukan dibandingkan Algoritma Rabin-Karp, selisih nilai *similarity* rata-rata dua algoritma tersebut 3,00%. Tata letak kalimat mempengaruhi nilai *similarity* algoritma Rabin-Karp,	Pada penelitian tidak menggunakan algoritma *stemming* sehingga kata yang dicocokkan tidak dirubah ke kata dasar.	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya penulis tidak menggunakan algoritma Rabin-Karp namun digantikan oleh algoritma Winnowing serta peneliti sebelumnya tidak

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				sedangkan algoritma Ratcliff/Obershelp sama sekali tidak dipengaruhi. Algoritma Rabin-Karp memiliki tingkat konsistensi yang rendah dibandingkan algoritma Ratcliff/Obershelp. Pada pengujian 10 data Algoritma Ratcliff/Obershelp memiliki nilai rata-rata 1,33 *milisecond* dengan 2 nilai kemiripan (94 dan 95) sedangkan Algoritma Rabin-Karp memiliki nilai rata-rata 4,2 *milisecond* dan menghasilkan 5 nilai kemiripan yang berbeda (93, 94, 95, 96 dan 97) dengan k-gram 2, 4, 6, 8 dan 10 terhadap pengujian yang sama.		menggunakan algoritma *stemming*.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
23	Implementasi Deteksi Plagiarisme Menggunakan Metode N-Gram dan Jaccard *Similarity* Terhadap Algoritma Winnowing	Sunardi, Anton Yudhana, Iif Alfiatul Mukaromah, Jurnal TRANSMISI, 2018	Mendeteksi plagiarisme / kesamaan dokumen karya ilmiah.	Hasilnya dengan melakuan pengujian dengan pembeda nilai n-gram dan w-gram diperoleh hasil nilai dengan n-gram 3 dan nilai dengan w-gram 3 mempunyai hasil nilai *similarity* 35,71%, nilai dengan n-gram 5 dan nilai dengan w-gram 3 mempunyai hasil nilai *similarity* 41,67%, nilai dengan n-gram 4 dan nilai dengan w-gram 5 mempunyai hasil nilai *similarity* 45,45%, nilai dengan n-gram 4 dan nilai dengan w-gram 4 mempunyai hasil nilai *similarity* 41,67%, nilai dengan n-gram 6 dan nilai dengan w-gram 6 mempunyai hasil nilai *similarity* 18,18%.	Peneliti hanya menggunakan contoh pengujian data sebanyak satu kali, tidak dijelaskan dataset yang digunakan darimana saja serta pada tahap *text preprocessing* tidak ada tahap *stemming* untuk mengubah *string* menjadi kata dasar.	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya penulis menggunakan perbandingan algoritma sedangkan penelitian sebelumnya hanya mengimplementasi algoritma.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
24	Rancang Bangun Aplikasi Deteksi Kemiripan Dokumen Teks Menggunakan Algoritma Ratcliff/Obershelp	Yudhy Lady Joane, Alicia Sinsuw, Agustinus Jacobus, E-Journal Teknik Informatika, 2017	Menerapkan algoritma Ratcliff/Obershelp untuk mendeteksi pola kesamaan *string* yang berbeda.	Banyaknya karakter didalam dokumen mempengaruhi waktu eksekusi dari proses pengidentifikasian kesamaan dokumen, dan letak kalimat yang sama penempatannya mempengaruhi nilai kesamaan atau *similarity* dari pendeteksian, pada prediksi dan hasil nilai perbandingan kesamaan dokumen 1 terhadap 11 dokumen memiliki rata-rata perbandingan antara prediksi dan nilai yaitu 50,00%:54,90%.	Pada tahap *stemming* tidak menggunakan algoritma *stemming* untuk menambah peforma pencarian kata dasar.	Perbedaan antara hasil dari penelitian sebelumnya / yang telah dilakukan dengan penelitian yang akan dilaksanakan, diantaranya penulis menggunakan algoritma *stemming* sedangkan pada penciiti sebelumnya tidak menggunakan algoritma *stemming*.
25	Ekstraksi Data Produk *E-Marketplace* Sebagai Strategi Pengolahan Segmentasi Pasar	Ade Surahman, A. Ferico Octaviansyah, Dedi Darwis, Jurnal Sistem Informasi, 2020	Melakukan ekstraksi data dengan web crawler untuk menyajikan segmentasi aktivitas	Pengujian prototype pada ekstraksi data produk *marketplace* dari pengolahan 250 responden yaitu pada kriteria kualitas	Tidak dijelaskan secara spesifik dalam melakukan pengujian ekstraksi pada penentuan jenis	Penulis menggunakan penelitian sebagai dasar terhadap persaingan produk, penelitian berbeda dalam penggunaan algoritma dan metodenya.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	Menggunakan *Web Crawler*		peletakan produk di *online* shop.	informasi mendapatkan skor aktual 80,00% untuk kriteria kualitas sistem mendapatkan skor aktual 79,00% untuk kualitas layanan mendapatkan skor aktual 77,00% untuk kriteria penggunaan mendapatkan skor aktual 79,00% untuk kriteria kepuasan pengguna mendapatkan skor aktual 78,00%. Maka dari keseluruhan untuk kesuksesan informasi segmentasi pasar yaitu sebesar 79,00%.	produk yang akan diteliti.	
26	Daya Saing Ott (*Over The Top*) *E-commerce* Indonesia Dalam Menghadapi Persaingan Global	Abdullah, Jurnal Ilmu Ekonomi Islam, 2018	Melakukan kajian daya saing OTT *e-commerce* untuk memetakan kondisi dan kemampuan *e-commerce* Indonesia	Mengkaji tentang perlunya penerapan dari rekomendasi laporan BEPS dengan memperbaiki peraturan pajak, dan belajar dari	Saran untuk lebih spesifik dalam menunjukkan data misalnya pada *e-commerce* dengan produk tertentu	Penulis menggunakan penelitian sebagai dasar terhadap persaingan produk, penelitian berbeda dalam penggunaan algoritma dan metodenya.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
			dalam menghadapi tantangan dan persaingan Global.	pengalaman dari Negara lain tentang perusahaan *e-commerce* multinasional dan penyedia layanan OTT.	sehingga akan diketahui persaingan yang tepat.	
27	Implementasi *E-commerce* Sebagai Media Penjualan *Online*	Shabur Miftah Maulana, Heru Susilo, Riyadi, Jurnal Administrasi Bisnis, 2015	Melakukan penerapan penggunaan *e-commerce* dengan platform opencart terhadap Pastbrik Malang	Melakukan perubahan penjualan konvensional/*offline* karena kurang efektif dan efisien dengan membuat *e-commerce* pada Pastbrik Malang untuk memperoleh hasil yang maksimal serta dapat bersaing dengan toko yang sejenis.	Tidak dijelaskan dengan detail tentang data yang diolah pada penelitian.	Penulis menggunakan penelitian sebagai dasar terhadap persaingan produk, penelitian berbeda dalam penggunaan algoritma dan metodenya.
28	Pengembangan Model *CYBER CLUSTER E-COMMERCE* Berbasis CMS dan SEO Produk UMKM	Dwi Agus Diartono, Yohanes Suhari, Aji Supriyanto, IJCCS (Indonesian Journal of Computing and	Mengoptimasi *e-commerce* produk UMKM hingga pasar global dengan teknik Search Engine Optimisazion (SEO)	Peneliti melakukan penerapan teknik seo seperti penerapan pada *meta description* serta *keywords*, penempatan *tag title* dan memberikan *backlink*. Dari statistik pada alexa rank menunjukkan rank	Tidak dijelaskan secara detail bebereapa *website* yang di uji, hanya menampilkan pengujian pada satu *website*.	Penulis menggunakan penelitian sebagai dasar terhadap persaingan produk, penelitian berbeda dalam penggunaan algoritma dan metodenya.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
		Cybernetics Systems), 2015		*website* yang diuji memiliki rank 40.867 di Indonesia.		
29	Inovasi Produk Melalui Strategi Imitasi Dalam Menghadapi Persaingan Produk Impor (Implementasi Strategi Imitasi Pada Studi Kasus Edam Burger Di Depok)	Hannisa Rahmaniar Hasnin, Jurnal Ilmiah Inovator, 2016	Menentukan inovasi pada perubahan strategi pemasaran produk.	Hasil perhitungan regresi linear berganda terhadap 80 data kuisoner memperoleh beberapa hasil koefisien. yaitu pada variabel intercept 7,35 variabel imitasi 0,02 variabel inovasi 0,42 menunjukkan bahwa terdapat pengaruh strategi imitasi terhadap daya saing, dan inovasi terhadap daya saing.	Tidak dijelaskan secara detail tentang data hasil wawancara dan kuisoner yang diproses pada penelitian.	Penulis menggunakan penelitian sebagai dasar terhadap persaingan produk, penelitian berbeda dalam penggunaan algoritma dan metodenya.
30	*Automated* Bahasa Indonesia *essay evaluation with latent semantic analysis*	A Amalia, D Gunawan, Y Fithri, I Aulia, International Conference on Computing and Applied	Membuat konsep evaluasi esai otomatis dengan perbandingan kesamaan jawaban mahasiswa dengan kunci jawaban dari dosen.	Tahap pengujiannya menggunakan *Input*an jawaban mahasiswa dengan *Input*an jawaban dosen dan diproses pada tahap *preprocessing* dan dihitung tingkat kemiripannya	Tidak dijelaskan jumlah detail pengujian setiap jawaban, dalam tahap penjelasan hanya menghitung percontohan 1 dokumen yang	Penulis menggunakan algoritma yang berbeda pada peneliti sebelumnya dan tujuan penelitian tidak sama, penulis hanya mengambil intisari yaitu tentang penggunaan

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
		Informatics, 2018		menggunakan Cosine *Similarity*, hasilnya dari 60 jawaban yang diuji dengan skenario yang ditentukan, menghasilkan akurasi sistem sebesar 83,3%.	diteliti, namun tidak disajikan hasil detail dari perhitungan semua dokumen dan hanya menampilkan hasil rata-rata akurasi.	*stemming* Nazief & Adriani.
31	Implementasi Cosine *Similarity* dan Algoritma Smith-Waterman untuk Mendeteksi Kemiripan Teks	Radiant Victor Imbar, Adelia, Mewati Ayub, Alexander Rehatta, Jurnal Informatika, 2014	Mengembangkan software yang menerapkan cosine *similarity* yang digunakan untuk mengukur teks pada kemunculan sebuah kata pada teks dan penerapan algoritma Smith-Waterman yang bekerja sebagai penghitung kemiripan teks berdasarkan pola urutan kata.	Software yang dibuat sebagai pendeteksi *similarity* teks dari identifikasi sangat mirip hingga sangat tidak mirip berdasarkan kemunculan sebuah kata yang diproses dengan menggunakan cosine *similarity*. Dengan nilai pengujian kemiripan terhadap kemunculan kata dan kemiripan terhadap urutan kata pada teks empat 24,66% dan 16,67% terhadap teks satu, 39,54% dan	Saran menggunakan dataset yang bervariasi dalam jumlah karakter dan jumlah katanya, karena pada penelitian tersebut menggunakan 4 dokumen yang hampir sama dalam jumlah karakter dan jumlah katanya.	Penulis hanya mengambil intisari dari pengertian dalam pendeteksian *similarity*, pada tujuan dan algoritma yang digunakan menggunakan algoritma yang berda dan tujuannya tidak sama.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				20,00% terhadap teks dua, 60,99% dan 13,33% terhadap teks tiga, 100% dan 100% terhadap teks empat.		
32	Investigating *Parallelization of Cross-language Plagiarism Detection System Using the* Winnowing *Algorithm in Cloud Based Implementation*	Anak Agung Putri Ratna, F. Astha Ekadiyanto, Ihsan Ibrahim, EEE 10th International Conference on Awareness Science and Technology, iCAST 2019	Tentang pendeteksi *similarity* lintas bahasa dari bahasa Indonesia terhadap bahasa Inggris menggunakan perbandingan 2 *core* processor, 4 *core* processor, dan 8 *core* processor.	Jumlah *core processor* yang lebih tinggi dapat memproses jumlah data yang lebih besar dengan nilai kecepatan yang lebih baik, di dapatkan nilai kecepatan 3,52 detik pada pengujian paper ke 3 dengan 22 paragraf dan 1.000 referensi dokumen pada pengujian dengan *processor* 8 *core*, sehingga tingkat efisiensi untuk varietas tersebut 50,91%.	Saran menggunakan algoritma *stemming* untuk proses merubah kata dasar.	Penulis menerapkan algoritma yang sama namun tujuan pada penelitian sebelumnya berbeda.
33	Implementasi Algoritma Winnowing untuk Mendeteksi	Rizky Maulana Fauzi, Julian Chandra Wibawa	Mengimplementasikan algoritma winnowing untuk mendeteksi kemiripan teks pada	Hasil penelitian yang dilakukan oleh peneliti dalam menguji 6 pembanding dalam	Dataset yang digunakan kurang jelas pada jumlah data ataupun banyaknya	Penulis menggunakan algoritma yang sama namun memiliki tujuan penelitian yang berbeda,

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Penciiti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	Kemiripan Teks pada Artikel		artikel tugas akhir mahasiswa pada program studi manajemen informatika Universitas Komputer Indonesia.	deteksi kesamaan memperoleh hasil 100%, 7,95%, 7,48%, 5,06%, 4,31%, dan 4,08%.	kata setiap dokumen yang di uji, namun peneliti hanya menampilkan 6 pengujian pada gambar hasil penelitian dan tidak dibahas secara lengkap tentang berapa banyaknya data uji.	dan juga penulis menggunakan algoritma *stemming* Nazief & Adriani sedangkan penelitian sebelumnya tidak menggunakan algoritma *stemming*.
34	Rancang Bangun Aplikasi Kemiripan Dokumen Dengan Sumber – Sumber Internet	Virggi Eko Jacob, Arie S. M. Lumenta, Agustinus Jacobus, Jurnal Teknik Informatik, 2019	Membuat sebuah *software* aplikasi yang dapat mengukur tingkat *similarity* tugas mahasiswa dengan mengujinya dari wikipedia dengan algoritma Ratcliff/Obershelp.	Setelah dilakukan pengujian terhadap 5 dokumen dengan artikel pada wikipedia dihasilkan terhadap dokumen 1 dengan prediksi nilai 0% dan memperoleh nilai 14,00%, sedangkan untuk dokumen 2 dengan prediksi nilai 25,00% dan memperoleh nilai pengujian 35,00%,	Saran untuk menambahkan algoritma *stemming* dan menguji algoritma sejenis untuk mengetahui hasil perbandingan akurasi.	Penulis menggunakan algoritma yang sama namun pada tahap *text preprocessing* pada penelitian sebelumnya tidak menggunakan algoritma *stemming*.

Tabel 2.1. Matriks literatur review dan posisi penelitian
Analisis Penerapan Algoritma Stemming Nazief & Adriani pada Perbandingan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Pendeteksi Kesamaan Produk di Marketplace (Lanjutan)

No	Judul	Pencliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				sedangkan untuk dokumen 3 dengan prediksi nilai 50,00% dan memperoleh nilai pengujian 67,00%, sedangkan untuk dokumen 4 dengan prediksi nilai pengujian 75,00% dan memperoleh nilai pengujian 72,00%, dan untuk dokumen 5 dengan prediksi nilai 100% dan memperoleh nilai pengujian 54,00%.		

2.3. Landasan Teori

2.3.1. Pengumpulan Data

Populasi merupakan wilayah yang bersifat generalisasi yang terdiri berdasarkan obyek ataupun subyek serta memiliki kualitas dan karakteristik yang ditetapkan oleh peneliti untuk nantinya dipelajari dan kemudian didapatkan kesimpulannya.

Sampel merupakan bagian yang berasal dari jumlah serta karakteristik yang dipunyai oleh populasi, jika populasi tersebut besar tetapi peneliti tidak mungkin mempelajari semua populasi tersebut, misalnya keterbatasan pada dana, tenaga ataupun waktu, maka dengan itu peneliti bisa menggunakan sampel yang diambil dari populasi tersebut. Sehingga nantinya apa yang dipelajari dari sampel itu, kesimpulannya akan dapat berlaku untuk dapat mewakili populasi (Sugiyono, 2016).

Populasi pada penelitian ini adalah produk digital yang dijual atau dipublikasi yang terdapat pada situs p-store.net. Mengingat jumlah total populasi tidak diketahui sehingga untuk menentukan jumlah pada ukuran sampel minimum harus cukup mewakili populasi responden yang diteliti maka dari itu untuk menentukan jumlah dari besaran sampel yang tidak diketahui jumlah populasinya dihitung dengan menggunakan rumus Wibisono (Riduwan dan Akdon, 2013) :

$$n = \left(\frac{Z_{\alpha/2} * \sigma}{e}\right)^2 \quad (1)$$

Informasi :

n = Jumlah sampel

Z_α = skor Z pada kepercayaan 95% = 1,96

σ = Standard deviasi populasi

e = Tingkat kesalahan penarikan sampel (dalam penelitian ini diambil 5%).

Contoh :

$$n = \left(\frac{(1{,}96) * (0{,}25)}{0{,}05}\right)^2$$

$$n = 96{,}04$$

Peneliti menggunakan skor Z pada tingkat kepercayaan 95% yang menyatakan bahwa sampel mempunyai hasil berukuran 96,04 dengan selisih estimasi dengan kurang dari 0,05. Jadi, sampel yang diambil sebesar 96 karena skor tersebut merupakan skor minimum pengambilan data maka peneliti membulatkan menjadi 100.

2.3.2. *Text Mining*

Text mining merupakan proses menemukan informasi yang tidak diketahui, menggunakan penggalian informasi secara otomatis dari berbagai sumber tertulis. Masalah utama adalah untuk menghubungkan informasi yang diekstraksi bersama untuk membangun fakta atau hipotesis baru untuk eksplorasi lebih lanjut dengan cara eksperimen yang lebih konvensional (Heidarian dan Dinneen, 2016). *Text mining* berbeda dari yang dilakukan oleh mesin pencari, pada dasarnya pencarian membantu pengguna untuk mencari sesuatu yang sudah diketahui dengan kata lain dalam pencarian telah dicoba untuk menyingkirkan semua materi yang tidak relevan untuk menemukan informasi yang relevan. Dalam *text mining*, tujuannya adalah untuk menemukan informasi yang tidak diketahui dari sebelumnya, sesuatu yang belum diketahui oleh siapa pun dan sebagainya. Bahkan menggunakan *text*

mining, informasi dapat diekstraksi untuk memperoleh abstrak kata-kata yang ada dalam dokumen atau untuk menghitung abstrak untuk dokumen berdasarkan kata yang terdapat atau terkandung di dalamnya. *Text mining* sebagai jenis penambangan data yang berbeda yang mencoba untuk menemukan sebuah model pola yang menarik berdasarkan kumpulan data yang sangat besar. Masalah utama yang membedakan penambangan data reguler dari *text mining* adalah bahwa dalam *text mining* pola-pola diekstraksi dari teks bahasa alami dan bukan dari database fakta yang terstruktur.

2.3.3. *Text preprocessing*

Fungsi dari *text preprocessing* yaitu sebagai pengubah data teks yang pertamanya tidak terstruktur menjadi data yang lebih terstruktur lalu disimpan pada basis data (Winarti, dkk, 2017). Tahapan dari *preprocessing* terdiri dari beberapa langkah, yaitu: *case folding*, *tokenization*, *filtering* atau *stopword removal* dan *stemming*. Proses *case folding* merupakan proses untuk merubah semua karakter atau huruf yang terdapat pada sebuah kalimat menjadi huruf kecil atau *lowercase* serta menghilangkan huruf atau karakter yang dianggap tidak valid seperti angka, tanda baca, serta Uniform Resources Locator (URL). Proses dari *tokenization* bertujuan untuk membagi atau memecah setiap kata pada kalimat dari seluruh dokumen pengetahuan ke dalam kata-kata (*term*) berdasarkan pembatasan tab atau sebuah spasi. Selanjutnya selain juga menjadikan kata menjadi huruf kecil yaitu menghilangkan tanda baca seperti tanda koma (,), titik(.), kurung(()), petik("'), tanda seru(!), tanda tanya(?) dan tanda baca lainnya. Berdasrakan hasil dari proses *tokenization* dilanjutkan dengan tahapan proses *filtering*. Proses *filtering* yaitu

merupakan proses untuk mengambil kata penting dari hasil *tokenization*, langkah pada proses ini dapat dilakukan dengan dua teknik, yaitu menghilangkan stopword (membuang kata-kata yang kurang penting) dan daftar kata (menyimpan kata-kata penting). Berdasarkan hasil pemfilteran selanjutnya diproses oleh *stemming*. Proses pada tahapan *stemming* adalah proses mencari akar kata dari setiap kata hasil penyaringan. Data kata teks yang diproses dalam penelitian ini adalah data teks judul dan deskripsi produk di *marketplace* yang berbahasa Indonesia.

2.3.4. Algoritma *Stemming* Nazief & Adriani

Algoritma *stemming* Nazief & Adriani (1996) dikembangkan menurut dari aturan morfologi Indonesia yaitu dengan mengklasifikasikan afiks menjadi prefiks (awalan), sisipan (sufiks), sufiks (sufiks) dan gabungan awalan (confiks) (Rahmatulloh, dkk, 2019). Algoritma ini dibuat oleh Bobby Nazief dan Mirna Adriani, memiliki tahapan sebagai berikut (Adriani, dkk, 2005):

a. Mencari kata-kata yang berasal dari kamus. Jika kata ditemukan, diasumsikan bahwa kata tersebut adalah kata dasar. Kemudian algoritma tersebut akan berhenti.
b. Sufiks Infleksi ("-lah", "-kah", "-ku", "-mu", atau "-nya") dibuang. Jika itu dalam bentuk partikel ("-lah", "-kah", "-tah" atau "-pun") maka langkah diulangi lagi untuk menghapus kata ganti obsesif ("-ku", "-mu", atau "-nya").
c. Hapus Sufiks Derivasi ("-i", "-an" atau "-kan"). Jika nantinya kata tersebut ditemukan dalam kamus, maka algoritma akan berhenti. Jika tidak, lanjutkan ke langkah selanjutnya: Jika "-an" telah dihapus serta huruf terakhir pada kata itu adalah "-k", maka "-k" juga dihapus. Langkah bagian satu : jika nantinya

kata tersebut ditemukan dalam kamus, algoritma akan berhenti. Namun jika tidak ditemukan maka akan dilakukan langkah bagian dua : Sufiks yang dihapus ("-i", "-an" atau "-kan") dikembalikan.

d. Hapus Awalan Derivation (be-, di, me-, pe-, se, te-). Apabila dalam langkah c ada sufiks yang dihapus lalu lanjutkan ke langkah d bagian satu yaitu : Periksa akhiran awalan tabel kombinasi yang tidak diizinkan. Namun jika ditemukan, algoritma akan berhenti, jika tidak masuk ke langkah bagian dua : Untuk i = 1 hingga 3, selanjutnya menentukan jenis awalan kemudian hapus awalan. Jika kata dasar belum ditemukan, maka dilakukan langkah 5, jika algoritma telah berhenti. Catatan: jika awalan kedua sama dengan awalan pertama dari algoritma berhenti.

e. Pengodean ulang.

f. Apabila semua langkah telah selesai dilakukan namun tidak berhasil, maka kata pertama diasumsikan sebagai kata dasar. Sehingga proses selesai.

2.3.5. Similarity

Terdapat tiga metode pendeteksi kesamaan dokumen untuk mengetahui tingkat plagiarisme pada dokumen yang dibandingkan antara lain (Stein dan zu Eissen, 2006):

a. Substring Matching

Metode ini menggunakan pendekatan pencocokan substring dengan mengidentifikasi secara penuh antara string yang dibandingkan kemudian hasil dari substring yang ditemukan akan dihitung sebagai indikator kesamaannya.

b. Keyword Similarity

Mengekstrak kata kunci yang berdasarkan bobot yang telah diberikan pada setiap kata kunci atau sebuah *term* yang selanjutnya bobot tersebut akan dibandingkan dengan dokumen pembanding. Kata kunci yang dibandingkan dilakukan secara rekursif untuk mendeteksi bobot setiap term pada keseluruhan dokumen yang dibandingkan untuk mengetahui tingkat ambang batas kesamaannya.

c. Fingerprint Analysis

Dokumen yang dibandingkan dilakukan pemotongan menggunakan nilai gram tertentu pada setiap dokumen yang kemudian dihitung nilai hash yang nantinya akan diketahui nilai *fingerprint* pada setiap pemotongan tersebut. Setelah ditemukan *fingerprint* selanjutnya dihitung kecocokan *fingerprint* antar dokumen yang dibandingkan.

Berdasarkan jumlah persentase dari jumlah kesamaan dokumen yang didapatkan, terdapat beberapa pembagian proporsi plagiarisme kesamaan dokumen antara lain (Sastroasmoro 2007):

a. Plagiarisme ringan

Proporsi dari plagiarisme ini berdasarkan dari hasil kesamaan dokumen yang mempunyai total persentase kurang dari 30%.

b. Plagiarisme sedang

Total persentase yang diatas dari 30% namun kurang dari 70% dianggap sebagai plagiarisme sedang atau dengan rentang 30-70%.

c. Plagiarisme berat atau total

Plagiarisme pada proporsi ini didapatkan ketika dokumen yang dibandingkan memiliki kesamaan sebesar lebih dari 70%.

2.3.6. Algoritma Winnowing

Algoritma Winnowing merupakan algoritma *fingerprinting* dokumen berfungsi untuk mendeteksi tingkat kesamaan dokumen menggunakan teknik *hashing*. Algoritma Winnowing menggunakan metode dengan proses perbandingan dokumen berupa teknik *fingerprint*ing. Dalam kasus deteksi plagiarisme, metode ini dapat mengidentifikasi bagian-bagian kecil yang serupa dalam sejumlah besar dokumen. *Input* yang diproses menggunakan algoritma ini adalah dokumen teks yang akan diproses untuk menghasilkan *output* dalam bentuk kumpulan nilai *hash*. Nilai *hash* adalah nilai numerik yang dibentuk dari perhitungan ASCII untuk setiap karakter. Pengumpulan nilai *hash* ini kemudian disebut sebagai *fingerprint*ing. *fingerprinting* digunakan sebagai indikator untuk membandingkan kesamaan antara dokumen teks (Riki, dkk, 2019). Langkah-langkah dalam algoritma Winnowing (ALAMSYAH, 2017) yang dijelaskan berdasarkan aturan algoritma winnowing (Schleimer, dkk, 2003):

a. Penghapusan karakter yang tidak relevan

   Teks yang akan dideteksi adalah deteksi plagiarisme. Langkah pertama dalam mengimplementasikan algoritma Winnowing adalah huruf kecil atau mengubah setiap karakter dalam *string* menjadi huruf kecil dan menghapus karakter yang didapatkan dari dokumen yang tidak relevan seperti tanda baca, spasi dan simbol lainnya. tanda baca, spasi, dan simbol selain alfabet dikatakan

tidak relevan karena nilai uniknya tidak dapat diambil dan tidak terkait *string* yang diproses.

Contoh : "E-learning adalah pembelajaran elektronik" akan dirubah menjadi "elearningadalahpembelajaranelektronik"

b. Pembentukan Rangkaian k-gram

Langkah kedua, membentuk teks dari langkah pertama ke dalam urutan k-gram. Pada tahapan ini, teks berdasarkan langkah pertama dalam bentuk kumpulan *string* akan dikelompokkan ke dalam set *string* baru di mana koleksi *string* baru adalah hasil dari penggabungan *string* langkah pertama dengan panjang *string* yang digabungkan adalah *k*.

Contoh : dengan ukuran $k = 5$ maka diperoleh potongsn *string* pada proses sebelumnya menjadi :

elear learn earni arnin rning ninga ingad ngada gadal adala dalah alahp lahpe ahpem hpemb pembe embel mbela belaj elaja lajar ajara jaran arane ranel anele nelek elekt lektr ektro ktron troni ronik

c. Perhitungan nilai *hash*

Langkah ketiga adalah melakukan proses Rolling Hash untuk menghasilkan nilai *hash* dari setiap gram yang telah terbentuk sebelumnya, menggunakan rumus:

$$H_{(c2\_ck+1)} = \left(H_{(c1\_ck+1)} - c1 * b^{(k-1)}\right) * b + c_{k+1} \qquad (2)$$

Informasi :

$H_{(c1\_ck+1)}$ = nilai *hash*

c_1 = nilai ASCII karakter dalam *string*

l = panjang *string*

b = nilai dasar *hash*

d. Pembentukan *window* dari nilai *hash*

Setelah diketahui nilai *hash* pada setiap gram yang diketahui, selanjutnya langkah keempat yaitu membentuk *window*.

e. Pemilihan *fingerprint* dari setiap *window*

Langkah kelima adalah memilih nilai *hash* terkecil dari setiap *window* yang akan digunakan sebagai *fingerprint* dokumen (Sanjaya dan Absar, 2015), jika ada nilai terkecil yang sama maka diambil salah satu nilai.

2.3.7. Jaccard *Similarity*

Hasil perbandingan *hash* yang diperoleh akan dihitung nilai kesamaan menggunakan koefisien Jaccard *Similarity* untuk membandingkan nilai satu dokumen dengan dokumen yang lain berdasarkan *string* yang sama (Riki, dkk, 2019). Berikut rumus yang dipakai untuk persamaan Jaccard *Similarity* (Ji, dkk, 2013) berdasarkan (Broder, 1997) :

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} x\ 100\% \qquad (3)$$

Informasi :

J(A,B) = Jaccard *Similarity*

A, B = Dokumen koresponden

2.3.8. Algoritma Ratcliff/Obershelp

Sequence Matcher merupakan metode yang digunakan untuk mengetahui tingkat kemiripan antar teks yang didasarkan pada algoritma Ratcliff/Obershelp (Saputra, dkk, 2019). Berikut rumus algoritma Ratcliff/Obershelp (Zaidi, dkk, 2019):

$$D_r o = (2 * k_m)/(|S_1| + |S_2| \quad (4)$$

Informasi :

k_m = Banyaknya huruf yang terdeteksi sama dari kedua *string*

$|S_1|, |S_2|$ = Menunjukkan panjang *string*, *Sub-string* terpanjang yang umum di S1 dan S2 disebut "Anchor".

BAB III

METODE PENELITIAN

3.1. Jenis, Sifat, dan Pendekatan Penelitian

3.1.1. Jenis Penelitian

Jenis penelitian yang digunakan oleh peneliti merupakan penelitian kuantitatif, dimana peneliti melakukan perhitungan matematis untuk menemukan hasil yang di inginkan.

3.1.2. Sifat Penelitian

Sifat penelitian yang akan dilaksanakan adalah ekspremental dimana peneliti melakukan sebuah eksperiment guna mendeteksi performa penerapan algoritma Nazief & Adriani dalam perbandingan akurasi dan efisiensi antara algoritma Winnowing dan algoritma Ratcliff/Obershelp dalam mendeteksi tingkat kesamaan produk.

3.1.3. Pendekatan Penelitian

Pendekatan kuantitatif di gunakan oleh peneliti dimana penelitian akan melakukan penelitian sesuai alur yang telah peneliti buat.

3.2. Metode Pengumpulan Data

Data yang yang dipakai pada analisis ini dari *text mining* yang mengambil teks judul dan deskripsi produk pada daftar produk yang dijual di p-store.net berjumlah 100 produk sebagai sampel yang diambil dari macam produk digital, data tersebut diambil oleh beberapa seller pada p-store dengan masukkan 2 *link* produk yang akan dibandingkan yang telah diindikasikan terlebih dahulu apakah produk

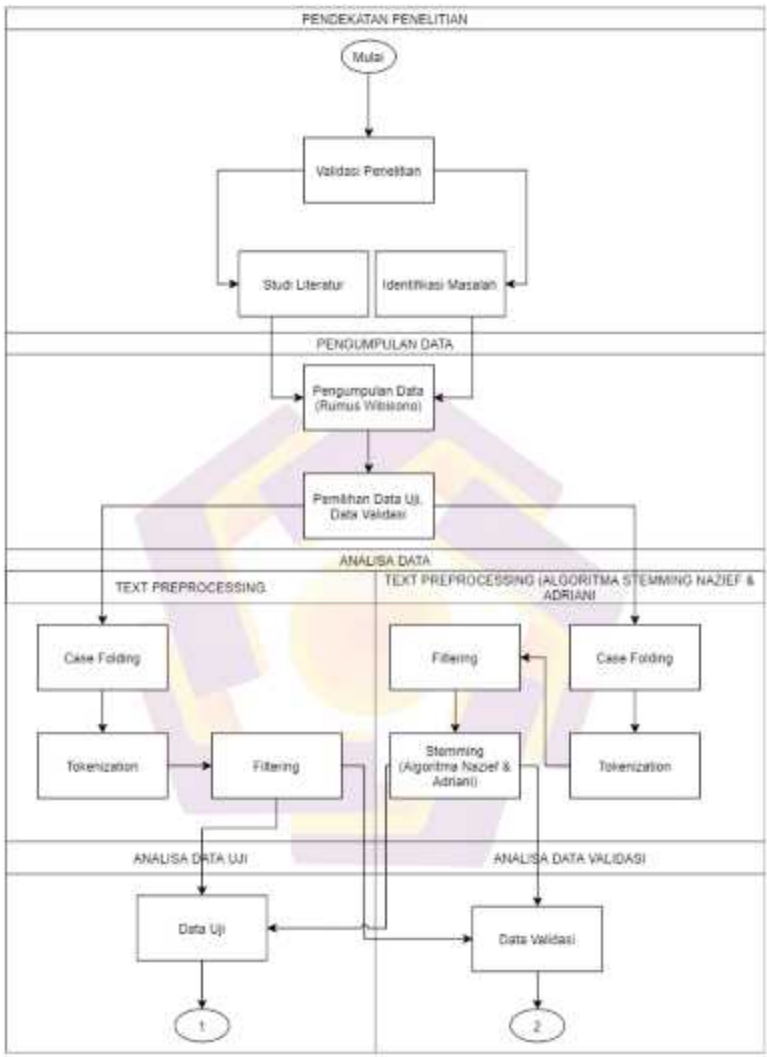
tersebut duplikat atau tidak duplikat dan selanjutnya akan di*scrape* menggunakan kode Curl PHP pada sistem yang dibuat. Jumlah data sampel diambil menurut hasil perhitungan dari besaran sampel yang tidak diketahui jumlah populasinya yang menggunakan rumus Wibisono. Serta menggunakan data sekunder. Data sekunder tersebut merupakan data yang diperoleh dari hasil studi literatur dan dokumen yang terkait.

3.3. Metode Analisis Data

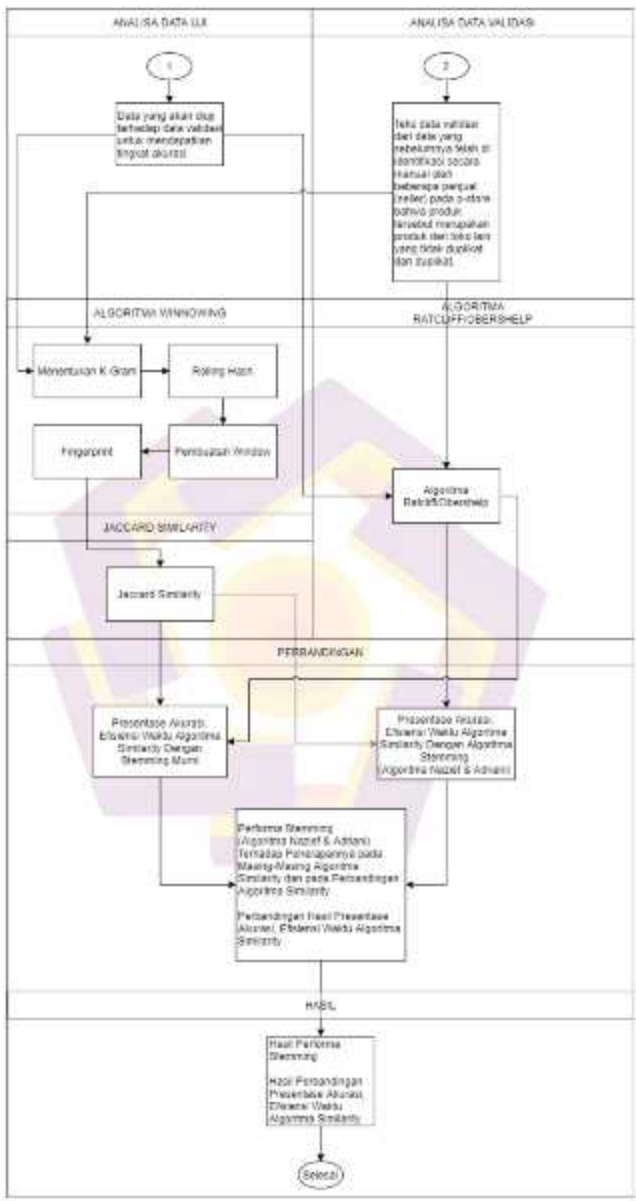
Analisa data dilakukan menggukanan *text preprocessing*, tahap ini melakukan proses *case folding*, *tokenization*, *filtering* atau *stopword removal* dan *stemming*, namun pada proses *stemming* dibedakan menjadi dua yaitu *text preprocessing* murni tanpa *stemming* dan *text preprocessing mengunakan stemming* algoritma Nazief & Adriani. Sehingga setiap data akan mempunyai dua hasil *text preprocessing*. Sedangkan untuk analisa perbandingan antara algoritma Winnowing dan algoritma Ratcliff/Obershelp dibandingkan berdasarkan akurasi hasil perhitungan dari nilai Jaccard *Similarity* untuk proses persentase tingkat *similarity* dari algoritma Winnowing dan dibandingkan dengan nilai tingkat persentase dari algoritma Ratcliff/Obershelp. Analisa performasi dilakukan menggunakan perhitungan kecepatan waktu dan selisih nilai hasil akurasi dalam pemrosesan jika data diolah menggunakan *text preprocessing* murni tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani.

3.4. Alur Penelitian

Alur penelitian yang digunakan pada penelitian ini ditunjukan pada Gambar 3.1 dan Gambar 3.2 sebagai berikut :



Gambar 3.1. Diagram alur penelitian bagian 1.



Gambar 3.2. Diagram alur penelitian bagian 2.

Berikut penjelasan pada alur penelitian pada Gambar 3.1 dan Gambar 3.2, dijelaskan pada beberapa poin menurut dari alur penelitian tersebut dan terdapat beberapa proses alur penelitian yang di jelaskan lebih detail seperti pada proses alur algoritma Winnowing, Jaccard *Similarity* dan algoritma Ratcliff/Obershelp. Penjelasannya sebagai berikut :

3.4.1. Pendekatan Penelitian (Validasi Penelitian, Studi Literatur, Identifikasi Masalah)

Tahap awal penelitian yaitu melakukan validasi penelitian tentang permasalahan serta studi literatur untuk mengetahui masalah yang akan di analisa serta penggunaan algoritma yang tepat berdasarkan studi literatur sehingga dalam pemilihan algoritma tersebut didasarkan pada hasil ilmu yang dianalisa oleh penelitian sebelumnya.

3.4.2. Pengumpulan Data (Pemilihan Data Uji, Data Validasi)

Produk digital yang dijual atau dipublikasi yang terdapat pada situs p-store.net merupakan populasi pada penelitian ini. Mengingat jumlah total populasi tidak diketahui sehingga untuk menentukan jumlah pada ukuran sampel minimum harus cukup mewakili populasi responden yang diteliti. Data sampel pada penelitian ini berjumlah 100 produk yang diambil dari macam produk digital, jumlah data sampel diambil menurut hasil perhitungan berdasarkan besaran sampel yang tidak diketahui total jumlah populasinya yang menggunakan rumus Wibisono. Pengumpulan data dilakukan dengan dua tipe data yaitu data uji dan data validasi, untuk data uji dilakukan *text mining* yaitu dengan *scraping* teks pada produk dari *marketplace* p-store.net sedangkan pada data validasi adalah *text mining* dari

produk yang telah dipilih manual oleh beberapa penjual dengan membandingkan produk secara manual dengan kriteria produk duplikat dan produk tidak duplikat. Berikut ditampilkan data produk yang telah didapatkan menggunakan proses *text mining*.

Tabel 3.1. Data Mentah *Text Mining*

ID	Judul	Deskripsi	Link	Pembanding	Indikasi
1	Jual software pembuat artikel unik Untuk $5	Software ... A: maksimal 1x24 jam	https://p-store.net/pemulisan/10171/software-pembuat-artikel-unik	1	Duplikat
2	Jual software pembuat artikel Untuk $5	Software pembuat artikel unik dalam waktu singkat	https://p-store.net/akun/10182/software-pembuat-artikel	1	Duplikat
3	Jual software pembuat video otomatis untuk bisnis youtube Untuk $5	Meng-otomatis-kan Video Creation Anda! Video Spinn adalah perangkat lunak berbasis desktop, ... ********** </h3> -	https://p-store.net/video/16684/software-pembuat-video-banyak-sekaligus	2	Duplikat

Tabel 3.1. Data Mentah *Text Mining* (Lanjutan)

ID	Judul	Deskripsi	Link	Pembanding	Indikasi
4	Jual software pembuat video banyak sekaligus Untuk $5	Saat ini video pemasaran jauh lebih penting untuk kesuksesan bisnis Anda daripada dulu. Berikut ini alasannya: - Cisco ... ********** </h3> -	https://p-store.net/bisnis/14698/software-pembuat-video-otomatis-untuk-bisnis-youtu	2	Duplikat
...	...	...	...	...	...
99	Jual 10.000 Artikel Bahasa Inggris Untuk $5	Dijual 10.000 artikel bahasa inggris unik seo, sangat cocok untuk blog bule atau untuk daftar adsense. Setiap artikel ada 550 ... kepada saya.	https://p-store.net/penulisan/15346/10000-artikel-bahasa-inggris	50	Tidak Duplikat
100	Jual English Articles (300.000+ 7000) + Untuk Ternak Blog Bahasa Inggris Untuk $1	Artikel bahasa inggris multi niche 300k + 7k lebih siap pakai. Bisa buat yang hobby ternak blog bule, atau buat daftar adsense. ... otomatis walau saya tidak online.	https://p-store.net/penulisan/16526/english-articles-3000007000-untuk-ternak-blog-baha	50	Tidak Duplikat

Berdasarkan Tabel 3.1 didapatkan hasil *text mining* berupa judul, deskripsi, serta tautan dan indikasi awal yang memiliki kriteria duplikat dan tidak duplikat. Sehingga nantinya data mentah dari *text mining* tersebut akan diproses kedalam *text preprocessing*.

3.4.3. Analisa Data (*Text preprocessing* Tanpa *Stemming* dan *Text preprocessing* Menggunakan Algoritma *Stemming* Nazief & Adriani)

Tahap ini melakukan proses *case folding*, *tokenization*, *filtering* atau *stopword removal* dan *stemming*. Proses *case folding* yaitu merubah semua karakter huruf pada sebuah kalimat menjadi huruf kecil dan menghilangkan karakter yang dianggap tidak valid seperti angka, tanda baca, dan Uniform Resources Locator (URL). Proses *tokenization* berguna untuk memecah setiap kalimat dari seluruh dokumen pengetahuan ke dalam kata-kata (*term*) dengan menggunakan pembatas tab dan karakter spasi. Hal yang perlu dilakukan juga adalah menjadikan kata menjadi huruf kecil menghilangkan karakter tanda baca seperti tanda titik(.), koma (,), petik("'), kurung(()), tanda tanya(?), tanda seru(!) dan tanda baca lainnya. Data hasil dari proses *tokenization* dilanjutkan dengan proses *filtering*. Proses *filtering* mengambil kata-kata penting dari hasil proses *tokenization*, langkah proses ini dapat dilakukan dengan dua teknik, yaitu menghilangkan kata sambung dan menyisahkan daftar kata. Data hasil pemfilteran kemudian diproses oleh *stemming*. Namun pada proses *stemming* dibedakan menjadi dua yaitu *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan *stemming* algoritma Nazief & Adriani. Sehingga setiap data akan mempunyai dua hasil *text preprocessing*.

3.4.3.1. *Case folding*

Berikut ditampilkan hasil dari *text preprocessing* pada tahapan *case folding* yang memproses data mentah untuk dihilangkan karakter yang tidak valid dan merubah huruf menjadi kecil yang ditampilkan pada Tabel 3.2.

Tabel 3.2. Hasil *Case folding*

ID	Pembanding	Hasil *Case folding* Judul	Hasil *Case folding* Deskripsi	Hasil *Case folding* Judul + Deskripsi
1	1	jual software pembuat artikel unik untuk	software pembuat artikel unik dalam waktu singkat dengan ... kirimkan a maksimal x jam	jual software pembuat artikel unik untuk software pembuat artikel unik dalam ... kirimkan a maksimal x jam
2	1	jual software pembuat artikel untuk	software pembuat artikel unik dalam waktu singkat dengan ... kirimkan a maksimal x jam	jual software pembuat artikel untuk software pembuat artikel unik dalam ... software akan di kirimkan a maksimal x jam
3	2	jual software pembuat video otomatis untuk bisnis youtube untuk	meng otomatis kan video creation anda video spinn adalah ... untuk berjualan agar laris faq	jual software pembuat video otomatis untuk bisnis youtube untuk meng ... untuk berjualan agar laris faq
4	2	jual software pembuat video banyak sekaligus untuk	saat ini video pemasaran jauh lebih penting untuk ... pengunjung tertarget bahka faq	jual software pembuat video banyak sekaligus untuk saat ini video ... pengunjung tertarget bahka faq
...	...	...	...	...
99	50	jual artikel bahasa inggris untuk	dijual artikel bahasa inggris unik seo sangat cocok untuk blog bule ... membagikan kepada saya	jual artikel bahasa inggris untuk dijual artikel bahasa inggris unik seo sangat ... teman saya sendiri dan dia membagikan kepada saya
100	50	jual english articles untuk ternak blog bahasa inggris untuk	artikel bahasa inggris multi niche *k* k lebih siap pakai bisa buat ... secara otomatis walau saya tidak online	jual english articles untuk ternak blog bahasa inggris untuk artikel bahasa inggris ... otomatis walau saya tidak online

3.4.3.2. *Tokenizing*

Setelah didapatkan hasil dari pemrosesan *case folding* pada Tabel 3.2, selanjutnya hasil pada *case folding* tersebut akan diproses kedalam tahap *tokenizing*

untuk memecah kata berdasarkan spasi. Berikut hasil pemrosesan pada tahap *tokenizing* ditampilkan pada Tabel 3.3.

Tabel 3.3. Hasil *Tokenizing*

ID	Pembandinг	Hasil *Tokenizing* Judul	Hasil *Tokenizing* Deskripsi	Hasil *Tokenizing*
1	1	["jual","software","pembuat","artikel","unik","untuk"]	["software","pembuat","artikel","unik","dalam","waktu","singkat", … "kirimkan","a","maksimal","x","jam"]	["jual","software","pembuat","artikel","unik","untuk","software","pembuat", … ,"kirimkan","a","maksimal","x","jam"]
2	1	["jual","software","pembuat","artikel","untuk"]	["software","pembuat","artikel","unik","dalam","waktu","singkat", … ,"kirimkan","a","maksimal","x","jam"]	["jual","software","pembuat","artikel","untuk","software","pembuat","artikel", ,"kirimkan","a","maksimal","x","jam"]
3	2	["jual","software","pembuat","video","otomatis","untuk","bisnis","youtube","untuk"]	["meng","otomatis","kan","video","creation","anda","video","spinn", … ,"untuk","berjualan","agar","laris","faq"]	["jual","software","pembuat","video","otomatis","untuk","bisnis","youtube", … ,"facebook","untuk","berjualan","agar","laris","faq"]
4	2	["jual","software","pembuat","video","banyak","sekaligus","untuk"]	["saat","ini","video","pemasaran","jauh","lebih","penting","untuk", … ,"pengunjung","tertarget","bahka","faq"]	["jual","software","pembuat","video","banyak","sekaligus","untuk","saat","ini", … "pengunjung","tertarget","bahka","faq"]
…	…	…	…	…
99	50	["jual","artikel","bahasa","inggris","untuk"]	["dijual","artikel","bahasa","inggris","unik","seo","sangat","cocok" … ,"dia","membagikan","kepada","saya"]	["jual","artikel","bahasa","inggris","untuk","dijual","","artikel","bahasa", … ,"dia","membagikan","kepada","saya"]
100	50	["jual","english","articles","","untuk","ternak","blog","bahasa","inggris","untuk"]	["artikel","bahasa","inggris","multi","niche","k","k","lebih","siap", … ,"walau","saya","tidak","online"]	["jual","english","articles","","untuk","ternak","blog","bahasa","inggris","untuk" … ,"secara","otomatis","walau","saya","tidak","online"]

3.4.3.3. *Filtering*

Tahap selanjutnya setelah mendapatkan hasil dari *tokenizing* yaitu mengolah hasil kata yang terpisah pada Tabel 3.3 kepada pemrosesan *filtering* untuk mengambil kata-kata penting. Berikut ditampilkan hasil pada tahapan *filtering* terdapat pada Tabel 3.4.

Tabel 3.4. Hasil *Filtering*

ID	Pembanding	Hasil *Filtering* Judul	Hasil *Filtering* Deskripsi	Hasil *Filtering*
1	1	["jual","software","pembuat","artikel","unik"]	["software","pembuat","artikel","unik","singkat","software","repot", … ,"kirimkan","a","maksimal","x","jam"]	["jual","software","pembuat","artikel","unik","software","pembuat","artikel", … ,"software","kirimkan","a","maksimal","x","jam"]
2	1	["jual","software","pembuat","artikel"]	["software","pembuat","artikel","unik","singkat","software","repot", … ,"kirimkan","a","maksimal","x","jam"]	["jual","software","pembuat","artikel","software","pembuat","artikel","unik", … ,"kirimkan","a","maksimal","x","jam"]
3	2	["jual","software","pembuat","video","otomatis","bisnis","youtube"]	["meng","otomatis","video","creation","video","spinn","perangkat", … ,"facebook","berjualan","laris","faq"]	["jual","software","pembuat","video","otomatis","bisnis","youtube","meng", … ,"domination","facebook","berjualan","laris","faq"]
4	2	["jual","software","pembuat","video"]	["video","pemasaran","kesuksesan","bisnis","alasannya","cisco", … ,"pengunjung","tertarget","bahka","faq"]	["jual","software","pembuat","video","video","pemasaran","kesuksesan", … ,"pengunjung","tertarget","bahka","faq"]
...	...	...	...	...

Tabel 3.4. Hasil *Filtering* (Lanjutan)

ID	Pembanding	Hasil *Filtering* Judul	Hasil *Filtering* Deskripsi	Hasil *Filtering*
99	50	["jual","artikel","bahasa","inggris"]	["dijual","artikel","bahasa","inggris","unik","seo","cocok","blog", ... ,"pembuatan","teman","membagikan"]	["jual","artikel","bahasa","inggris","dijual","artikel","bahasa","inggris","unik", ... ,"produk","pembuatan","teman","membagikan"]
100	50	["jual","english","articles","","ternak","blog","bahasa","inggris"]	["artikel","bahasa","inggris","multi","niche","k","k","pakai","hobby", ... ,"langsung","file","otomatis","online"]	["jual","english","articles","","ternak","blog","bahasa","inggris","artikel", ... ,"langsung","file","otomatis","online"]

3.4.3.4. *Stemming*

Berdasarkan Tabel 3.4 telah didapatkan kata-kata penting yang telah dihilangkan kata sambungnya, selanjutnya hasil dari *filtering* tersebut akan diproses kedalam tahapan *stemming*, untuk proses *stemming* pertama yaitu dilakukan proses *text preprocessing* tanpa *stemming* atau tanpa aturan. Hasil *text preprocessing* tanpa *stemming* ditampilkan pada Tabel 3.5.

Tabel 3.5. Hasil *Text preprocessing* Tanpa *Stemming*

ID	Pembanding	Hasil *Text preprocessing* Judul	Hasil *Text preprocessing* Deskripsi	Hasil *Text preprocessing* Judul + Deskripsi
1	1	["jual","software","pembuat","artikel","unik"]	["software","pembuat","artikel","unik","singkat","software","repot", ... ,"kirimkan","a","maksimal","x","jam"]	["jual","software","pembuat","artikel","unik","software","pembuat","artikel", ... ,"software","kirimkan","a","maksimal","x","jam"]
2	1	["jual","software","pembuat","artikel"]	["software","pembuat","artikel","unik","singkat","software","repot", ... ,"kirimkan","a","maksimal","x","jam"]	["jual","software","pembuat","artikel","software","pembuat","artikel","unik", ... ,"kirimkan","a","maksimal","x","jam"]

Tabel 3.5. Hasil *Text preprocessing* Tanpa *Stemming* (Lanjutan)

ID	Pembanding	Hasil *Text preprocessing* Judul	Hasil *Text preprocessing* Deskripsi	Hasil *Text preprocessing* Judul + Deskripsi
3	2	["jual","software","pembuat","video","otomatis","bisnis","youtube"]	["meng","otomatis","video","creation","video","spinn","perangkat", … ,"facebook","berjualan","laris","faq"]	["jual","software","pembuat","video","otomatis","bisnis","youtube","meng", … ,"domination","facebook","berjualan","laris","faq"]
4	2	["jual","software","pembuat","video"]	["video","pemasaran","kesuksesan","bisnis","alasannya","cisco", … ,"pengunjung","tertarget","bahka","faq"]	["jual","software","pembuat","video","video","pemasaran","kesuksesan", … ,"pengunjung","tertarget","bahka","faq"]
…	…	…	…	…
99	50	["jual","artikel","bahasa","inggris"]	["dijual","artikel","bahasa","inggris","unik","seo","cocok","blog", … ,"pembuatan","teman","membagikan"]	["jual","artikel","bahasa","inggris","dijual","artikel","bahasa","inggris","unik", … ,"produk","pembuatan","teman","membagikan"]
100	50	["jual","english","articles","","ternak","blog","bahasa","inggris"]	["artikel","bahasa","inggris","multi","niche","k","k","pakai","hobby", … ,"langsung","file","otomatis","online"]	["jual","english","articles","","ternak","blog","bahasa","inggris","artikel", … ,"langsung","file","otomatis","online"]

Selain hasil dari *text preprocessing* tanpa *stemming* yang telah ditampilkan pada Tabel 3.5 selanjutnya juga dilakukan pemrosesan *stemming* dengan menggunakan algoritma Nazief & Adriani. Berikut hasil pemrosesan *stemming* yang dilakukan menggunakan algoritma Nazief & Adriani ditampilkan pada Tabel 3.6.

Tabel 3.6. Hasil *Text preprocessing* Menggunakan *Stemming* Algoritma Nazief & Adriani

ID	Pembanding	Hasil *Text preprocessing* + Algoritma Nazief & Adriani Judul	Hasil *Text preprocessing* + Algoritma Nazief & Adriani Deskripsi	Hasil *Text preprocessing* + Algoritma Nazief & Adriani Judul + Deskripsi
1	1	["jual","software","buat","artikel","unik"]	["software","buat","artikel","unik","singkat","software","repot", ... ,"kirim","a","maksimal","x","jam"]	["jual","software","buat","artikel","unik","software","buat","artikel","unik", ... ,"q","software","kirim","a","maksimal","x","jam"]
2	1	["jual","software","buat","artikel"]	["software","buat","artikel","unik","singkat","software","repot", ... ,"software","kirim","a","maksimal","x","jam"]	["jual","software","buat","artikel","software","buat","artikel","unik","singkat", ... ,"software","kirim","a","maksimal","x","jam"]
3	2	["jual","software","buat","video","otomatis","bisnis","youtube"]	["meng","otomatis","video","creation","video","spinn","angkat", ... ,"facebook","berjualan","laris","faq"]	["jual","software","buat","video","otomatis","bisnis","youtube","meng", ... ,"domination","facebook","berjualan","laris","faq"]
4	2	["jual","software","buat","video"]	["video","pasar","sukses","bisnis","alas","cisco","klaim","trafik", ... ,"mudah","unjung","target","bahka","faq"]	["jual","software","buat","video","video","pasar","sukses","bisnis","alas", ... ,"mudah","unjung","target","bahka","faq"]
...	...	...	...	...
99	50	["jual","artikel","bahasa","inggris"]	["jual","artikel","bahasa","inggris","unik","o","cocok","blog", ... ,"faq","produk","buat","teman","bagi"]	["jual","artikel","bahasa","inggris","jual","","artikel","blog","download","faq", ... ,"produk","buat","teman","bagi"]
100	50	["jual","english","articles","","ternak","blog","bahasa","inggris"]	["artikel","bahasa","inggris","multi","niche","k","k","pakai","hobby", ... ,"bayar","langsung","file","otomatis","online"]	["jual","english","articles","","ternak","blog","bahasa","inggris","artikel", ... ,"bayar","langsung","file","otomatis","online"]

Berdasarkan Tabel 3.6 telah ditampilkan hasil dari pemrosesan *stemming* menggunakan algoritma Nazief & Adriani. Sehingga proses pada *text*

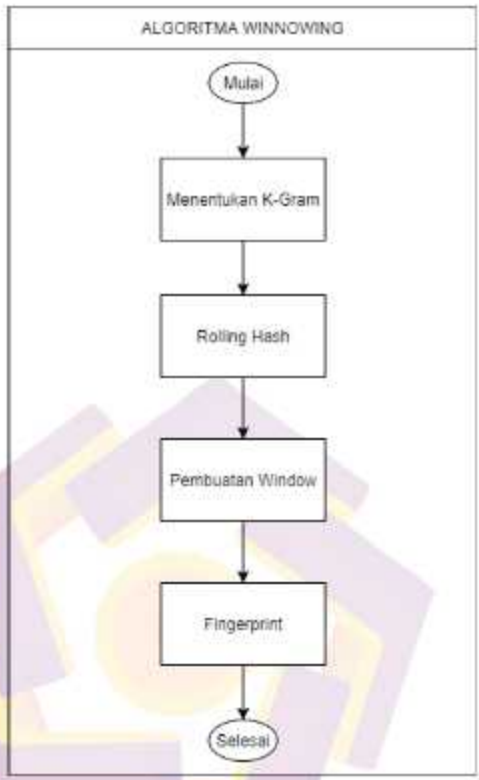
preprocessing telah selesai selanjutnya hasil *stemming* akan dilanjutkan pada pemrosesan pada tahapan algoritma *similarity*.

3.4.4. Analisa Data Uji dan Data Validasi

Data uji dan data validasi selanjutnya masing-masing akan memiliki dua varian dari tahap *text preprocessing*, data validasi ini data yang sebelumnya dipilih oleh beberapa *seller* untuk diindetifikasi manual dengan kriteria produk sejenis tapi duplikasi, produk sejenis tidak duplikasi. Data uji dan data Validasi akan diproses ke dalam algoritma *similarity* untuk mengetahui tingkat *similarity* dan akan dinilai akurasinya berdasarkan kriteria yang di tentukan.

3.4.5. Algoritma Winnowing

Berikut detail dari alur penelitian pada tahap algoritma Winnowing :



Gambar 3.3. Diagram alur penelitian tahap Algoritma Winnowing.

Gambar 3.3 menjelaskan bahwa tahap pada langkah ini menentukan variasi k-gram terlebih dahulu, selanjutnya data uji dan data validasi akan di hitung nilai *hashing*nya menggunakan Rolling Hash yang nantinya akan dibuat beberapa *window* dan setiap *window* akan dicari nilai terkecil atau *fingerprint*.

3.4.5.1. Menentukan K-Gram

Berikut ditampilkan hasil dari pemrosesan data *stemming* kedalam tahapan pemotongan dengan jumlah *k* dengan bilangan prima yaitu 2, 3, 5, dan 7

sehingga hasil dari perpotongan kata berdasarkan k-gram dengan *input*an data *text preprocessing* tanpa *stemming* ditampilkan pada Tabel 3.7.

Tabel 3.7. Hasil Perpotongan K-Gram pada *Text preprocessing* Tanpa *Stemming*

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Judul	["ju","ua","al","ls","so","of","ft","tw","wa", ... ,"ke","el","lu","un","ni","ik"]	["jua","ual","als","lso","soft","oft","ftw","twa","war","are", ... ,"ike","kel","elu","lun","uni","nik"]	["juals","ualso","alsof","lsoft","softw","oftwa","ftwar", ... ,"rtike","tikel","ikelu","kelun","eluni","lunik"]	["jualsof","ualsoft","alsoftw","lsoftwa","softwar","oftware","ftwarep", ... ,"rtikelu","tikelun","ikeluni","kelunik"]
1	Deskripsi	["so","of","ft","tw","wa","ar","re", ... ,"al","lx","xj","ja","am"]	["sof","oft","ftw","twa","war","are","rep","epe","pem","emb", ... ,"sim","ima","mal","alx","lxj","xja","jam"]	["softw","oftwa","ftwar","tware","warep","arepe","repem", ... ,"simal","imalx","malxj","alxja","lxjam"]	["softwar","oftware","ftwarep","twarepe","warepem","arepemb", ... ,"ksimalx","simalxj","imalxja","malxjam"]
1	Judul + Deskripsi	["ju","ua","al","ls","so","of","ft","tw", ... ,"al","lx","xj","ja","am"]	["jua","ual","als","lso","soft","oft","ftw","twa","war","are", ... ,"ima","mal","alx","lxj","xja","jam"]	["juals","ualso","alsof","lsoft","softw","oftwa","ftwar", ... ,"simal","imalx","malxj","alxja","lxjam"]	["jualsof","ualsoft","alsoftw","lsoftwa","softwar","oftware","ftwarep", ... ,"ksimalx","simalxj","imalxja","malxjam"]
2	Judul	["ju","ua","al","ls","so","of","ft","tw","wa", ... ,"rt","ti","ik","ke","el"]	["jua","ual","als","lso","soft","oft","ftw","twa","war","are", ... ["ata","tar","art","rti","tik","ike","kel"]	["juals","ualso","alsof","lsoft","softw","oftwa","ftwar", ... ,"atart","tarti","artik","rtike","tikel"]	["jualsof","ualsoft","alsoftw","lsoftwa","softwar","oftware","ftwarep", ... ,"buatart","uatarti","atartik","tartike","artikel"]
2	Deskripsi	["so","of","ft","tw","wa","ar","re","ep", ... ,"al","lx","xj","ja","am"]	["sof","oft","ftw","twa","war","are","rep","epe","pem","emb", ... ,"sim","ima","mal","alx","lxj","xja","jam"]	["softw","oftwa","ftwar","tware","warep","arepe","repem", ... ,"simal","imalx","malxj","alxja","lxjam"]	["softwar","oftware","ftwarep","twarepe","warepem","arepemb", ... ,"ksimalx","simalxj","imalxja","malxjam"]

Tabel 3.7. Hasil Perpotongan K-Gram pada *Text preprocessing* Tanpa *Stemming* (Lanjutan)

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
2	Judul + Deskripsi	["ju","ua","al","ls","so","of","ft","tw","wa", ... ,"ma","al","lx","xj","ja","am"]	["jua","ual","als","lso","soft","ofi","ftw","twa","war","are", ... ,"mal","alx","lxj","xja","jam"]	["juals","ualso","alsof","lsoft","softw","oftwa","ftwar", ... ,"simal","imalx","malxj","alxja","lxjam"]	["jualsof","ualsoft","alsoftw","lsoftwa","softwar","oftware","ftwarep", ... ,"simalxj","imalxja","malxjam"]
...	...	...	...	...	...
100	Judul + Deskripsi	["ju","ua","al","le","en","ng","gl","li","is", ... ,"nl","li","in","ne"]	["jua","ual","ale","len","eng","ngl","gli","lis","ish","sha","har", ... ,"tis","iso","son","onl","nli","lin","ine"]	["juale","ualen","aleng","lengl","engli","nglis","glish", ... ,"tison","isonl","sonli","onlin","nline"]	["jualeng","ualengl","alengli","lenglis","english","nglisha","glishar", ... ,"matison","atisonl","tisonli","isonlin","sonline"]

Selain dari pemrosesan pemotongan karakter berdasarkan jumlah nilai *k* pada K-Gram yang diterapkan pada *text preprocessing* tanpa *stemming* selanjutnya dilakukan pemotongan karakter pada hasil *stemming* yang menggunakan algoritma Nazief & Adriani. Berikut ditampilkan hasil dari pemotongan karakter pada hasil *stemming* Nazief & Adriani ditampilkan pada Tabel 3.8.

Tabel 3.8. Hasil Perpotongan K-Gram pada *Text preprocessing* Menggunakan *Stemming* Algoritma Nazief & Adriani

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Judul	["ju","ua","al","ls","so","of","ft","tw","wa", ... ,"ke","el","lu","un","ni","ik"]	["jua","ual","als","lso","soft","oft","ftw","twa","war","are", ... ,"ike","kel","elu","lun","uni","nik"]	["juals","ualso","alsof","lsoft","softw","oftwa","ftwar", ... ,"ikelu","kelun","eluni","lunik"]	["jualsof","ualsoft","alsoftw","lsoftwa","softwar","oftware","ftwareb", ... ,"rtikelu","tikelun","ikeluni","kelunik"]
1	Deskripsi	["so","of","ft","tw","wa","ar","re","eb", ... ,"al","lx","xj","ja","am"]	["sof","oft","ftw","twa","war","are","reb","ebu","bua","uat", ... ,"ima","mal","alx","lxj","xja","jam"]	["softw","oftwa","ftwar","tware","wareb","arebu","rebua", ... ,"simal","imalx","malxj","alxja","lxjam"]	["softwar","oftware","ftwareb","twarebu","warebua","arebuat", ... ,ksimalx","simalxj","imalxja","malxjam"]

Tabel 3.8. Hasil Perpotongan K-Gram pada *Text preprocessing* Menggunakan *Stemming* Algoritma Nazief & Adriani (Lanjutan)

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Judul + Deskripsi	["ju","ua","al", "ls","so","of","ft","tw","wa", ... ,"al","lx","xj", "ja","am"]	["jua","ual","als","lso","sof","ofi","ftw","twa","war","are", ... ,"sim","ima","mal","alx","lxj","xja","jam"]	["juals","ualso","alsof","lsoft","softw","oftwa","ftwar", ... ,"imalx","malxj","alxja","lxjam"]	["jualsof","ualsoft","alsoftw","lsoftwa","softwar","oftware","ftwareb", ... ,"ksimalx","simalxj","imalxja","malxjam"]
2	Judul	["ju","ua","al", "ls","so","of","ft","tw","wa", ... ,"ar","rt","ti","ik","kc","cl"]	["jua","ual","als","lso","sof","ofi","ftw","twa","war","are", ... ,"ata","tar","art","rti","tik","ikc","kcl"]	["juals","ualso","alsof","lsoft","softw","oftwa","ftwar", ... ,"atart","tarti","artik","rtikc","tikcl"]	["jualsof","ualsoft","alsoftw","lsoftwa","softwar","oftware","ftwareb", ... ,"buatart","uatarti","atartik","tartikc","artikcl"]
2	Deskripsi	["so","of","ft", "tw","wa","ar","rc","cb", ... ,"al","lx","xj", "ja","am"]	["sof","oft","ftw","twa","war","arc","rcb","cbu","bua","uat", ... ,"ima","mal","alx","lxj","xja","jam"]	["softw","oftwa","ftwar","tware","warcb","arcbu","rcbua", ... ,"imalx","malxj","alxja","lxjam"]	["softwar","oftware","ftwarcb","twarcbu","warcbua","arcbuat", ... ,"simalxj","imalxja","malxjam"]
2	Judul + Deskripsi	["ju","ua","al", "ls","so","of","ft","tw","wa", ... ,"al","lx","xj", "ja","am"]	["jua","ual","als","lso","sof","ofi","ftw","twa","war","are", ... ,"sim","ima","mal","alx","lxj","xja","jam"]	["juals","ualso","alsof","lsoft","softw","oftwa","ftwar", ... ,"simal","imalx","malxj","alxja","lxjam"]	["jualsof","ualsoft","alsoftw","lsoftwa","softwar","oftware","ftwareb", ... ,"ksimalx","simalxj","imalxja","malxjam"]
...	...	...	...	...	...
100	Judul + Deskripsi	["ju","ua","al", "le","en","ng", "gl","li","is", ... ,"on","nl","li", "in","ne"]	["jua","ual","ale","len","eng","ngl","gli","lis","ish","sha", ... ,"tis","iso","son","onl","nli","lin","ine"]	["juale","ualen","aleng","lengl","engli","nglis","glish", ... ,"tison","isonl","sonli","onlin","nline"]	["jualeng","ualengl","alengli","lenglis","englisha","nglisha","glishar", ... ,"atisonl","tisonli","isonlin","sonline"]

3.4.5.2. Rolling Hash

Setelah mendapatkan hasil dari potongan karakter berdasarkan K-Gram selanjutnya menghitung nilai *hash* menggunakan rumus Rolling Hash, berikut perhitungan Rolling Hash pada data ID 1 dengan tipe *input*an Judul pada *text preprocessing* tanpa *stemming* dengan hasil pada K-Gram 2 pada potongan karakter "ju" dengan nilai basis bilangan prima = 11, sebagai berikut :

$$H_{(ju)} = ascii(j) * 11^{(1)} + ascii(u) * 11^{(0)}$$

$$= 106 * 11 + 117 * 1$$

$$= 1283$$

Sehingga dapat dihasilkan perubahan dari potongan karakter menjadi *hash* berdasarkan perhitungan Rolling Hash, berikut ditampilkan hasil dari perhitungan Rolling Hash pada *text preprocessing* tanpa *stemming* berdasarkan potongan K-Gram ditampilkan pada Tabel 3.9.

Tabel 3.9. Hasil Rolling Hash Berdasarkan K-Gram pada *Text preprocessing* Tanpa *Stemming*

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Judul	[1283,1384,1175,1303,1376,1323,1238, ... ,1219,1305,1397,1315,1262]	[14210,15332,13040,14444,15238,14669,13737,15442,15580, ... ,13983,14166,13526,14465,15472,14572]	[1720713,1856548,1579163,1748962,1845193,1776355, ... ,1693248,1715483,1637961,1751527]	[208207596,224643546,191080118,211625808,223269534, ... ,224126914,204884323,207574705]
1	Deskripsi	[1376,1323,1238,1395,1406,1181,1355, ... ,1175,1308,1426,1263,1176]	[15238,14669,13737,15442,15580,13092,15017,13554,14772, ... ,14001,14364,13045,14494,15783,14002]	[1845193,1776355,1663358,1869837,1886403,1585465, ... ,1695429,1739470,1579708,1754950]	[223269534,214940310,201267541,226251610,228255983, ... ,222379340,205148172,210477046]
1	Judul + Deskripsi	[1283,1384,1175,1303,1376,1323,1238, ... ,1175,1308,1426,1263,1176]	[14210,15332,13040,14444,15238,14669,13737,15442,15580, ... ,14001,14364,13045,14494,15783,14002]	[1720713,1856548,1579163,1748962,1845193,1776355, ... ,1695429,1739470,1579708,1754950]	[208207596,224643546,191080118,211625808,223269534, ... ,222379340,205148172,210477046]
2	Judul	[1283,1384,1175,1303,1376,1323,1238, ... ,1370,1381,1262,1278,1219]	[14210,15332,13040,14444,15238,14669,13737,15442,15580, ... ,15217,13107,15175,15298,13983,14166]	[1720713,1856548,1579163,1748962,1845193,1776355, ... ,1842638,1587209,1837453,1852277]	[208207596,224643546,191080118,211625808,223269534, ... ,192110542,222960476,192053508]

Tabel 3.9. Hasil Rolling Hash Berdasarkan K-Gram pada *Text preprocessing* Tanpa *Stemming* (Lanjutan)

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
2	Deskripsi	[1376,1323,1238,1395,1406,1181,1355, … ,1175,1308,1426,1263,1176]	[15238,14669,13737,15442,15580,13092,15017,13554,14772, … ,14001,14364,13045,14494,15783,14002]	[1845193,1776355,1663358,1869837,1886403,1585465, … ,1695429,1739470,1579708,1754950]	[223269534,214940310,201267541,226251610,228255983, … ,222379340,205148172,210477046]
2	Judul + Deskripsi	[1283,1384,1175,1303,1376,1323,1238, … ,1175,1308,1426,1263,1176]	[14210,15332,13040,14444,15238,14669,13737,15442, … ,14364,13045,14494,15783,14002]	[1720713,1856548,1579163,1748962,1845193,1776355, … ,1695429,1739470,1579708,1754950]	[208207596,224643546,191080118,211625808,223269534, … ,209773321,222379340,205148172,210477046]
…	…	…	…	…	…
100	Judul + Deskripsi	[1283,1384,1175,1289,1221,1313,1241, … ,1331,1318,1293,1265,1311]	[14210,15332,13026,14289,13534,14551,13756,14338,14074, … ,14081,15246,14749,14603,14333,14016]	[1720699,1856393,1577459,1730210,1638907,1761941, … ,1705119,1846059,1785894,1768274]	[208205892,224624794,190873832,209356680,198309116, … ,192228452,224257490,206320064,223374450]

Selain itu juga dilakukan perhitungan pada *stemming* yang menggunakan algoritma Nazief & Adriani, berikut hasil dari perhitunga *hash* menggunakan Rolling Hash pada *stemming* menggunakan algoritma Nazief & Adriani berdasarkan K-Gram yang terbentuk ditampilkan pada Tabel 3.10.

Tabel 3.10. Hasil Rolling Hash Berdasarkan K-Gram pada *Text preprocessing* Menggunakan *Stemming* Algoritma Nazief & Adriani

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Judul	[1283,1384,1175,1303,1376,1323,1238, … ,1305,1397,1315,1262]	[14210,15332,13040,14444,15238,14669,13737,15442,15580, … ,13983,14166,13526,14465,15472,14572]	[1720713,1856548,1579163,1748962,1845193,1776355, … ,1693248,1715483,1637961,1751527]	[208207596,224643546,191080118,211625808,223269534, … ,222333118,224126914,204884323,207574705]

Tabel 3.10. Hasil Rolling Hash Berdasarkan K-Gram pada *Text preprocessing* Menggunakan *Stemming* Algoritma Nazief & Adriani (Lanjutan)

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Deskripsi	[1376,1323,1238,1395,1406,1181,1355, ... ,1296,1175,1308,1426,1263,1176]	[15238,14669,13737,15442,15580,13092,15003,13416,13242, ... ,14001,14364,13045,14494,15783,14002]	[1845193,1776355,1663358,1869837,1886389,1585327 , ... ,1695429,1739470,1579708,1754950]	[223269534,214940310,201267527,226251472,228254453, ... ,209773321,222379340,205148172,210477046]
1	Judul + Deskripsi	[1283,1384,1175,1303,1376,1323,1238, ... ,,1308,1426,1263,1176]	[14210,15332,13040,14444,15238,14669,13737,15442,15580, ... ,14001,14364,13045,14494,15783,14002]	[1720713,1856548,1579163,1748962,1845193,1776355, ... ,1695429,1739470,1579708,1754950]	[208207596,224643546,191080118,211625808,223269534,214940310, ... ,209773321,222379340,205148172,210477046]
2	Judul	[1283,1384,1175,1303,1376,1323,1238, ... ,1370,1381,1262,1278,1219]	[14210,15332,13040,14444,15238,14669,13737,15442,15580, ... ,15217,13107,15175,15298,13983,14166]	[1720713,1856548,1579163,1748962,1845193,1776355, ... ,1842638,1587209,1837453,1852277]	[208207596,224643546,191080118,211625808,223269534,214940310, ... ,224737222,192110542,222960476,192053508]
2	Deskripsi	[1376,1323,1238,1395,1406,1181,1355, ... ,1175,1308,1426,1263,1176]	[15238,14669,13737,15442,15580,13092,15003,13416,13242, ... ,14364,13045,14494,15783,14002]	[1845193,1776355,1663358,1869837,1886389,1585327 , ... ,1695429,1739470,1579708,1754950]	[223269534,214940310,201267527,226251472,228254453,191825750, ... ,222379340,205148172,210477046]
2	Judul + Deskripsi	[1283,1384,1175,1303,1376,1323,1238 , ... ,1175,1308,1426,1263,1176]	[14210,15332,13040,14444,15238,14669,13737,15442,15580, ... ,14001,14364,13045,14494,15783,14002]	[1720713,1856548,1579163,1748962,1845193,1776355 , ... ,1695429,1739470,1579708,1754950]	[208207596,224643546,191080118,211625808,223269534, ... ,209773321,222379340,205148172,210477046]
...	...				
100	Judul + Deskripsi	[1283,1384,1175,1289,1221,1313,1241, ... ,1331,1318,1293,1265,1311]	[14210,15332,13026,14289,13534,14551,13756,14338,14074, ... ,14081,15246,14749,14603,14333,14016]	[1720699,1856393,1577459,1730210,1638907,1761941, ... ,1705119,1846059,1785894,1768274]	[208205892,224624794,190873832,209356680,198309116,213196102, ... ,192228452,224257490,206320664,223374450]

3.4.5.3. Pembuatan *Window*

Setelah diketahui *hash* setiap sebaran K-Gram langkah selanjutnya yaitu pembuatan *window* dengan jumlah w = 2. Sehingga *hash* yang ada akan dikelompokkan kedalam *window* yang terbentuk. Berikut hasil dari *window* pada *text preprocessing* tanpa *stemming* berdasarkan K-Gram yang terbentuk ditampilkan pada Tabel 3.11.

Tabel 3.11. Hasil Sebaran *Hash* pada *Window* Berdasarkan K-Gram pada *Text preprocessing* Tanpa *Stemming*

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Judul	[[1283,1384],[1384,1175],[1175,1303], ... ,[1397,1315],[1315,1262]]	[[14210,15332],[15332,13040],[13040,14444],[14444,15238], ... ,[14465,15472],[15472,14572]]	[[1720713,1856548],[1856548,1579163],[1579163,1748962], ... ,[1715483,1637961],[1637961,1751527]]	[[208207596,224643546],[224643546,191080118], ... ,[224126914,204884323],[204884323,207574705]]
1	Deskripsi	[[1376,1323],[1323,1238],[1238,1395], ... ,[1426,1263],[1263,1176]]	[[15238,14669],[14669,13737],[13737,15442],[15442,15580], ... ,[14494,15783],[15783,14002]]	[[1845193,1776355],[1776355,1663358],[1663358,1869837], ... ,[1739470,1579708],[1579708,1754950]]	[[223269534,214940310],[214940310,201267541], ... [222379340,205148172],[205148172,210477046]]
1	Judul + Deskripsi	[[1283,1384],[1384,1175],[1175,1303], ... ,[1426,1263],[1263,1176]]	[[14210,15332],[15332,13040],[13040,14444],[14444,15238], ... ,[14494,15783],[15783,14002]]	[[1720713,1856548],[1856548,1579163],[1579163,1748962], ... ,[1739470,1579708],[1579708,1754950]]	[[208207596,224643546],[224643546,191080118], ... ,[222379340,205148172],[205148172,210477046]]
2	Judul	[[1283,1384],[1384,1175],[1175,1303], ... ,[1262,1278],[1278,1219]]	[[14210,15332],[15332,13040],[13040,14444],[14444,15238], ... ,[15298,13983],[13983,14166]]	[[1720713,1856548],[1856548,1579163],[1579163,1748962], ... ,[1587209,1837453],[1837453,1852277]]	[[208207596,224643546],[224643546,191080118], ... ,[192110542,222960476],[222960476,192053508]]

Tabel 3.11. Hasil Sebaran *Hash* pada *Window* Berdasarkan K-Gram pada *Text preprocessing* Tanpa *Stemming* (Lanjutan)

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
2	Deskripsi	[[1376,1323],[1323,1238],[1238,1395], ... ,[1426,1263],[1263,1176]]	[[15238,14669],[14669,13737],[13737,15442],[15442,15580], ... ,[14494,15783],[15783,14002]]	[[1845193,1776355],[1776355,1663358],[1663358,1869837], ... ,[1739470,1579708],[1579708,1754950]]	[[223269534,214940310],[214940310,201267541], ... ,[222379340,205148172],[205148172,210477046]]
2	Judul + Deskripsi	[[1283,1384],[1384,1175],[1175,1303], ... ,[1426,1263],[1263,1176]]	[[14210,15332],[15332,13040],[13040,14444],[14444,15238], ... ,[14494,15783],[15783,14002]]	[[1720713,1856548],[1856548,1579163],[1579163,1748962], ... ,[1739470,1579708],[1579708,1754950]]	[[208207596,224643546],[224643546,191080118],[191080118,211625808], ... ,[222379340,205148172],[205148172,210477046]]
...	...	...	...	...	...
100	Judul + Deskripsi	[[1283,1384],[1384,1175],[1175,1289], ... ,[1293,1265],[1265,1311]]	[[14210,15332],[15332,13026],[13026,14289],[14289,13534], ... ,[14603,14333],[14333,14016]]	[[1720699,1856393],[1856393,1577459],[1577459,1730210], ... ,[1846059,1785894],[1785894,1768274]]	[[208205892,224624794],[224624794,190873832],[190873832,209356680], ... ,[224257490,206320664],[206320664,223374450]]

Berikut ditampilkan hasil dari pembentukan *window* terhadap *hash* yang terbentuk pada proses sebelumnya pada *hash* yang terbentuk pada *stemming* algoritma Nazief & Adriani ditampilkan pada Tabel 3.12.

Tabel 3.12. Hasil Sebaran *Hash* pada *Window* Berdasarkan K-Gram pada *Text preprocessing* Menggunakan *Stemming* Algoritma Nazief & Adriani

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Judul	[[1283,1384],[1384,1175],[1175,1303], ... ,[1397,1315],[1315,1262]]	[[14210,15332],[15332,13040],[13040,14444],[14444,15238], ... ,[14465,15472],[15472,14572]]	[[1720713,1856548],[1856548,1579163],[1579163,1748962], ... ,[1715483,1637961],[1637961,1751527]]	[[208207596,224643546],[224643546,191080118],[191080118,211625808], ... ,[224126914,204884323],[204884323,207574705]]

Tabel 3.12. Hasil Sebaran *Hash* pada *Window* Berdasarkan K-Gram pada *Text preprocessing* Menggunakan *Stemming* Algoritma Nazief & Adriani (Lanjutan)

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Deskripsi	[[1376,1323],[1323,1238],[1238,1395], ...,[1426,1263],[1263,1176]]	[[15238,14669],[14669,13737],[13737,15442],[15442,15580], ...,[14494,15783],[15783,14002]]	[[1845193,1776355],[1776355,1663358],[1663358,1869837], ...,[1739470,1579708],[1579708,1754950]]	[[223269534,214940310],[214940310,201267527],[201267527,226251472], ...,[222379340,205148172],[205148172,210477046]]
1	Judul + Deskripsi	[[1283,1384],[1384,1175],[1175,1303], ...,[1426,1263],[1263,1176]]	[[14210,15332],[15332,13040],[13040,14444],[14444,15238], ...,[14494,15783],[15783,14002]]	[[1720713,1856548],[1856548,1579163],[1579163,1748962], ...,[1739470,1579708],[1579708,1754950]]	[[208207596,224643546],[224643546,191080118],[191080118,211625808], ...,[222379340,205148172],[205148172,210477046]]
2	Judul	[[1283,1384],[1384,1175],[1175,1303], ...,[1262,1278],[1278,1219]]	[[14210,15332],[15332,13040],[13040,14444],[14444,15238], ...,[15298,13983],[13983,14166]]	[[1720713,1856548],[1856548,1579163],[1579163,1748962], ...,[1587209,1837453],[1837453,1852277]]	[[208207596,224643546],[224643546,191080118],[191080118,211625808], ...,[192110542,222960476],[222960476,192053508]]
2	Deskripsi	[[1376,1323],[1323,1238],[1238,1395], ...,[1426,1263],[1263,1176]]	[[15238,14669],[14669,13737],[13737,15442],[15442,15580], ...,[14494,15783],[15783,14002]]	[[1845193,1776355],[1776355,1663358],[1663358,1869837], ...,[1739470,1579708],[1579708,1754950]]	[[223269534,214940310],[214940310,201267527],[201267527,226251472], ...,[222379340,205148172],[205148172,210477046]]
2	Judul + Deskripsi	[[1283,1384],[1384,1175],[1175,1303],, ...,[1426,1263],[1263,1176]]	[[14210,15332],[15332,13040],[13040,14444],[14444,15238], ...,[14494,15783],[15783,14002]]	[[1720713,1856548],[1856548,1579163],[1579163,1748962], ...,[1739470,1579708],[1579708,1754950]]	[[208207596,224643546],[224643546,191080118],[191080118,211625808], ...,[222379340,205148172],[205148172,210477046]]

Tabel 3.12. Hasil Sebaran *Hash* pada *Window* Berdasarkan K-Gram pada *Text preprocessing* Menggunakan *Stemming* Algoritma Nazief & Adriani (Lanjutan)

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
...	...	...	...	...	...
100	Judul + Deskripsi	[[1283,1384],[1384,1175],[1175,1289], ...,[1293,1265],[1265,1311]]	[[14210,15332],[15332,13026],[13026,14289],[14289,13534], ...,[14603,14333],[14333,14016]]	[[1720699,1856393],[1856393,1577459],[1577459,1730210], ...,[1846059,1785894],[1785894,1768274]]	[[208205892,224624794],[224624794,190873832],[190873832,209356680], ...,[224257490,206320664],[206320664,223374450]]

3.4.5.4. *Fingerprint*

Setelah terbentuk *window* pada *hash text preprocessing* tanpa *stemming* dan *text preprocessing* mengunakan *stemming* algoritma Nazief & Adriani selanjutnya pada setiap *window* akan dipilih *fingerprint* yaitu nilai terkecil pada setiap *window* dan jika bertemu dengan nilai yang sama akan diambil nilai paling kanan dan ketika ada nilai yang sama akan dipilih salah satu. Sehingga hasil dari *fingerprint* didapatkan dengan nilai terkecil, berikut penentuan *fingerprint* pada *window* K-Gram pada data 1 dengan tipe judul dengan *window* dari *text preprocessing* tanpa *stemming* :

[[**1283**,1384],[1384,**1175**],[1175,1303],[**1303**,1376],[1376,**1323**],[1323,**1238**],[1238,1395],[**1395**,1406],[1406,1181],[**1181**,1355],[1355,1223],[**1223**,1333],[1333,1220],[**1220**,1297],[1297,1195],[**1195**,1384],[1384,1183],[**1183**,1373],[1373,1181],[**1181**,1370],[**1370**,1381],[1381,1262],[**1262**,1278],[1278,1219],[**1219**,1305],[**1305**,1397],[1397,**1315**],[1315,**1262**]]

Sehingga *fingerprint* ditemukan sebagai berikut :

[1283,1175,1303,1323,1238,1395,1181,1223,1220,1195,1183,1370,1262,1219,1305,1315]

Berikut hasil *fingerprint* pada *window* yang terbentuk dari *text preprocessing* tanpa *stemming* ditampilkan pada Tabel 3.13.

Tabel 3.13. Hasil *Fingerprint* Berdasarkan *Window* pada *Text preprocessing* Tanpa *Stemming*

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Judul	[1283,1175,1303,1323,1238,1395,1181, ... ,1262,1219,1305,1315]	[14210,13040,14444,14669,13737,15442,13092,13554, ... ,15175,13983,13526,14465,14572]	[1720713,1579163,1748962,1776355,1663358,1869837, ... ,1587209,1837453,1693248,1637961]	[208207596,191080118,211625808,214940310,201267541, ... ,192110542,192053508,222333118,204884323]
1	Deskripsi	[1323,1238,1395,1181,1223,1220,1195, ... ,1264,1174,1308,1263]	[14669,13737,15442,13092,13554,13518,13242,13110, ... ,14124,13029,14001,13045,14494,14002]	[1776355,1663358,1869837,1585465,1641331,1637062, ... ,1577773,1733653,1695429,1579708]	[214940310,201267541,226251610,191842562,198602435, ... ,190973749,190911708,209773321,205148172]
1	Judul + Deskripsi	[1283,1175,1303,1323,1238,1395,1181, ... ,1218,1264,1174,1308,1263]	[14210,13040,14444,14669,13737,15442,13092,13554, ... ,14124,13029,14001,13045,14494,14002]	[1720713,1579163,1748962,1776355,1663358,1869837, ... ,1577773,1733653,1695429,1579708]	[208207596,191080118,211625808,214940310,201267541,226251610, ... ,190973749,190911708,209773321,205148172]
2	Judul	[1283,1175,1303,1323,1238,1395,1181,1223,1220,1195,1183,1370,1262,1219]	[14210,13040,14444,14669,13737,15442,13092,13554,13518,13242,13110,13107,15175,13983]	[1720713,1579163,1748962,1776355,1663358,1869837, ... ,1603655,1587680,1587209,1837453]	[208207596,191080118,211625808,214940310,201267541, ... ,198085875,194043625,192110542,192053508]
2	Deskripsi	[1323,1238,1395,1181,1223,1220,1195, ... ,1174,1308,1263]	[14669,13737,15442,13092,13554,13518,13242,13110, ... ,14001,13045,14494,14002]	[1776355,1663358,1869837,1585465,1641331,1637062, ... ,1577773,1733653,1695429,1579708]	[214940310,201267541,226251610,191842562,198602435,198085875, ... ,190973749,190911708,209773321,205148172]

Tabel 3.13. Hasil *Fingerprint* Berdasarkan *Window* pada *Text preprocessing* Tanpa *Stemming* (Lanjutan)

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
2	Judul + Deskripsi	[1283,1175,1303,1323,1238,1395,1181, ... ,1218,1264,1174,1308,1263]	[14210,13040,14444,14669,13737,15442,13092,13554, ... ,14001,13045,14494,14002]	[1720713,1579163,1748962,1776355,1663358,1869837, ... ,1577773,1733653,1695429,1579708]	[208207596,191080118,211625808,214940310,201267541, ... ,190973749,190911708,209773321,205148172]
...	...	...	...	...	...
100	Judul + Deskripsi	[1283,1175,1221,1241,1270,1181,1370, ... ,1222,1337,1330,1318,1293]	[14210,13026,13534,13756,14074,13765,13107,15175, ... ,13118,14081,14749,14603,14333,14016]	[1720699,1577459,1638907,1665845,1704135,1666946, ... ,1588654,1705119,1785894,1768274]	[208205892,190873832,19830911 6,201568426,206201716, ... ,215786484,210575453,192228452,206320664]

Berikut hasil dari *fingerprint* berdasarkan *hash window* yang berasal dari *stemming* Nazief & Adriani, ditampilkan pada Tabel 3.14.

Tabel 3.14. Hasil *Fingerprint* Berdasarkan *Window* pada *Text preprocessing* Menggunakan *Stemming* Algoritma Nazief & Adriani

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Judul	[1283,1175,1303,1323,1238,1395,1181,1209,1195,1183,1370,1262,1219,1305,1315]	[14210,13040,14444,14669,13737,15442,13092,13416,13242,13110,13107,15175,13983,13526,14465,14572]	[1720713,1579163,1748962,1776355,1663358,1869837,1585327,1624519,1603655,1587680,1587209,1837453,1693248,1637961]	[208207596,191080118,211625808,214940310,201267527,226251472,191825750,196567980,194043625,192110542,192053508,222333118,204884323]
1	Deskripsi	[1323,1238,1395,1181,1209,1195,1183, ... ,1218,1264,1174,1308,1263]	[14669,13737,15442,13092,13416,13242,13110,13107,15175, ... ,14064,14001,13029,13045,14494,14002]	[1776355,1663358,1869837,1585327,1624519,1603655, ... ,1733653,1695429,1579708]	[214940310,201267527,226251472,191825750,196567980, ... ,190973749,190911708,209773321,205148172]

Tabel 3.14. Hasil *Fingerprint* Berdasarkan *Window* pada *Text preprocessing* Menggunakan *Stemming* Algoritma Nazief & Adriani (Lanjutan)

ID	Tipe	$k = 2$	$k = 3$	$k = 5$	$k = 7$
1	Judul + Deskripsi	[1283,1175,1303,1323,1238,1395,1181, ... ,1218,1264,1174,1308,1263]	[14210,13040,14444,14669,13737,15442,13092,13416,13242, ... ,14064,14001,13029,13045,14494,14002]	[1720713,1579163,1748962,1776355,1663358,1869837, ... ,1577773,1733653,1695429,1579708]	[208207596,191080118,211625808,214940310,201267527, ... ,190973749,190911708,209773321,205148172]
2	Judul	[1283,1175,1303,1323,1238,1395,1181,1209,1195,1183,1370,1262,1219]	[14210,13040,14444,14669,13737,15442,13092,13416,13242,13110,13107,15175,13983]	[1720713,1579163,1748962,1776355,1663358,1869837,1585327,1624519,1603655,1587680,1587209,1837453]	[208207596,191080118,211625808,214940310,201267527,226251472,191825750,196567980,194043625,192110542,192053508]
2	Deskripsi	[1323,1238,1395,1181,1209,1195,1183, ... ,1218,1264,1174,1308,1263]	[14669,13737,15442,13092,13416,13242,13110,13107, ... ,14001,13029,13045,14494,14002]	[1776355,1663358,1869837,1585327,1624519,1603655, ... ,1577773,1733653,1695429,1579708]	[214940310,201267527,226251472,191825750,196567980, ... ,190973749,190911708,209773321,205148172]
2	Judul + Deskripsi	[1283,1175,1303,1323,1238,1395,1181, ... ,1218,1264,1174,1308,1263]	[14210,13040,14444,14669,13737,15442,13092,13416,13242, ... ,14001,13029,13045,14494,14002]	[1720713,1579163,1748962,1776355,1663358,1869837, ... ,1577773,1733653,1695429,1579708]	[208207596,191080118,211625808,214940310,201267527, ... ,190973749,190911708,209773321,205148172]
...	...	...	...	...	...
100	Judul + Deskripsi	[1283,1175,1221,1241,1270,1181,1370, ... ,1330,1331,1318,1293]	[14210,13026,13534,13756,14074,13765,13107,15175, ... ,14749,14603,14333,14016]	[1720699,1577459,1638907,1665845,1704135,1666946, ... ,1588654,1705119,1785894,1768274]	[208205892,190873832,19830911 6,201568426,206201716, ... ,215786484,210575453,192228452,206320664]

Berdasarkan Tabel 3.13 dan Tabel 3.14 telah didapatkan hasil *fingerprint* untuk setiap tipe data yaitu judul, deksripsi dan judul+deskripsi dengan proses *stemming* yang berbeda yaitu dengan *text preprocessing* tanpa *stemming* dan *text*

preprocessing mengunakan *stemming* algoritma Nazief & Adriani, dehingga setiap id dokumen memiliki *fingerprint* masing-masing.

3.4.6. Jaccard *Similarity*

Berikut detail dari alur penelitian pada tahap Jaccard *Similarity* :

Gambar 3.4. Diagram alur penelitian tahap Jaccard *Similarity*.

Gambar 3.4 menjelaskan bahwa setelah pemrosesan algoritma Winnowing hasil dari *fingerprint* tersebut dihitung tingkat persentase *similarity*nya menggunakan koefisien Jaccard *Similarity* untuk membandingkan nilai *fingerprint* satu dokumen dengan dokumen yang lain berdasarkan *fingerprint* yang sama. Dengan urutan dari deklarasi koresponden *fingerprint string* A dan

B selanjutnya akan diketahui irisan dan gabungan antara koresponden *fingerprint string* A dan B dan terakhir perhitungan Jaccard *Similarity* dapat di lakukan.

3.4.6.1. Deklarasi *Fingerprint* Koresponden A dan B

Berikut ditampilkan perhitungan untuk mengetahui hasil dari tingkat *similarity* berdasarkan hasil *fingerprint* pada id dokumen 1 dibandingkan dengan dokumen 2 yang telah dihasilkan pada proses algoritma Winnowing, dengan memakai data pada *text preprocessing* mengunakan *stemming* algoritma Nazief & Adriani dengan tipe judul dan K-Gram dengan nilai $k = 2$.

Total *Fingerprint* Data 1 : 15

[1283,1175,1303,1323,1238,1395,1181,1209,1195,1183,1370,1262,1219,1305, 1315]

Total *Fingerprint* Data 2 : 13

[1283,1175,1303,1323,1238,1395,1181,1209,1195,1183,1370,1262,1219]

3.4.6.2. Deklarasi Irisan dan Gabungan Koresponden A dan B

Berdasarkan total *fingerprint* yang ada pada data koresponden A dan B sebelumnya maka diketahui total dari *fingerprint* pada setiap koresponden selanjutnya maka dari itu akan diketahui *fingerprint* yang sama antara koresponden A dan B sebagai berikut :

Fingerprint yang sama : 13

[1283,1175,1303,1323,1238,1395,1181,1209,1195,1183,1370,1262,1219]

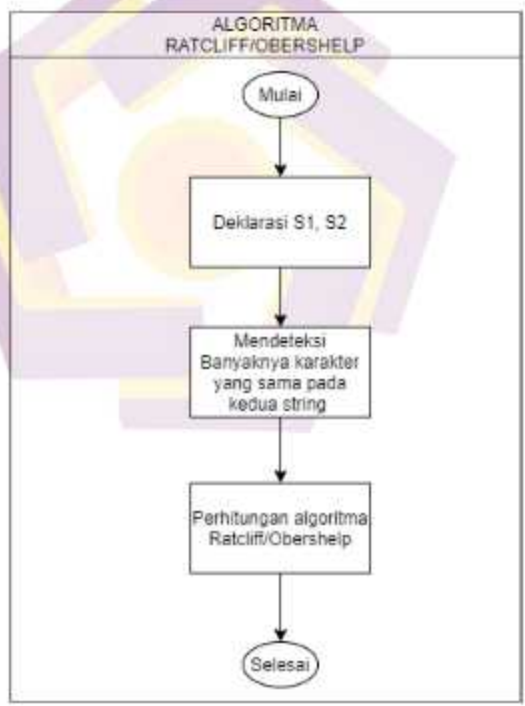
3.4.6.3. Perhitungan Jaccard *Similarity*

Setelah mengetahui *fingerprint* disetiap koresponden data dan mengetahui *fingerprint* yang sama antara kedua koresponden selanjutnya perhitungan Jaccard *Similarity* dapat dilakukan sebagai berikut :

$$J(A,B) = \frac{13}{((15+13)-13)}\ x100\% = 86{,}67\%$$

3.4.7. Algoritma Ratcliff/Obershelp

Berikut detail dari alur penelitian pada tahap algoritma Ratcliff/Obershelp:



Gambar 3.5. Diagram alur penelitian tahap Algoritma Ratcliff/Obershelp.

Gambar 3.5 menjelaskan bahwa, pada data uji dan data validasi juga dicari pencocokan *string* menggunakan algoritma ini, dengan menemukan sub *string* terpanjang yang memiliki kesamaan dari *string* satu dan *string* dua yang di sebut anchor, kemudian bagian yang tersisa dari *string* sebelah kiri dan kanan dari anchor harus diperiksa sebagai *string-string* yang baru, proses tersebut di ulangi sampai semua karakter dari *string string* satu dan *string* dua di analisa, dengan proses dari pendeklarasian S_1 dan S_2 selanjutnya mendeteksi banyaknya karakter yang sama pada kedua string lalu langkah terakhir menghitung algoritma Ratcliff/Obershelp dengan persamaan DrO.

3.4.7.1. Deklarasi S_1 dan S_2

Berikut contoh dari perhitungan dari alur proses algoritma Ratcliff/Obershelp dengan memakai data *text preprocessing* mengunakan *stemming* algoritma Nazief & Adriani pada data 1 dan data 2 dengan jenis data *input*an karakter dari *stemming* Judul, sehingga deklarasi S_1 dan S_2 sebagai berikut :

Panjang Karakter Data 1 (S_1) = 27

"jualsoftwarebuatartikelunik"

Panjang Karakter Data 2 (S_2) = 23

"jualsoftwarebuatartikel"

3.4.7.2. Mendeteksi Banyaknya Karakter yang Sama pada Kedua String

Setelah diketahui deretan karakter dan jumlah karakter yang terdapat pada S_1 dan S_2 Maka dapat ditentukan kesamaan karakter yang ada pada keduanya, sehingga penentuan identifikasi huruf yang sama antara S_1 dan S_2 sebagai berikut :

Tabel 3.15. Identifikasi Huruf yang Sama pada S_1 dan S_2

ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
S_1	j	u	a	l	s	o	f	t	w	a	r	e	b	u	a	t	a	r	t	i	k	e	l	u	n	i	k
S_2	j	u	a	l	s	o	f	t	w	a	r	e	b	u	a	t	a	r	t	i	k	e	l				

Berdasarkan Tabel 3.15 diketahui bahwa blok karakter yang sama dengan jumlah yang panjang atau disebut anchor ditemukan pada karakter **jualsoftwarebuatartikel** yang terdapat pada blok karakter 0 sampai 23 pada S_1 dan terdapat pada blok karakter 0 sampai 23 pada S_2. Namun setelah anchor ditemukan tidak ditemukan lagi string yang sama dari perpotongan dari anchor karena karakter pada S_2 telah habis dan tidak ada lagi sehingga jumlah string yang sama yaitu sebanyak 23.

k_m = 23

3.4.7.3. Perhitungan Algoritma Ratcliff/Obershelp

Setelah mengetahui nilai dari panjang karakter pada S_1 dan S_2 serta mengetahui banyaknya karakter yang sama antara keduanya, sehingga perhitungan dari algoritma Ratcliff/Obershelp sebagai berikut :

$$D_ro = \frac{(2 * 23)}{27 + 23}$$

$$= \frac{46}{50}$$

$$= 0{,}92\ (92\%)$$

3.4.8. Perbandingan

Penerapan *text preprocessing* tanpa *stemming* dan *text preprocessing* mengunakan *stemming* algoritma Nazief & Adriani dilakukan pada tahap proses *stemming* di masing-masing algoritma Winnowing dan algoritma Ratcliff/Obershelp. Selanjutnya terdapat dua hasil presentase akurasi dan efisiensi

waktu pada setiap penerapan algoritma karena menggunakan dua jenis penerapan *stemming*, selanjutnya hasil tersebut dibandingkan pada masing-masing algoritma maka akan diketahui performa *stemming* Nazief & Adriani. Serta dilakukan perbandingan pada kedua algoritma antara algoritma Winnowing dan algoritma Ratcliff/Obershelp untuk mengetahui performa *stemming* Nazief & Adriani pada perbandingan kedua algoritma tersebut dan akan diketahui hasil dari perbandingan presentase akurasi dan efisiensi waktu pada kedua algoritma tersebut.

Jika penerapan *stemming* Nazief & Adriani pada masing-masing algoritma memperoleh hasil bahwa mempercepat pemrosesan algoritma maka dapat disimpulkan penerapan *stemming* Nazief & Adriani mempercepat efisiensi waktu. Performa dalam hal akurasi algoritma *similarity* dengan penerapan *stemming* Nazief & Adriani diketahui berdasarkan pada masing-masing persentase hasil *similarity*, menurut ketentuan jika data validasi dari produk yang dibandingkan memiliki indikasi produk tidak duplikat maka akurasi jika semakin mendekati nilai 0% maka performa *stemming* Nazief & Adriani tersebut baik karena dengan nilai hasil *similarity* dibawah kurang dari 30% merupakan nilai plagiarisme ringan, jika validasi dari produk yang dibandingkan memiliki indikasi produk duplikat maka akurasi jika semakin mendekati nilai 100% maka performa *stemming* Nazief & Adriani tersebut baik karena nilai lebih dari 70% untuk nilai *similarity* berdasarkan hasil produk yang dibandingkan merupakan plagiarisme berat atau total.

Ketentuan tersebut juga digunakan pada penerapan *stemming* Nazief & Adriani pada perbandingan kedua algoritma. Jika terjadi proses percepatan waktu pada perbandingan antara algoritma Winnowing dan Ratcliff/Obershelp maka dapat

disimpulkan penerapan *stemming* Nazief & Adriani mempercepat efisiensi waktu pada algoritma yang dibandingkan. Perbandingan pada performa penerapan *stemming* Nazief & Adriani juga berlaku sama dengan tingkat akurasi yaitu jika nilainya berkurang pada kriteria validasi dari produk yang dibandingkan tidak memiliki indikasi kesamaan atau indikasi produk tidak duplikat maka akurasi jika semakin mendekati nilai 0% dengan nilai dibawah kurang dari 30% maka performa *stemming* Nazief & Adriani tersebut baik karena dari nilai tersebut termasuk plagiarisme ringan, jika validasi dari produk yang dibandingkan memiliki indikasi kesamaan atau indikasi produk duplikat maka akurasi jika semakin mendekati nilai 100% maka performa *stemming* Nazief & Adriani tersebut baik karena nilai lebih dari 70% termasuk plagiarisme berat atau total.

Hasil persentase akurasi *similarity* pada algoritma Winnowing dan Ratcliff/Obershelp tersebut dibandingkan dengan kriteria yang ditentukan manual pada data validasi, jika data validasi dari produk yang dibandingkan memiliki indikasi produk tidak duplikat maka jika nilai *similarity* mendekati nilai 0% maka algoritma tersebut semakin akurat, jika validasi produk yang dibandingkan memiliki indikasi produk duplikat maka jika nilai *similarity* mendekati nilai 100% maka algoritma tersebut semakin akurat. Hasil kecepatan pemrosesan pada masing-masing algoritma dibandingkan, waktu pemrosesan tercepat berarti memiliki efisiensi waktu lebih cepat.

3.4.9. Dokumentasi Hasil Penelitian

Hasil pada penelitian ini adalah mengetahui hasil performa *stemming* Nazief & Adriani dan mengetahui hasil perbandingan presentase akurasi, efisiensi waktu algoritma Winnowing dan algoritma Ratcliff/Obershelp.

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

4.1. Gambaran Umum Penelitian

Dalam penelitian ini data yang akan diteliti adalah teks yang diambil menggunakan *text mining* yang berasal dari teks judul dan deskripsi pada produk pada p-store.net yang berjumlah 100 produk yang nantinya akan dibedakan menjadi 3 masukan data yaitu bertipe judul, deskripsi dan judul + deskripsi. Jumlah data tersebut ditentukan oleh rumus Wibisono. Data yang diambil saling berpasang-pasangan untuk nantinya dibandingkan dan memiliki indikasi awal antara duplikat dan tidak duplikat. Setelah dilakukan pengambilan data selanjutnya data yang ada akan dibandingkan keduanya diproses pada tahap *text preprocessing*, pada tahap *text preprocessing* dibedakan menjadi dua yaitu *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan *stemming* algoritma Nazief & Adriani. Setelah dilakukan tahap *text preprocessing* selanjutnya diproses pada tahapan algoritma Winnowing dengan pemilihan nilai K-Gram bilangan prima 2, 3, 5 dan 7 dengan nilai *window* yaitu 2. Berdasarkan algoritma Winnowing akan dilakukan tahapan pemrosesan yaitu menentukan nilai K-Gram, Rolling Hash, pembuatan *window* dan *fingerprint*. Setelah terbentuk *fingerprint* pada setiap data maka akan diproses pada perhitungan Jaccard *Similarity* untuk mengetahui hasil *similarity* pada data yang diproses menggunakan algoritma Winnowing. Berikutnya yaitu melakukan pemrosesan pada algoritma Ratcliff/Obershelp dengan menggunakan data *text preprocessing* tanpa *stemming* dan *text preprocessing* mengunakan

stemming algoritma Nazief & Adriani. Data *text preprocessing* diolah dengan menggabungkan semua karakternya yang akan diketahui jumlah panjanga pada setiap data yang dibandingkan, selanjutnya akan dicari karakter yang sama diantara keduanya jika ditemukan karakter yang sama dengan jumlah yang paling panjang maka akan disebut anchor. Setelah diketahui jumlah panjang karakter pada setiap data dan jumlah karakter yang sama diantara kedua data yang dibandingkan maka akan dihitung *similarity*nya menggunakan rumus dari algoritma Ratcliff/Obershelp.

Berdasarkan hasil perhitungan tingkat kesamaan kedua data yang dibandingkan menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp. Maka akan dibandingkan performa yang dilakukan menggunakan perhitungan kecepatan waktu serta akurasi penggunaan *stemming* algoritma Nazief & Adriani pada setiap algoritma itu sendiri yaitu dengan membandingkan performa hasil *similarity* berdasarkan *text preprocessing* tanpa *stemming* dan *text preprocessing* mengunakan *stemming* algoritma Nazief & Adriani, serta membandingkan hasil *similarity* pada algoritma Winnowing dan algoritma Ratcliff/Obershelp.

4.2. Pengumpulan Data

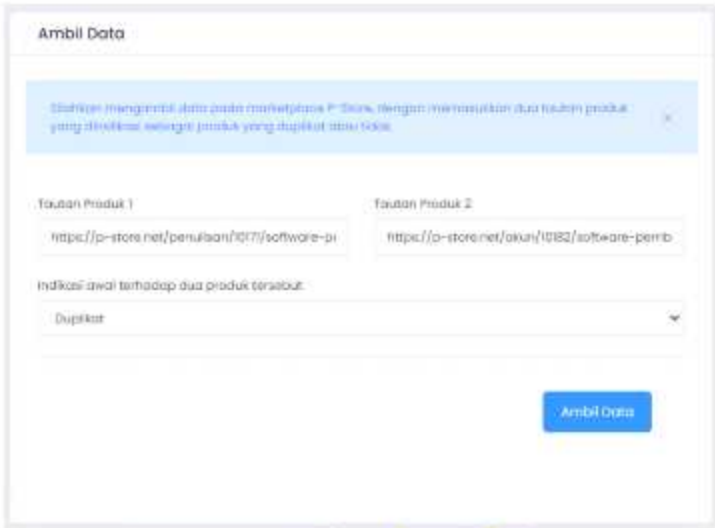
Pada penelitian ini menggunakan pemilihan sampel data berdasarkan rumus Wibisono dan didapatkan data sebanyak 100 produk yang akan dibagai menjadi 50% untuk data dengan indikasi duplikat dan 50% untuk data dengan indikasi tidak duplikat, berdasarkan indikasi awal tersebut data akan dibandingkan dengan pasangannya sehingga akan terbentuk 25 analisa setiap indikasinya. Data tersebut diambil berdasarkan masukan berupa 2 *link* produk dan produk tersebut telah

diindikasikan sebagai produk duplikat atau produk tidak duplikat yang dilakukan oleh seller, sehingga data yang dikumpulkan diambil berdasarkan beberapa seller pada p-store dan selanjutnya data terebut akan di*scrape* sehingga dapat mendapatkan data yang dibutuhkan yaitu judul, deskripsi, link, penjual, terjual, produk_dibuat, produk_di*scrape*, pembanding, indikasi dan pelaku. Untuk proses pengambilan data dilakukan dengan menggunakan *scrape* dari PHP Curl dan akan diparsing. *Source Code* tersebut dapat dilihat pada Gambar 4.1.

```
function ambil_post($domain){
    $hasil = array();
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $domain);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_ENCODING, "");
    curl_setopt($ch, CURLOPT_MAXREDIRS, 10);
    curl_setopt($ch, CURLOPT_TIMEOUT, 0);
    curl_setopt($ch, CURLOPT_FOLLOWLOCATION, true);
    curl_setopt($ch, CURLOPT_HTTP_VERSION, CURL_HTTP_VERSION_1_1);
    curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "GET");
    curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0);
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
    $headers = array();
    $headers[] = 'Upgrade-Insecure-Requests: 1';
    $headers[] = 'User-Agent: Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/76.0.3809.100 Mobile Safari/537.36';
    $headers[] = 'Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,appl
ication/signed-exchange;v=b3;q=0.9';
    $headers[] = 'Sec-Fetch-Site: none';
    $headers[] = 'Sec-Fetch-Mode: navigate';
    $headers[] = 'Sec-Fetch-Dest: document';
    $headers[] = 'Accept-Language: id,en-US;q=0.9,en;q=0.8';
    curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);
    $result = curl_exec($ch);
    if (curl_errno($ch)) {
        echo 'Error:' . curl_error($ch);
    }
    curl_close($ch);
    if(!empty($result)){
        $batas_awal_judul = explode('<h1>',$result);
        $batas_akhir_judul = explode('</h1>',$batas_awal_judul[1]);
        $judul = $batas_akhir_judul[0];

        $batas_awal_deskripsi_notif = explode('<div class="gig-description"
align="left">',$result);
        $batas_akhir_deskripsi_notif = explode('<img data-cfsrc="https://p-
store.net/images/logo-blue.png",$batas_awal_deskripsi_notif[1]);
        $deskripsi_notif = $batas_akhir_deskripsi_notif[0];
        $batas_awal_deskripsi = explode('<div class="alert-info" ,$deskripsi_notif);
        $batas_akhir_deskripsi = explode(' Berbelanja/an</p></div><br
/>',$batas_awal_deskripsi[1]);
        $deskripsi = $batas_akhir_deskripsi[0];

        $batas_awal_json_meta = explode('<script type="application/ld+json">',$result);
        $batas_akhir_json_meta = explode('</script>',$batas_awal_json_meta[2]);
        $json_meta = json_decode($batas_akhir_json_meta[0],true);

        $batas_awal_dibuat_meta = explode('Dibuat : ',$result);
        $batas_akhir_dibuat_meta = explode('<',$batas_awal_dibuat_meta[1]);
        $dibuat_meta = strtotime($batas_akhir_dibuat_meta[0]);
        $dibuat_meta = date('Y-m-d H:i:s', $dibuat_meta);

        $batas_awal_terjual = explode('TERJUAL : ',$result);
        $batas_akhir_terjual = explode('</span>',$batas_awal_terjual[1]);
        $terjual = $batas_akhir_terjual[0];

        $batas_awal_penjual = explode('<b>Produk/Jasa lainnya dari : ',$result);
        $batas_akhir_penjual = explode("</b>",$batas_awal_penjual[1]);
        $penjual = $batas_akhir_penjual[0];
    }
    $hasil['judul'] = $judul;
    $hasil['deskripsi'] = $deskripsi;
    $hasil['link'] = $domain;
    $hasil['terjual'] = $terjual;
    $hasil['penjual'] = $penjual;
    $hasil['produk_dibuat'] = $dibuat_meta;
    $hasil['produk_discrape'] = date('Y-m-d H:i:s');
    return $hasil;
}
```

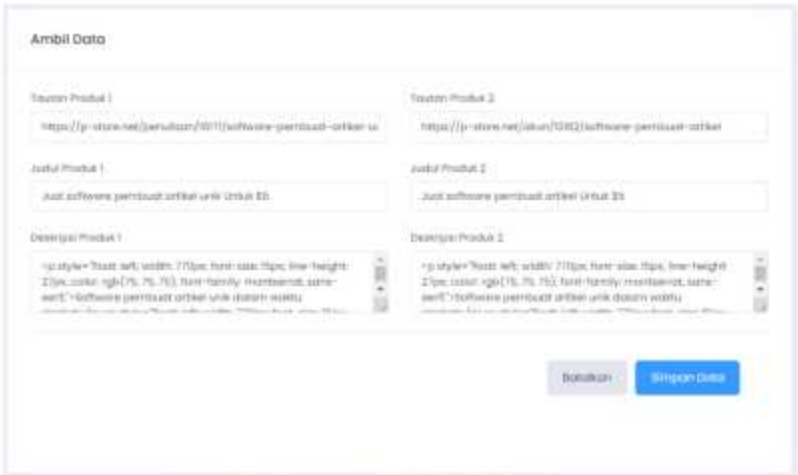
Gambar 4.1. *Source Code* PHP *Scrape* Data

Berdasarkan Gambar 4.1 maka akan membutuhkan masukan berupa link sebuah produk yang nantinya akan dilakukan proses *scraping*, sehingga antarmuka untuk melakukan pengambilan data dengan masukan berupa link produk ditampilkan pada Gambar 4.2.



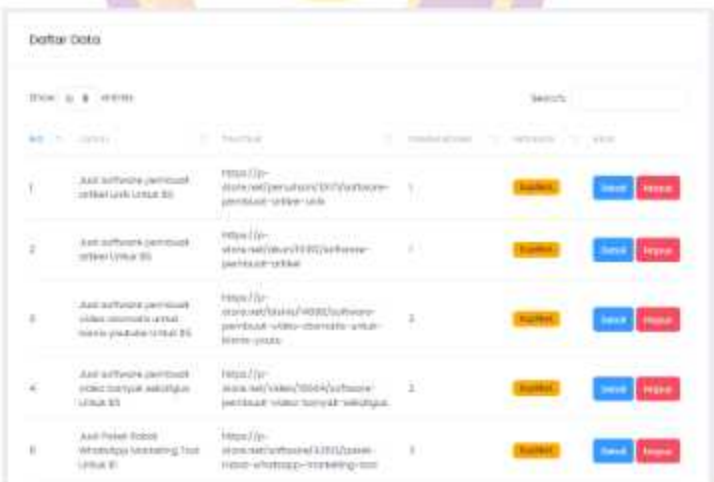
Gambar 4.2. Antarmuka Pengambilan Data

Setelah dilakukan *scraping* data selanjutnya data akan ditampilkan untuk mengkonfirmasi dari detail data yang di*scrape*, antarmuka konfirmasi penyimpanan data ditampilkan pada Gambar 4.3.



Gambar 4.3. Antarmuka Konfirmasi Pengambilan Data

Setelah dilakukan konfirmasi penyimpanan data selanjutnya data disimpan berdasarkan indikasi dan pembandingnya, data yang disimpan dapat dilihat pada antarmuka daftar data yang ditampilkan pada Gambar 4.4.



Gambar 4.4. Antarmuka Daftar Data

Berdasarkan data yang disimpan akan dapat dilihat detail dari setiap data yang ada untuk mengetahui detail lebih lanjut mengenai setiap data tersebut, antarmuka detail data ditampilkan pada Gambar 4.5.

Gambar 4.5. Antarmuka Detail Data

4.3. *Text preprocessing*

Proses selanjutnya setelah melakukan penyimpanan data, data yang telah dipasangkan berdasarkan indikasi dan pembandingnya selanjutnya data tersebut diolah pada proses *text preprocessing*.

4.3.1 *Case folding*

Langkah pertama pada *text preprocessing* yaitu *case folding* untuk menghilangkan karakter yang tidak valid dan merubah huruf menjadi kecil. Berikut *Source Code* yang digunakan untuk pemrosesan *case folding* ditampilkan pada Gambar 4.6.

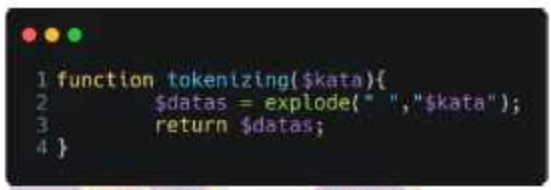
Gambar 4.6. *Source Code* PHP *Case folding*

Berdasarkan Gambar 4.6 data teks yang nantinya diproses pada tahap awal *text preprocessing* akan dibedakan menjadi 3 masukan yaitu bertipe judul, deskripsi dan judul + deskripsi. Pada baris ke-dua pada gambar tersebut akan menghilangkan syntax html dan akan diganti dengan spasi hal ini dilakukan menggunakan strip_tags karena jika menggunakan strip_tags maka syntax akan dihilangkan namun tanpa menggantinya dengan spasi maka efeknya banyak kata yang akan bersatu tanpa spasi, maka dari itu dilakukan regex #<[^>]+>#. Baris ke-tiga dengan membuat karakter menjadi kecil menggunakan strtolower serta baris ke-empat akan

memfilter karakter hanya huruf kecil a sampai z sehingga akan membuang karakter yang tidak penting.

4.3.2 *Tokenizing*

Tahap berikutnya adalah memecah dari hasil *Tokenizing* kedalam beberapa array dan di pecah berdasarkan spasi sehingga nantinya data akan dapat diolah per kata. *Source Code* dari *tokenizing* ditampilkan pada Gambar 4.7.



```
function tokenizing($kata){
        $datas = explode(" ","$kata");
        return $datas;
}
```

Gambar 4.7. *Source Code* PHP *Tokenizing*

Gambar 4.7 menjelaskan tentang proses pemotongan kata, pada baris ke-2 menjelaskan bahwa kata akan dipisah menggunakan explode yang akan membentuk sebuah array.

4.3.3 *Filtering*

Filtering dilakukan untuk menghilangkan kata yang tidak penting seperti kata sambung sehingga nantinya hanya akan menyisakan kata penting, berikut *Source Code* dari *filtering* ditampilkan pada Gambar 4.8.

```
function filtering($datas){
        global $mysqli;
        foreach($datas as $data){
        $query = $mysqli->query("SELECT stopword FROM td_stopword WHERE
stopword='$data'");
            $row = $query->num_rows;
                if($row!=1){
                    $stopword[] = "$data";
                }
        }
        if (empty($stopword)){
            $stopword[] = "null";
        }
        return $stopword;
}
```

Gambar 4.8. *Source Code* PHP *Filtering*

Berdasarkan Gambar 4.8 dilakukan proses pencocokan setiap kata yang ada pada array yang telah terbentuk sebelumnya pada proses *tokenizing*, pada baris ke-4 menunjukkan bahwa kata tersebut dicocokkan kedalam tabel stopword, jika kata tersebut ditemukan pada tabel stopword maka kata itu akan dibuang dan sebaliknya jika kata tersebut tidak ditemukan maka kata akan disimpan.

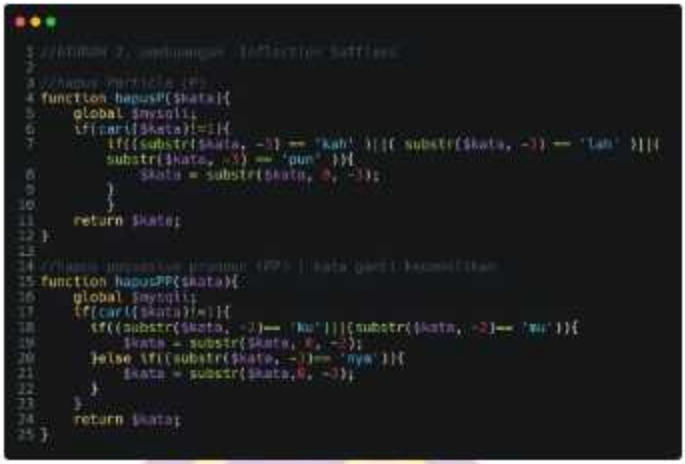
4.3.4 *Stemming*

Langkah terakhir pada tahap *text preprocessing* yaitu *stemming* yang mengubah kata menjadi kata dasar, pada penelitian ini menggunakan dua tipe keluaran *stemming* yaitu tanpa *stemming* yang datanya berasal dari *filtering* dan *stemming* dengan menggunakan algoritma *stemming* Nazief & Adriani. Berikut ditampilkan *Source Code* yang digunakan untuk mengubah kata menjadi kata dasar berdasarkan aturan algoritma Nazief & Adriani, *Source Code* aturan pada langkah ke-1 ditampilkan pada Gambar 4.9.

```
1 // fungsi 1. pencarian kata dasar
2 function cari($kata){
3     global $mysqli;
4     $query = $mysqli->query("SELECT * FROM tb_katadasar WHERE
  katadasar='$kata'");
5     return $query->num_rows;
6 }
```

Gambar 4.9. *Source Code* PHP *Stemming* Nazief & Adriani Aturan ke-1

Gambar 4.9 menjelaskan bahwa pada baris ke-4 akan mencari masukan kata kedalam kamus kata dasar, jika kata ditemukan maka diasumsikan bahwa kata tersebut adalah kata dasar. Jika tidak maka akan lanjut pada aturan ke-2 dengan *Source Code* yang ditunjukkan pada Gambar 4.10.



```
//ATURAN 2, menghilangkan Infleksi Suffixes

//hapus particle (P)
function hapusP($kata){
    global $mysqli;
    if(car($kata)!=1){
        if((substr($kata, -3) == 'kah' )||( substr($kata, -3) == 'lah' )||(
        substr($kata, -3) == 'pun' )){
            $kata = substr($kata, 0, -3);
        }
        }
    return $kata;
}

//hapus possessive pronoun (PP) | kata ganti kepemilikan
function hapusPP($kata){
    global $mysqli;
    if(car($kata)!=1){
      if((substr($kata, -2)== 'ku')||(substr($kata, -2)== 'mu')){
          $kata = substr($kata, 0, -2);
      }else if((substr($kata, -3)== 'nya')){
          $kata = substr($kata,0, -3);
      }
    }
    return $kata;
}
```

Gambar 4.10. *Source Code* PHP *Stemming* Nazief & Adriani Aturan ke-2

Gambar 4.10 merupakan proses langkah ke-2 yaitu melakukan pemeriksaan aturan ke-2 yaitu menghapus Sufiks Infleksi ("-lah", "-kah", "-ku", "-mu", atau "-nya"), pada baris ke-tujuh menandakan bahwa pertama menghapus ("-lah", "-kah", "-tah" atau "-pun") dan langkah diulangi lagi untuk menghapus kata ganti obsesif yang ditunjukkan pada baris ke-delapan belas dan baris ke-dua puluh ("-ku", "-mu", atau "-nya"). Selanjutnya dilakukan pengecekan untuk aturan ke-3 ditunjukkan *Source Code*nya pada Gambar 4.11.

```
//ATURAN 3 menghapus akhiran
function hapusakhiran($kata){
global $mysqli;
if(cari($kata)!=1){

    if (substr($kata, -3) == "kan" ){
       $kata = substr($kata, 0, -3);
       //if(cari($kata)!=1){ $kata = $kata."kan";}
    }else if(substr($kata, -2)== "an"){
        $kata = substr($kata, 0, -2);
        //if(cari($kata)!=1){ $kata = $kata."an";}
    }else if(substr($kata, -1)== "i" ){
        $kata = substr($kata, 0, -1);
        //if(cari($kata)!=1){ $kata = $kata."i";}
    }
}
    return $kata;
}
```

Gambar 4.11. *Source Code* PHP *Stemming* Nazief & Adriani Aturan ke-3

Berdasarkan Gambar 4.11 dilakukan proses penghapusan akhiran yang merupakan sufiks derivasi ("-i", "-an" atau "-kan") yang ditunjukkan pada baris ke-enam, ke-sembilan dan ke-dua belas. Langkah selanjutnya adalah aturan ke-4 yaitu menghapus awalan *Source Code*nya ditampilkan pada Gambar 4.12.

Gambar 4.12. *Source Code* PHP *Stemming* Nazief & Adriani Aturan ke-4

Source Code pada Gambar 4.12 menjelaskan mengenai pengecekkan kata dengan awalan (be, di, ke, me, se) yang beserta diidentifikasi imbuhan belakang dari kata tersebut ditunjukkan pada baris ke-lima sampai ke-sebelas selanjutnya kata akan di proses untuk melakukan pemotongan, pada baris ke-lima belas dan sampai selesai dilakukan proses pemotongan kata awalan seperti (ber, bel, be, per dan lainnya). Selanjutnya langkah pada aturan ke-5 yaitu tentang pengodean ulang yaitu mengubah kata yang telah dipotong awalan dan akhiran untuk dapat menjadi kata dasar yang sesuai, *Source Code* pada aturan ke-5 ditampilkan pada Gambar 4.13.

Gambar 4.13. *Source Code* PHP *Stemming* Nazief & Adriani Aturan ke-5

Berdasarkan potongan *Source Code* pada Gambar 4.13 dijelaskan bahwa terjadinya proses pengodean ulang seperti pada baris ke-tiga mengidentifikasi apakah huruf awal pertama sampai empat berupa "meny" jika iya akan dirubah menjadi "s" begitu pula dengan terusan kode lainnya yang mengidentifikasi untuk penggantian huruf. Selanjutnya langkah ke-6 yaitu jika semua langkah telah selesai

dilakukan namun tidak berhasil, maka kata pertama diasumsikan sebagai kata dasar dan proses diselesaikan.

4.4 Data Uji dan Data Validasi

Data akan memiliki dua varian dari tahap *text preprocessing* yaitu *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma *stemming* Nazief & Adriani, data validasi ini data yang sebelumnya dipilih oleh beberapa seller untuk diindetifikasi manual dengan kriteria produk sejenis tapi duplikasi, produk sejenis tidak duplikasi. Data uji dan data Validasi akan diproses ke dalam algoritma *similarity* untuk mengetahui tingkat *similarity* dan akan dinilai akurasinya berdasarkan kriteria yang di tentukan.

4.5 Algoritma Winnowing

Tahapan proses algoritma Winnowing yaitu menentukan nilai K-Gram, Rolling Hash, pembuatan *window* serta terakhir menentukan *fingerprint*. Sehingga nantinya setiap data pada jenis masukkan tipe data yaitu judul, deskripsi, dan judul+deskripsi memiliki identitas *fingerprint* masing-masing.

4.5.1 Menentukan K-Gram

Pada penelitian ini digunakan nilai *k* yaitu merupakan bilangan prima 2, 3, 5 dan 7. Nilai *k* tersebut selanjutnya digunakan untuk memecah rangkaian karakter yang berasal dari hasil *text preprocessing*. Berikut *Source Code* yang digunakan untuk memotong karakter sebanyak jumlah *k* ditampilkan pada Gambar 4.14.

Gambar 4.14. *Source Code* PHP Algoritma Winnowing Tahap Menentukan K-Gram

Berdasarkan Gambar 4.14 dilakukan proses pemotongan karakter berdasarkan masukan nilai *k*, nilai *k* sendiri jika tidak dideklarasikan memiliki nilai = 3, pada baris ke-sembilan sampai ke-sebelas dilakukan proses pengecekkan panjang karakter dari kata dan dipotong berdasarkan jumlah nilai *k* tersebut.

4.5.2 Rolling Hash

Setelah dilakukan pemotongan karakter selanjutnya dilakukan proses *hashing* menggunakan Rolling Hash dengan nilai basis bilangan prima 11. Berikut *Source Code* Rolling Hash ditampilkan pada Gambar 4.15.

Gambar 4.15. *Source Code* PHP Algoritma Winnowing Tahap Rolling Hash

Tahapan pembuatan *hashing* menggunakan Rolling Hash yang ditampilkan pada Gambar 4.15 menjelaskan bahwa terjadi perhitungan *hash* dengan rumus Rolling Hash yang ada pada baris ke-tujuh yang menggunakan masukkan bilangan ascii pada string, nilai basis dan nilai pangkat dari jumlah karakter gram.

4.5.3 Pembuatan *Window*

Setelah diketahui nilai *hash* pada setiap gram yang terbentuk selanjutnya memasukkan nilai *hash* tersebut kedalam sebuah *window* sehingga akan berkelompok membentuk *window* masing-masing sesuai urutannya. *Source Code* pembuatan *window* ditampilkan pada Gambar 4.16.

```php
function window($hash, $dibagi){
    $keluaran = array();
    $panjang = count($hash);
    $index = 0;
    if ($panjang <= $dibagi) {
        for($i=0;$i<$panjang;$i++){
        $keluaran[$index][$i] = $hash[$i];
        }
    }else{
        for($batas=0;$batas<=$panjang-$dibagi;$batas++){
            for($i=0;$i<$dibagi;$i++){
                $masukan = $index+$i;
                if($hash[$masukan]){
                    $keluaran[$index][$i] = $hash[$masukan];
                }
            }
            $index++;
        }
    }
    return $keluaran;
}
```

Gambar 4.16. *Source Code* PHP Algoritma Winnowing Tahap Pembuatan *Window*

Source Code pada Gambar 4.16 menjelaskan bahwa terjadinya pembuatan *window* yang terdapat pada baris ke-sepuluh sampai baris ke-delapan belas, yang membuat array dengan masukkan berupa perulangan data *hash* sampaia atau sesuai dengan jumlah *window*.

4.5.4 *Fingerprint*

Langkah *fingerprint* adalah menentukan nilai terkecil pada setiap *window* dan jika ada nilai yang sama maka akan diambil salah satu. *Source Code* dari penentuan nilai *fingerprint* ditampilkan pada Gambar 4.17.

```php
function fingerprint_kecil($window){
    $keluaran = array();
    $panjang = count($window);
    for($i=0;$i<$panjang;$i++){
        $min_value = min($window[$i]);
        $keluaran[$i] = $min_value;
    }

    $keluaran = array_values(array_unique($keluaran));
    return $keluaran;
}
```

Gambar 4.17. *Source Code* PHP Algoritma Winnowing Tahap *Fingerprint*

Gambar 4.17 menjelaskan proses penentuan nilai *fingerprint* dengan menentukan nilai terkecil yaitu pada baris ke-lima dilakukan pengecekan nilai terkecil dari anggota *window* serta pada baris ke-sembilan dilakukan identifikasi nilai unik pada setiap *fingerprint* yang ada.

4.6 Jaccard *Similarity*

Setelah ditemukan nilai *fingerprint* dari setiap sebaran nilai *k* yang berbeda serta pada sebaran tipe data yang berbeda selanjutnya dilakukan pencocokan nilai *fingerprint* yang sama untuk mendapatkan nilai *similarity*.

4.6.1 Deklarasi *Fingerprint* Koresponden A dan B

Langkah pertama pada proses Jaccard *Similarity* yaitu mendeklarasikan jumlah *fingerprint* dari setiap koresponden, berikut *Source Code* yang digunakan untuk mengidentifikasi *fingerprint* pada setiap koresponden ditampilkan pada Gambar 4.18.

```php
function bandingkan_fingerprint($fing1, $fing2){
    $simpan = array();

    $jumlah_1 = count($fing1);
    $jumlah_2 = count($fing2);

    ...

    return $keluaran;
}
```

Gambar 4.18. *Source Code* PHP Jaccard *Similarity* Tahap Deklarasi *Fingerprint*

Gambar 4.18 menjelaskan tentang alur dari identifikasi *fingerprint*, pada baris ke-satu menunjukkan masukkan dari fungsi adalah berupa data *fingerprint* data A dan data B, pada baris ke-empat dan ke-lima adalah untuk mengidentifikasi jumlah *fingerprint* pada setiap koresponden.

4.6.2 Deklarasi Irisan dan Gabungan Koresponden A dan B

Setelah diketahui *fingerprint* pada setiap koresponden serta diketahui jumlah *fingerprint* masing-masing koresponden selanjutnya diidentifikasi mengenai kesamaan *fingerprint* pada keduanya. *Source Code* penentuan *fingerprint* yang sama ditampilkan pada Gambar 4.19.

```
1  function bandingkan_fingerprint($fing1, $fing2){
2      ...
3
4      if($jumlah_1<$jumlah_2){
5          for($i=0;$i<count($fing1);$i++){
6              if (in_array($fing1[$i], $fing2)){
7                  $simpan[] = $fing1[$i];
8              }
9          }
10     }else{
11         for($i=0;$i<count($fing2);$i++){
12             if (in_array($fing2[$i], $fing1)){
13                 $simpan[] = $fing2[$i];
14             }
15         }
16     }
17     $deteksi_sama = count($simpan);
18
19     ...
20
21     return $keluaran;
22 }
```

Gambar 4.19. *Source Code* PHP Jaccard *Similarity* Tahap Gabungan *Fingerprint*

Baris ke-empat sampai ke-enam belas digunakan untuk mendeteksi *fingerprint* yang sama dengan in_array, dan pada baris ke-tujuh belas untuk menghitung jumlah *fingerprint* yang sama sehingga diketahui jumlah *fingerprint* yang sama diantara kedua koresponden tersebut.

4.6.3 Perhitungan Jaccard *Similarity*

Langkah terakhir adalah menghitung *similarity* dengan rumus Jaccard *Similarity*, *Source Code* perhitungan tersebut ditampilkan pada Gambar 4.20.

```php
function bandingkan_fingerprint($fing1, $fing2){
    $simpan = array();

    ...

    $cek_zero = (($jumlah_1 + $jumlah_2)-$deteksi_sama);
    if($cek_zero == 0 || $deteksi_sama == 0){
        $hasil = 0;
    }else{
        $hasil = $deteksi_sama / (($jumlah_1 + $jumlah_2)-$deteksi_sama);
    }

    $keluaran["jumlah_1"] = $jumlah_1;
    $keluaran["jumlah_2"] = $jumlah_2;
    $keluaran["deteksi_sama"] = $deteksi_sama;

    $keluaran["data_sama"] = json_encode($simpan);
    $keluaran["hasil"] = str_replace( '.', ',',round($hasil,4));
    return $keluaran;
}
```

Gambar 4.20. *Source Code* PHP Jaccard *Similarity* Tahap Perhitungan *Similarity*

Berdasarkan Gambar 4.20 dijelaskan bahwa pada baris ke-enam dan ke-tujuh melakukan deteksi dan identifikasi apakah tidak ada *fingerprint* yang sama dan akan menghasilkan nilai 0, sedangkan untuk baris ke-sepuluh dilakukan proses perhitungan Jaccard *Similarity*. Sehingga pada tahap ini nilai dari *similarity* pada algoritma Winnowing yang dihitung dengan Jaccard *Similarity* dapat diketahui.

4.7 Algoritma Ratcliff/Obershelp

Proses berikutnya yaitu menghitung tingkat *similarity* menggunakan algoritma Ratcliff/Obershelp. Masukkan dari algoritma ini yaitu berupa hasil dari *text preprocessing* yang telah dibedakan sebelumnya yaitu *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma *stemming* Nazief & Adriani. Tahapan dari algoritma Ratcliff/Obershelp yaitu mendeklarasikan S_1 dan S_2,

mendeteksi banyaknya karakter yang sama, dan melakukan perhitungan Ratcliff/Obershelp.

4.7.1 Deklarasi S_1 dan S_2

Langkah pertama pada algoritma Ratcliff/Obershelp yaitu mengidentifikasi dan deklarasi dari masukan data berupa S_1 sebagai koresponden data 1dan S_2 sebagai koresponden data 2. *Source Code* dari deklarasi S_1 dan S_2 ditampilkan pada Gambar 4.21.

```php
function panggil_ratcliff_obershelp(array $id_postingan){
    global $mysqli;
    global $sumbution;
    $junk = array();
    $options = array();
    $keluaran = array();
    for($k=1;$k<=2;$k++){
        for($i=1;$i<=3;$i++){
            $simpan_sementara = array();
            foreach($id_postingan as $post){
                if($k == 1){
                    $query    = "SELECT stemming, tipe from text_preprocessing_biasa WHERE
post_id='$post' and tipe='$i'";
                }else{
                    $query    = "SELECT stemming, tipe from text_preprocessing_nazief WHERE
post_id='$post' and tipe='$i'";
                }
                $results  = $mysqli->query($query);
                $isinya   = mysqli_fetch_assoc($results);
                if (mysqli_num_rows($results) >= 1) {
                    $array_stemming = json_decode($isinya['stemming'],true);
                    $simpan_sementara['data'][] = implode(" ",$array_stemming);
                    $simpan_sementara['panjang'][] = strlen(implode(" ",$array_stemming));
                    $simpan_sementara['tipe'] = $isinya['tipe'];
                    $simpan_sementara['text_preprocessing'] = $k;
                }else{
                    $simpan_sementara['data'][] = "";
                    $simpan_sementara['tipe'] = $i;
                    $simpan_sementara['text_preprocessing'] = $k;
                }
            }
        ...
    return $keluaran;
}
```

Gambar 4.21. *Source Code* PHP Algoritma Ratcliff/Obershelp Tahap Deklarasi S_1 dan S_2

Source Code pada Gambar 4.21 menunjukkan tentang pemanggilan data yang terdapat pada *text preprocessing* sebelumnya yang dibagi menjadi dua serta difilter oleh tipe pemanggilan data yaitu judul, deskripsi atau judul+deskripsi. Baris ke-dua belas dan baris ke-empat belas memanggil data yang nantinya pada baris ke-

dua puluh data akan disatukan dengan implode tanpa pemisal spasi, dan baris ke-dua puluh satu panjang dari karakter yang disatukan akan diketahui sehingga S_1 dan S_2 dapat diproses ke tahap selanjutnya.

4.7.2 Mendeteksi Banyaknya Karakter yang Sama pada Kedua String

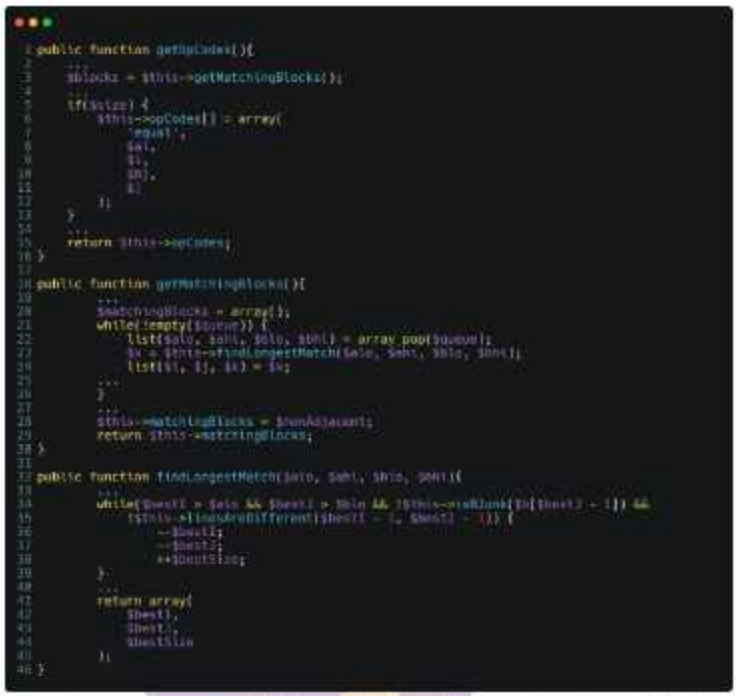
Berdasarkan S_1 dan S_2 yang telah diketahui selanjutnya mengidentifikasi jumlah karakter yang sama dengan mengidentifikasi anchor dan karakter yang sama. Berikut *Source Code* untuk mendeteksi karakter yang sama ditampilkan pada Gambar 4.22.

```
1  function panggil_ratcliff_obershelp(array $id_postingan){
2      ...
3      $Diff_SequenceMatcher = new Diff_SequenceMatcher($simpan_sementara["data"]
   [0],$simpan_sementara["data"][1],$junk,$options);
4      $penyimpan_block = $Diff_SequenceMatcher->getOpCodes();
5      $penyimpan_hasil = $Diff_SequenceMatcher->Ratio();
6      $penyimpan_hasil = str_replace('.', ',',round(($penyimpan_hasil), 4));
7      ...
8      $penyimpan_sama_deskripsi = array();
9      $penyimpan_sama = array();
10     for($i_block=0;$i_block<count($penyimpan_block);$i_block++){
11         if($penyimpan_block[$i_block][0] == 'equal'){
12             $batas_awal_1 = $penyimpan_block[$i_block][1];
13             $batas_akhir_1 = $penyimpan_block[$i_block][2];
14             $akhir = $batas_akhir_1-$batas_awal_1;
15
16             $batas_awal_2 = $penyimpan_block[$i_block][3];
17             $batas_akhir_2 = $penyimpan_block[$i_block][4];
18
19             $penyimpan_sama[] = substr($simpan_sementara["data"][0], $batas_awal_1,$akhir);
20             $penyimpan_sama_deskripsi[] = substr($simpan_sementara["data"][0],
   $batas_awal_1,$akhir) .' (Terdapat pada blok karakter $batas_awal_1 sampai $batas_akhir_1
   pada data 1 dan terdapat pada blok karakter $batas_awal_2 sampai $batas_akhir_2 pada data
   2)";
21         }
22     }
23     ...
24     return $keluaran;
25 }
```

Gambar 4.22. *Source Code* PHP Algoritma Ratcliff/Obershelp Tahap Mendeteksi Karakter yang Sama Bagian 1

Berdasarkan Gambar 4.22 dijelaskan pada baris ke-tiga dilakukan pemanggilan class dari Ratcliff/Obershelp dengan masukan S_1 dan S_2, pada baris ke-empat dilakukan pemanggilan identifikasi karakter yang sama dan pada bris ke-sembilan belas dan baris ke-dua puluh dibuatkan deskripsi letak blok karakter yang

sama, berikut penjelasan lebih detail pada baris ke-empat ditampilkan pada Gambar 4.23



```
1  public function getOpCodes(){
2      ...
3      $blocks = $this->getMatchingBlocks();
4      ...
5      if($size) {
6          $this->opCodes[] = array(
7              'equal',
8              $ai,
9              $i,
10             $bj,
11             $j
12         );
13     }
14     ...
15     return $this->opCodes;
16 }
17
18 public function getMatchingBlocks(){
19     ...
20         $matchingBlocks = array();
21         while(!empty($queue)) {
22             list($alo, $ahi, $blo, $bhi) = array_pop($queue);
23             $x = $this->findLongestMatch($alo, $ahi, $blo, $bhi);
24             list($i, $j, $k) = $x;
25         ...
26         }
27         ...
28         $this->matchingBlocks = $nonAdjacent;
29         return $this->matchingBlocks;
30 }
31
32 public function findLongestMatch($alo, $ahi, $blo, $bhi){
33     ...
34         while($besti > $alo && $bestj > $blo && !$this->isBJunk($b[$bestj - 1]) &&
35             !$this->isStringsDifferent($besti - 1, $bestj - 1)) {
36                 --$besti;
37                 --$bestj;
38                 ++$bestsize;
39         }
40         ...
41         return array(
42             $besti,
43             $bestj,
44             $bestsize
45         );
46 }
```

Gambar 4.23. *Source Code* PHP Algoritma Ratcliff/Obershelp Tahap Mendeteksi Karakter yang Sama Bagian 2

Berdasarkan alurnya dilakukan pemanggilan fungsi pada bagian 1 Gambar 4.22 dibaris ke-empat yang memanggil fungsi getOpCodes, pada Gambar 4.23 pada baris ke-tiga dalam fungsi getOpCodes dilakukan pemanggilan fungsi getMatchingBlocks dan pada baris ke-dua puluh tiga dilakukan pemanggilan fungsi findLongestMatch, pada fungsi tersebut mengidentifikasi karakter yang tidak berbeda atau memiliki karakter yang sama dengan panjang yang paling baik di jelaskan pada baris ke-tiga puluh empat dan sampai selesai. Sehingga kembalian

nilai pada baris ke-enam dapat diketahui nantinya tag yang tepat yaitu equal atau tidak pada blok yang diproses.

4.7.3 Perhitungan Algoritma Ratcliff/Obershelp

Setelah diketahui deklarasi S_1 dan S_2, serta jumlah karakter yang sama selanjutnya diproses pada perhitungan algoritma Ratcliff/Obershelp. Berikut *Source Code* untuk perhitungan algoritma Ratcliff/Obershelp ditampilkan pada Gambar 4.24.

```
1 public function Ratio(){
2     $matches = array_reduce($this->getMatchingBlocks(), array($this, 'ratioReduce'), 0);
3     return $this->calculateRatio($matches, count ($this->a) + count ($this->b));
4 }
```

Gambar 4.24. *Source Code* PHP Algoritma Ratcliff/Obershelp Tahap Perhitungan

Berdasarkan Gambar 4.24 dilakukan proses perhitungan algoritma Ratcliff/Obershelp yang dijelaskan pada baris ke-tiga yang membutuhkan masukkan data jumlah karakter yang sama, dan banyaknya karakter S_1 dan S_2. Sehingga pada tahap ini hasil *similarity* berdasarkan algoritma Ratcliff/Obershelp dapat diketahui.

4.8 Uji Coba dan Evaluasi

4.8.1 Lingkungan Uji Coba

Lingkungan uji coba dalam penelitian ini berupa perangkat keras dan perangkat lunak yang digunakan untuk melakukan uji coba penerapan algoritma *stemming* Nazief & Adriani terhadap perbandingan antara algoritma Winnowing dan Algoritma Ratcliff/Obershelp. Lingkungam uji coba ini berupa komputer dengan spesifikasi perangkat keras yang digunakan sebagai berikut :

Motherboard : MS-7A69

Processor : Intel(R) Core(TM) i5-7400 (4 CPUs), ~3.0GHz

Memory(RAM) : 8 GB

VGA Card : NVIDIA GeForce GTX 970

Perangkat lunak yang digunakan sebagai berikut :

Sistem Operasi : *Windows* 10 Education 64-bit

Pengembang : XAMPP

Bahasa Pemrograman : PHP, Javascript (Jquery)

Database : MySQL

Perangkat Pembantu : Bracket

4.8.2 Skenario Uji Coba

Terdapat beberapa skenario uji coba yang dilakukan diantaranya :

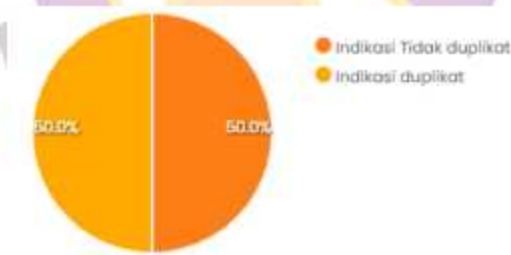
d. Perhitungan tingkat *similarity* dan waktu pemrosesan berdasarkan algoritma Winnowing dengan data berupa hasil dari *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani. Data masukkan nilai *k* pada K-Gram berupa nilai bilangan prima 2, 3, 5 dan 7. Menggunakan nilai *window* bilangan prima 2, serta menggunakan nilai basis bilangan prima 11. Data yang di uji menggunakan data berupa judul, deskripsi dan judul + deskripsi.

e. Perhitungan tingkat *similarity* dan waktu pemrosesan berdasarkan algoritma Ratcliff/Obershelp dengan data berupa hasil dari *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani. Data yang di uji menggunakan data berupa judul, deskripsi dan judul + deskripsi.

f. Perbandingan Hasil tingkat *similarity* dan waktu pemrosesan berdasarkan algoritma Winnowing dan algoritma Ratcliff/Obershelp dengan data berupa hasil dari *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani.

4.8.3 Pembagian Data

Pada penelitian ini menggunakan pemilihan sampel data berdasarkan rumus Wibisono dan didapatkan data sebanyak 100 produk yang akan dibagai menjadi 50% untuk data dengan indikasi duplikat dan 50% untuk data dengan indikasi tidak duplikat, berdasarkan indikasi awal tersebut data akan dibandingkan dengan pasangannya sehingga akan terbentuk 25 analisa setiap indikasinya. Berikut ditampilkan sebaran data berdasarkan indikasi pada Gambar 4.25.



Gambar 4.25. Sebaran Data Berdasarkan Indikasi

Berdasarkan Gambar 4.25 diketahui bahwa data memiliki sebaran yang seimbang dengan 50% untuk indikasi duplikat dan 50% untuk indikasi tidak duplikat.

4.8.4 Skenario 1 Perhitungan Tingkat *Similarity* dan Waktu Pemrosesan Berdasarkan Algoritma Winnowing

Pada skenario pengujian ini dilakukan perhitungan tingkat *similarity* dan waktu pemrosesan dari berbagai macam perbandingan seperti pemakaian variasi data yaitu judul, deskripsi dan judul+deskripsi. Pemakaian hasil *text preprocessing* berdasarkan dari *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani. Serta menggunakan nilai *k* pada K-Gram yang bervariasi yaitu 2, 3, 5 dan 7.

Berikut hasil dari rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Winnowing dari data dengan indikasi tidak duplikat pada *text preprocessing* tanpa *stemming* ditampilkan pada Tabel 4.1.

Tabel 4.1. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Algoritma Winnowing Berdasarkan *Text preprocessing* pada indikasi Tidak duplikat

Tipe	Presentase *Similarity*				Waktu pemrosesan			
	k = 2	k = 3	k = 5	k = 7	k = 2	k = 3	k = 5	k = 7
Judul	38,85%	28,57%	19,53%	13,01%	0,01 detik	0,01 detik	0,01 detik	0,01 detik
Deskripsi	51,89%	23,7%	10,55%	8,41%	0,15 detik	0,15 detik	0,15 detik	0,15 detik
Judul + Deskripsi	53,12%	24,61%	10,89%	8,55%	0,14 detik	0,15 detik	0,15 detik	0,15 detik

Berdasarkan Tabel 4.1 dengan indikasi tidak duplikat maka nilai hasil *similarity* terkecil merupakan nilai yang paling baik, berdasarkan penggunaan data dari *text preprocessing* tanpa *stemming* diketahui bahwa nilai $k = 7$ pada tipe data deskripsi dengan rata-rata nilai *similarity* 8,41%. Sedangkan untuk pemrosesan waktu dengan pemrosesan tercepat didapatkan pada pemrosesan data dengan tipe data judul dengan nilai $k = 2, 3, 5$ dan 7 memiliki rata-rata waktu pemrosesan tercepat yang sama yaitu 0,01 detik. Selanjutnya berdasarkan pemrosesan algoritma *stemming* Nazief & Adriani, berikut hasil dari rata-rata perhitungan *similarity* dan

waktu pemrosesan menggunakan algoritma Winnowing dari data dengan indikasi tidak duplikat pada *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani ditampilkan pada Tabel 4.2.

Tabel 4.2. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Algoritma Winnowing Berdasarkan *Text preprocessing* Menggunakan Algoritma *Stemming* Nazief & Adriani pada indikasi Tidak duplikat

Tipe	Presentase *Similarity*				Waktu pemrosesan			
	k = 2	k = 3	k = 5	k = 7	k = 2	k = 3	k = 5	k = 7
Judul	38,11%	28,41%	19,26%	12,57%	0,48 detik	0,48 detik	0,48 detik	0,48 detik
Deskripsi	49,84%	22,33%	10,21%	8,12%	9,60 detik	9,60 detik	9,60 detik	9,60 detik
Judul + Deskripsi	51,14%	23,26%	10,56%	8,26%	10,17 detik	10,17 detik	10,18 detik	10,18 detik

Terhadap pemrosesan data dengan indikasi tidak duplikat yang menggunakan data berdasarkan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani pada Tabel 4.2 ditemukan nilai rata-rata *similarity* terkecil dari pengujian berdasarkan nilai $k = 7$ pada tipe data deskripsi memiliki nilai rata-rata *similarity* terkecil yaitu 8,12%. Sedangkan untuk pemrosesan tercepat didapatkan pada pengujian menggunakan tipe data judul dengan nilai $k = 2, 3, 5$ dan 7 memiliki nilai kecepatan pemrosesan tercepat yang sama yaitu 0,48 detik. Selanjutnya dengan data indikasi duplikat, hasil rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Winnowing dari data dengan indikasi duplikat pada *text preprocessing* tanpa *stemming* ditampilkan pada Tabel 4.3.

Tabel 4.3. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Algoritma Winnowing Berdasarkan *Text preprocessing* pada indikasi duplikat

Tipe	Presentase *Similarity*				Waktu pemrosesan			
	k = 2	k = 3	k = 5	k = 7	k = 2	k = 3	k = 5	k = 7
Judul	78,07%	72,48%	68,64%	65,63%	0,01 detik	0,01 detik	0,01 detik	0,01 detik
Deskripsi	79,41%	60,87%	53,94%	49,65%	0,18 detik	0,18 detik	0,19 detik	0,19 detik
Judul + Deskripsi	79,89%	60,7%	53,3%	44,8%	0,18 detik	0,18 detik	0,18 detik	0,18 detik

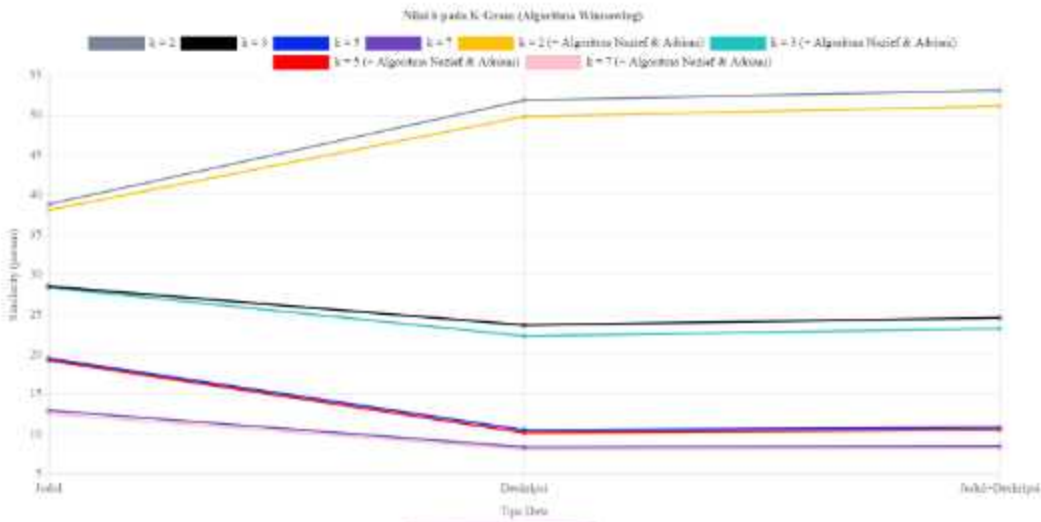
Berdasarkan Tabel 4.3 dengan indikasi duplikat maka nilai hasil *similarity* terbesar merupakan nilai yang paling baik, berdasarkan penggunaan data dari *text preprocessing* tanpa *stemming* diketahui bahwa nilai k = 2 pada tipe data judul + deskripsi dengan rata-rata nilai *similarity* 79,89%. Sedangkan untuk pemrosesan waktu dengan pemrosesan tercepat didapatkan pada pemrosesan data dengan tipe data judul dengan nilai k = 2, 3, 5 dan 7 memiliki rata-rata waktu pemrosesan tercepat yang sama yaitu 0,01 detik. Berdasarkan pemrosesan algoritma *stemming* Nazief & Adriani, berikut hasil dari rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Winnowing dari data dengan indikasi duplikat pada *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani ditampilkan pada Tabel 4.4.

Tabel 4.4. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Algoritma Winnowing Berdasarkan *Text preprocessing* Menggunakan Algoritma *Stemming* Nazief & Adriani pada indikasi duplikat

Tipe	Presentase *Similarity*				Waktu pemrosesan			
	k = 2	k = 3	k = 5	k = 7	k = 2	k = 3	k = 5	k = 7
Judul	77,82%	72,32%	68,39%	65,29%	0,45 detik	0,45 detik	0,45 detik	0,45 detik
Deskripsi	79,11%	59,9%	53,55%	51,23%	12,11 detik	12,11 detik	12,11 detik	12,11 detik
Judul + Deskripsi	79,51%	59,61%	52,89%	48,06%	12,53 detik	12,53 detik	12,54 detik	12,54 detik

Pemrosesan data dengan indikasi duplikat yang menggunakan data berdasarkan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani pada Tabel 4.4 ditemukan nilai rata-rata *similarity* terbesar dari pengujian berdasarkan nilai k = 2 pada tipe data judul+deskripsi dengan memiliki nilai rata-rata *similarity* terbesar yaitu 79,51%. Sedangkan untuk pemrosesan tercepat didapatkan pada pengujian menggunakan tipe data judul dengan nilai k = 2, 3, 5 dan 7 memiliki nilai kecepatan pemrosesan tercepat yang sama yaitu 0,45 detik.

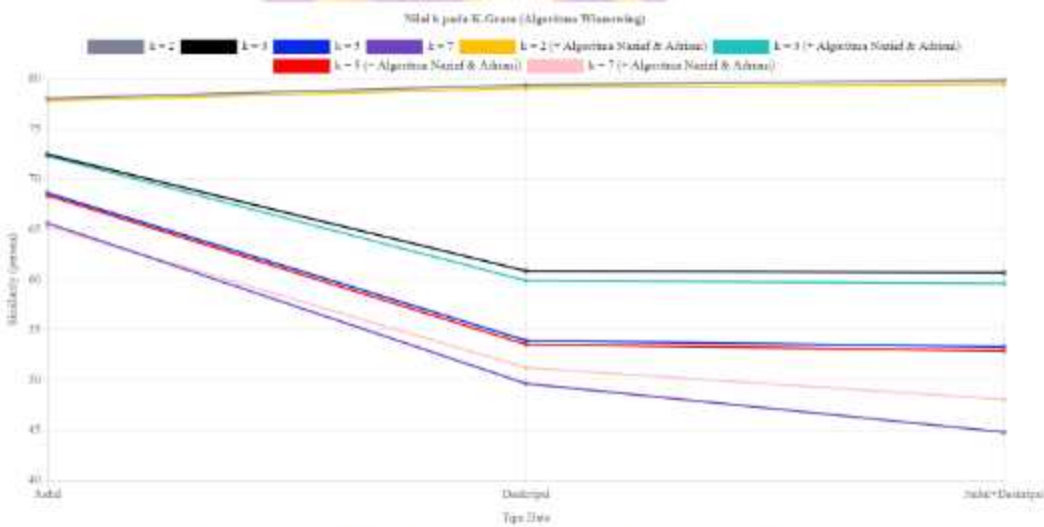
Berdasarkan sebaran hasil pengujian dapat divisualisasikan pada grafik untuk mengetahui perbandingan antara hasil rata-rata nilai *similarity* dan rata-rata waktu pemrosesan. Berikut grafik rata-rata hasil *similarity* menggunakan algoritma Winnowing dengan perbandingan antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani pada Indikasi produk tidak duplikat ditampilkan pada Gambar 4.26.



Gambar 4.26. Grafik Perbandingan Rata-Rata Hasil *Similarity* Menggunakan Algoritma Winnowing pada Indikasi Produk Tidak Duplikat

Grafik yang ditampilkan pada Gambar 4.26 dan hasil yang ditampilkan pada Tabel 4.1 dan Tabel 4.2 dengan data tidak duplikat maka nilai rata-rata *similarity* terkecil merupakan nilai yang paling baik. Perbandingan antara data yang diproses menggunakan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan *stemming* Nazief & Adriani mendapatkan hasil bahwa nilai rata-rata *similarity* terkecil didapatkan oleh data dari *text preprocessing* menggunakan

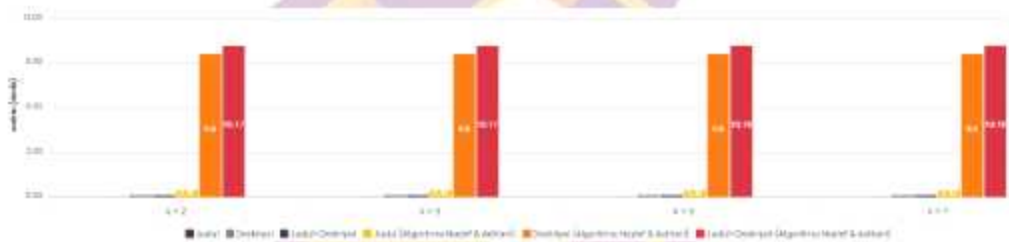
stemming Nazief & Adriani pada nilai $k = 7$ dengan tipe data deskripsi dengan nilai rata-rata *similarity* yaitu 8,12%. Nilai rata-rata *similarity* tersebut terjadi penurunan sebesar 0,29% dari nilai 8,41% jika tidak menggunakan *stemming* Nazief & Adriani menjadi 8,12 jika menggunakan *stemming* Nazief & Adriani. Berdasarkan nilai tersebut karena didapatkan hasil dibawah kurang dari 30% maka termasuk plagiarisme ringan. Selanjutnya berikut sebaran grafik rata-rata hasil *similarity* menggunakan algoritma Winnowing dengan perbandingan antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani pada Indikasi produk duplikat ditampilkan pada Gambar 4.27.



Gambar 4.27. Grafik Rata-Rata Hasil *Similarity* Menggunakan Algoritma Winnowing pada Indikasi Produk Duplikat

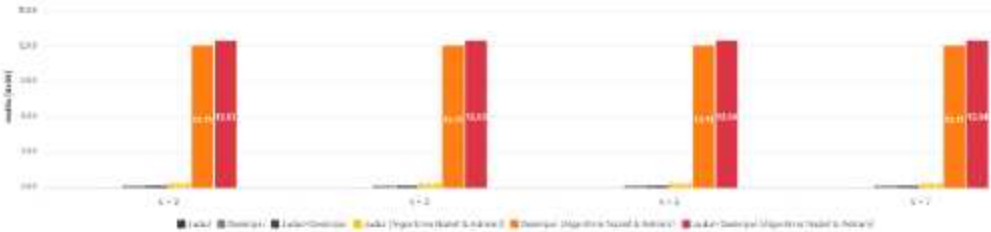
Berdasarkan indikasi produk duplikat, Gambar 4.27 menampilkan bahwa nilai rata-rata *similarity* dari Tabel 4.3 dan Tabel 4.4 dengan nilai terbesar yaitu didapatkan pada data yang menggunakan *text preprocessing* tanpa *stemming* pada

nilai k = 2 dengan tipe data judul + deskripsi dengan nilai rata-rata *similarity* 79,89%. Nilai tersebut lebih besar daripada ketika pada nilai k = 2 *stemming* Nazief & Adriani yaitu mendapatkan nilai 79,51% karena nilai tersebut mengalami penurunan sebesar 0,38%. Nilai tersebut termasuk plagiarisme berat atau total karena nilainya diatas 70%. Setelah divisualisasikan grafik rata-rata hasil *similarity* selanjutnya ditampilkan grafik rata-rata waktu pemrosesan menggunakan algoritma Winnowing dengan perbandingan antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani yang ditampilkan pada Gambar 4.28.



Gambar 4.28. Grafik Perbandingan Waktu Pemrosesan Menggunakan Algoritma Winnowing pada Indikasi Produk Tidak Duplikat

Grafik perbandingan kecepatan pemrosesan waktu pada Gambar 4.28 didapatkan nilai tercepat yaitu pada pemrosesan menggunakan *text preprocessing* tanpa *stemming* pada nilai k = 2, 3, 5 dan 7 pada tipe data judul yang memiliki rata-rata waktu pemrosesan yang sama yaitu 0,01 detik. Berikut sebaran grafik waktu pemrosesan menggunakan algoritma Winnowing dengan perbandingan antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani pada Indikasi produk duplikat ditampilkan pada Gambar 4.29.



Gambar 4.29. Grafik Perbandingan Waktu Pemrosesan Menggunakan Algoritma Winnowing pada Indikasi Produk Duplikat

Berdasarkan Gambar 4.29 pada perbandingan hasil rata-rata waktu pemrosesan dengan menggunakan data berdasarkan pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani didapatkan hasil nilai rata-rata waktu pemrosesan tercepat yaitu pada data yang berasal dari pemrosesan *text preprocessing* tanpa *stemming* pada $k = 2$, 3, 5 dan 7 pada tipe data judul dengan nilai rata-rata waktu pemrosesan yang sama yaitu 0,01 detik.

4.8.5 Skenario 2 Perhitungan Tingkat *Similarity* dan Waktu Pemrosesan Berdasarkan Algoritma Ratcliff/Obershelp

Pada skenario pengujian ini dilakukan perhitungan tingkat *similarity* dan waktu pemrosesan dari berbagai macam perbandingan seperti pemakaian variasi data yaitu judul, deskripsi dan judul+deskripsi. Pemakaian hasil *text preprocessing* berdasarkan dari *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani.

Berikut hasil dari rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Ratcliff/Obershelp dari data dengan indikasi tidak duplikat pada *text preprocessing* tanpa *stemming* ditampilkan pada Tabel 4.5.

Tabel 4.5. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Algoritma Ratcliff/Obershelp Berdasarkan *Text preprocessing* pada indikasi Tidak duplikat

Tipe	Persentase *Similarity*	Waktu Pemrosesan
Judul	54,32%	0,01 detik
Deskripsi	9,86%	0,14 detik
Judul + Deskripsi	10,98%	0,14 detik

Tabel 4.5 menjelaskan bahwa pada data indikasi tidak duplikat dengan data berdasarkan pemrosesan *text preprocessing* tanpa *stemming* mendapatkan hasil nilai rata-rata *similarity* dengan nilai terkecil yang didapatkan pada tipe data deskripsi dengan nilai rata-rata *similarity* 9,86%. Sedangkan untuk rata-rata waktu pemrosesan yang memiliki waktu pemrosesan tercepat yaitu pada pemrosesan data pada tipe judul dengan rata-rata waktu pemrosesan 0,01 detik. Selanjtunya berdasarkan pemrosesan algoritma *stemming* Nazief & Adriani, berikut hasil dari rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Ratcliff/Obershelp dari data dengan indikasi tidak duplikat pada *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani ditampilkan pada Tabel 4.6.

Tabel 4.6. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Algoritma Ratcliff/Obershelp Berdasarkan *Text preprocessing* Menggunakan Algoritma *Stemming* Nazief & Adriani pada indikasi Tidak duplikat

Tipe	Persentase *Similarity*	Waktu Pemrosesan
Judul	54,09%	0,48 detik
Deskripsi	9,69%	9,59 detik
Judul + Deskripsi	11,17%	10,16 detik

Berdasarkan Tabel 4.6 didapatkan hasil bahwa nilai rata-rata *similarity* terkecil pada data indikasi tidak duplikat yang menggunakan data berdasarkan pemrosesan dari *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani mendapatkan bahwa pada data tipe deskripsi memiliki nilai rata-rata terkecil yaitu 9,69%. Sedangkan untuk nilai rata-rata waktu pemrosesan yang memiliki nilai tercepat yaitu pada tipe data judul dengan nilai rata-rata waktu

pemrosesan 0,48 detik. Selanjutnya dengan data indikasi duplikat, berikut hasil dari rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Ratcliff/Obershelp dari data dengan indikasi duplikat pada *text preprocessing* tanpa *stemming* ditampilkan pada Tabel 4.7.

Tabel 4.7. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Algoritma Ratcliff/Obershelp Berdasarkan *Text preprocessing* pada indikasi duplikat

Tipe	Persentase *Similarity*	Waktu Pemrosesan
Judul	86,91%	0,01 detik
Deskripsi	48,17%	0,17 detik
Judul + Deskripsi	44,89%	0,17 detik

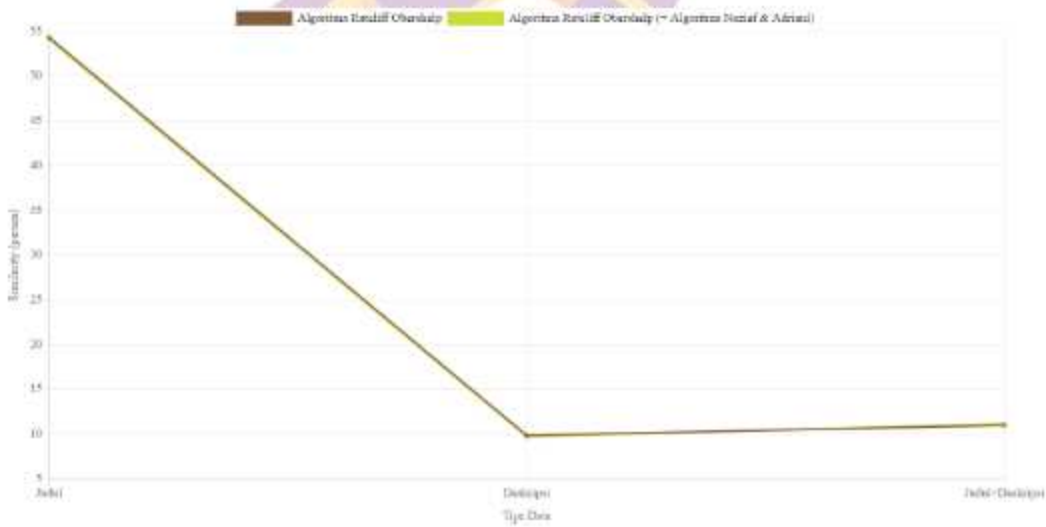
Terhadap pemrosesan data indikasi duplikat yang ditampilkan pada Tabel 4.7 didapatkan bahwa nilai rata-rata *similarity* terbesar dengan data berdasarkan dari pemrosesan *text preprocessing* tanpa *stemming* didapatkan nilai rata-rata terbesar pada tipe data judul dengan nilai 86,91%. Sedangkan untuk nilai rata-rata waktu pemrosesan tercepat didapatkan pada pemrosesan data dengan tipe judul yang memiliki nilai rata-rata waktu pemrosesan 0,01 detik. Berdasarkan pemrosesan algoritma *stemming* Nazief & Adriani, berikut hasil dari rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Ratcliff/Obershelp dari data dengan indikasi duplikat pada *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani ditampilkan pada Tabel 4.8.

Tabel 4.8. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Algoritma Ratcliff/Obershelp Berdasarkan *Text preprocessing* Menggunakan Algoritma *Stemming* Nazief & Adriani pada indikasi duplikat

Tipe	Persentase *Similarity*	Waktu Pemrosesan
Judul	86,87%	0,45 detik
Deskripsi	47,29%	12,10 detik
Judul + Deskripsi	41,08%	12,52 detik

Pemrosesan data pada indikasi duplikat dengan data berdasarkan dari pemrosesan *text preprocessing* dengan menggunakan algoritma *stemming* Nazief &

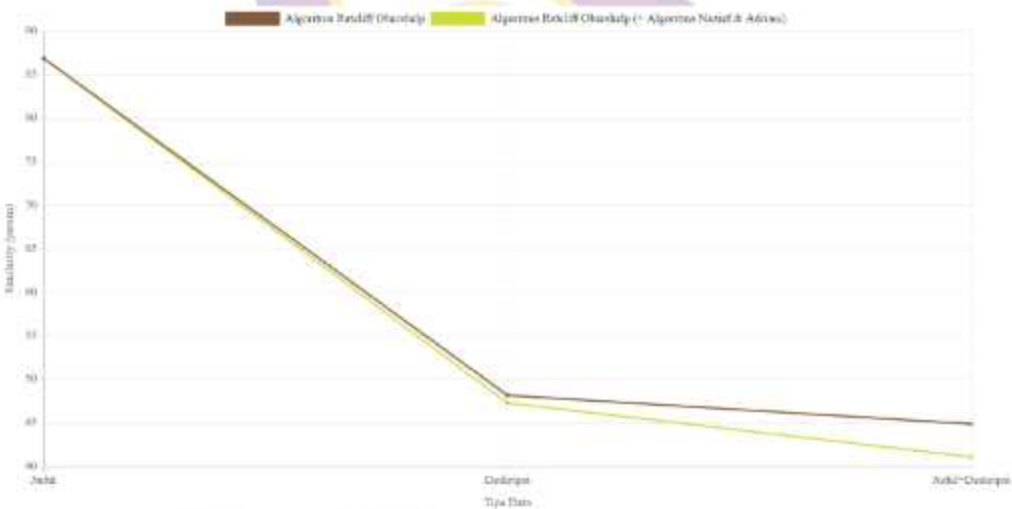
Adriani didapatkan hasil nilai rata-rata *similarity* terbesar yaitu pada tipe data judul dengan nilai rata-rata *similarity* 86,87%. Sedangkan untuk kecepatan waktu pemrosesan didapatkan nilai tercepat pada tipe data judul yaitu 0,45 detik. Berdasarkan sebaran hasil pengujian dapat divisualisasikan pada grafik untuk mengetahui perbandingan antara hasil rata-rata nilai *similarity* dan rata-rata waktu pemrosesan. Berikut grafik rata-rata hasil *similarity* menggunakan algoritma Ratcliff/Obershelp dengan perbandingan antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani pada Indikasi produk tidak duplikat ditampilkan pada Gambar 4.30.



Gambar 4.30. Grafik Perbandingan Rata-Rata Hasil *Similarity* Menggunakan Algoritma Ratcliff/Obershelp pada Indikasi Produk Tidak Duplikat

Grafik pada Gambar 4.30 menampilkan bahwa perbandingan antara data berdasarkan Tabel 4.5 dan Tabel 4.6 pada pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief &

Adriani mendapatkan hasil nilai rata-rata *similarity* dengan nilai terkecil pada indikasi data tidak duplikat diperoleh oleh data dengan tipe deskripsi yang berasal dari *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani dengan nilai rata-rata *similarity* 9,69%. Nilai tersebut lebih kecil sebesar 0,17% dibandingkan nilai rata-rata *similarity* tanpa *stemming* yaitu sebesar 9,86%. Sehingga nilai 9,69% tersebut termasuk dalam proposi plagiarisme ringan karena dibawah 30%. Berikut sebaran grafik rata-rata hasil *similarity* menggunakan algoritma Ratcliff/Obershelp dengan perbandingan antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani pada Indikasi produk duplikat ditampilkan pada Gambar 4.31.



Gambar 4.31. Grafik Rata-Rata Hasil *Similarity* Menggunakan Algoritma Ratcliff/Obershelp pada Indikasi Produk Duplikat

Berdasarkan Gambar 4.31 grafik perbandingan pada data indikasi duplikat dengan perbandingan data yang berdasarkan Tabel 4.7 dan Tabel 4.8 pada

pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani mendapatkan hasil nilai rata-rata *similarity* terbesar yaitu pada data yang berasal dari *text preprocessing* tanpa *stemming* pada tipe data judul dengan nilai rata-rata *similarity* 86,91%. Nilai tersebut lebih besar daripada nilai rata-rata *similarity* yang menggunakan *stemming* Nazief & Adriani yaitu dengan nilai 86,87%. Berdasarkan nilai 86,91% tersebut diketahui bahwa termasuk plagiarisme berat atau total karena nilainya lebih dari 70%. Setelah divisualisasikan grafik rata-rata hasil *similarity* selanjutnya ditampilkan grafik rata-rata waktu pemrosesan menggunakan algoritma Ratcliff/Obershelp dengan perbandingan antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani yang ditampilkan pada Gambar 4.32.

Gambar 4.32. Grafik Perbandingan Waktu Pemrosesan Menggunakan Algoritma Ratcliff/Obershelp pada Indikasi Produk Tidak Duplikat

Terhadap grafik perbandingan yang ditampilkan pada Gambar 4.32 untuk waktu pemrosesan didapatkan hasil nilai rata-rata waktu pemrosesan tercepat pada data indikasi tidak duplikat yaitu pada tipe data judul dengan data yang berasal dari pemrosesan *text preprocessing* tanpa *stemming* dengan rata-rata waktu pemrosesan 0,01 detik. Berikut sebaran grafik waktu pemrosesan menggunakan algoritma

Ratcliff/Obershelp dengan perbandingan antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani pada Indikasi produk duplikat ditampilkan pada Gambar 4.33.

Gambar 4.33. Grafik Perbandingan Waktu Pemrosesan Menggunakan Algoritma Ratcliff/Obershelp pada Indikasi Produk Duplikat

Berdasarkan Gambar 4.33 pada grafik perbandingan kecepatan waktu pemrosesan pada data indikasi duplikat dengan data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani didapatkan hasil nilai rata-rata kecepatan waktu pemrosesan tercepat pada tipe data judul yang berasal dari pemrosesan *text preprocessing* tanpa *stemming* dengan nilai rata-rata 0,01 detik.

4.8.6 Skenario 3 Perbandingan Hasil Tingkat *Similarity* dan Waktu Pemrosesan Berdasarkan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp

Pada skenario pengujian ini dilakukan perbandingan dari hasil tingkat *similarity* dan waktu pemrosesan dari berbagai macam perbandingan seperti pemakaian variasi data yaitu judul, deskripsi dan judul+deskripsi. Pemakaian hasil *text preprocessing* berdasarkan dari *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani.

Berikut perbandingan hasil dari rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp dari data dengan indikasi tidak duplikat pada *text preprocessing* tanpa *stemming* ditampilkan pada Tabel 4.9.

Tabel 4.9. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Berdasarkan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp Berdasarkan *Text preprocessing* pada indikasi Tidak duplikat

Tipe	Presentase *Similarity* (Algoritma Winnowing)				Waktu pemrosesan (Algoritma Winnowing)				Presentase *Similarity* (Algoritma Ratcliff/Obershelp)	Waktu pemrosesan (Algoritma Ratcliff/Obershelp)
	k = 2	k = 3	k = 5	k = 7	k = 2	k = 3	k = 5	k = 7		
Judul	38,85%	28,57%	19,53%	13,01%	0,01 detik	0,01 detik	0,01 detik	0,01 detik	54,32%	0,01 detik
Deskripsi	51,89%	23,7%	10,55%	8,41%	0,15 detik	0,15 detik	0,15 detik	0,15 detik	9,86%	0,14 detik
Judul+Deskripsi	53,12%	24,61%	10,89%	8,55%	0,14 detik	0,15 detik	0,15 detik	0,15 detik	10,98%	0,14 detik

Terdapat hasil perbandingan dari rata-rata nilai *similarity* terkecil pada data indikasi tidak duplikat dengan data data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* yaitu didapatkan oleh pemrosesan dari algoritma Winnowing dengan nilai $k = 7$ dengan tipe data deskripsi yang memiliki nilai rata-rata *similarity* yaitu 8,4%. Sedangkan untuk waktu pemrosesan tercepat didapatkan pada algoritma Winnowing dengan nilai k = 2, 3, 5 dan 7 serta algoritma Ratcliff/Obershelp pada tipe data judul dengan rata-rata nilai yang sama yaitu 0,01 detik. Berdasarkan pemrosesan algoritma *stemming* Nazief & Adriani, berikut hasil dari rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp dari data dengan indikasi tidak duplikat pada *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani ditampilkan pada Tabel 4.10.

Tabel 4.10. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Berdasarkan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp Berdasarkan *Text preprocessing* Menggunakan Algoritma *Stemming* Nazief & Adriani pada indikasi Tidak duplikat

Tipe	Presentase *Similarity* (Algoritma Winnowing)				Waktu pemrosesan (Algoritma Winnowing)				Presentase *Similarity* (Algoritma Ratcliff/Obershelp)	Waktu pemrosesan (Algoritma Ratcliff/Obershelp)
	k = 2	k = 3	k = 5	k = 7	k = 2	k = 3	k = 5	k = 7		
Judul	38,11%	28,41%	19,26%	12,57%	0,48 detik	0,48 detik	0,48 detik	0,48 detik	54,09%	0,48 detik
Deskripsi	49,84%	22,33%	10,21%	8,12%	9,60 detik	9,60 detik	9,60 detik	9,60 detik	9,69%	9,59 detik
Judul+Deskripsi	51,14%	23,26%	10,56%	8,26%	10,17 detik	10,17 detik	10,18 detik	10,18 detik	11,17%	10,16 detik

Berdasarkan Tabel 4.10 didapatkan hasil nilai *similarity* terkecil pada data indikasi tidak duplikat yang berasal dari *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani mendapatkan hasil bahwa pemrosesan berdasarkan algoritma Winnowing memiliki nilai terkecil pada $k = 7$ dengan tipe data deskripsi yang memiliki nilai 8,12%. Sedangkan untuk waktu pemrosesan tercepat didapatkan pada algoritma Winnowing dengan nilai $k = 2, 3, 5$ dan 7 serta algoritma Ratcliff/Obershelp pada tipe data judul dengan rata-rata nilai yang sama yaitu 0,48 detik pada skenario yang telah dilakukan. Selanjutnya dengan data indikasi duplikat, berikut hasil dari rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp dari data dengan indikasi duplikat pada *text preprocessing* tanpa *stemming* ditampilkan pada Tabel 4.11.

Tabel 4.11. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Berdasarkan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp Berdasarkan *Text preprocessing* pada indikasi duplikat

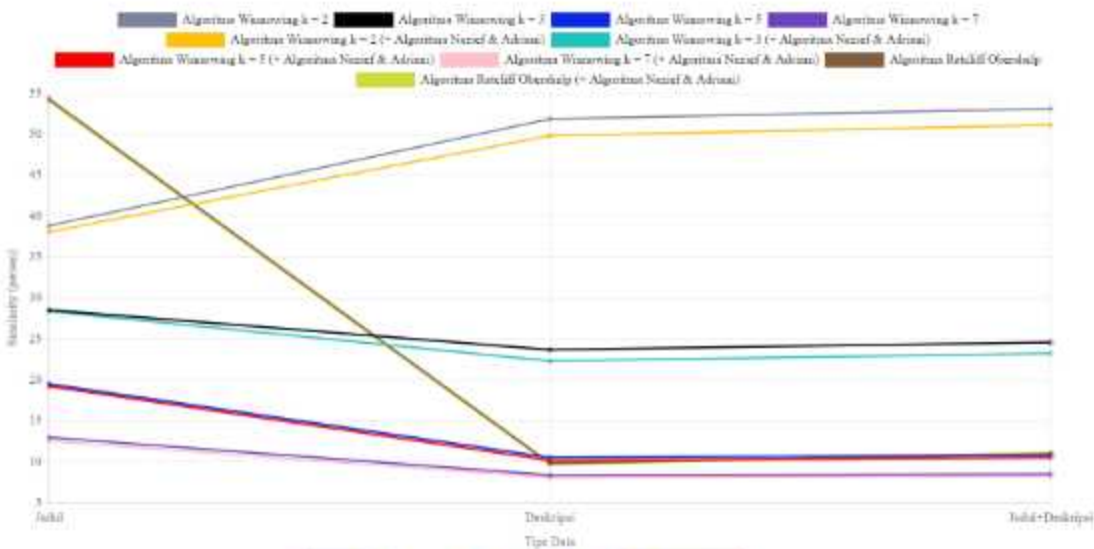
Tipe	Presentase *Similarity* (Algoritma Winnowing)				Waktu pemrosesan (Algoritma Winnowing)				Presentase *Similarity* (Algoritma Ratcliff/Obershelp)	Waktu pemrosesan (Algoritma Ratcliff/Obershelp)
	k = 2	k = 3	k = 5	k = 7	k = 2	k = 3	k = 5	k = 7		
Judul	78,07%	72,48%	68,64%	65,63%	0,01 detik	0,01 detik	0,01 detik	0,01 detik	86,91%	0,01 detik
Deskripsi	79,41%	60,87%	53,94%	49,65%	0,18 detik	0,18 detik	0,19 detik	0,19 detik	48,17%	0,17 detik
Judul+Deskripsi	79,89%	60,7%	53,3%	44,8%	0,18 detik	0,18 detik	0,18 detik	0,18 detik	44,89%	0,17 detik

Tabel 4.11 mendapatkan hasil nilai rata-rata *similarity* dengan nilai terbesar yang menggunakan data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* didapatkan oleh pemrosesan algoritma algoritma Ratcliff/Obershelp pada tipe data judul dengan nilai rata-rata *similarity* 86,91%. Sedangkan untuk waktu pemrosesan tercepat didapatkan pada algoritma Winnowing dengan nilai k = 2, 3, 5 dan 7 serta algoritma Ratcliff/Obershelp pada tipe data judul dengan rata-rata nilai yang sama yaitu 0,01 detik. Berdasarkan pemrosesan algoritma *stemming* Nazief & Adriani, berikut hasil dari rata-rata perhitungan *similarity* dan waktu pemrosesan menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp dari data dengan indikasi duplikat pada *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani ditampilkan pada Tabel 4.12.

Tabel 4.12. Rata-Rata Hasil *Similarity* dan Waktu Pemrosesan Berdasarkan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp Berdasarkan *Text preprocessing* Menggunakan Algoritma *Stemming* Nazief & Adriani pada indikasi duplikat

Tipe	Presentase *Similarity* (Algoritma Winnowing)				Waktu pemrosesan (Algoritma Winnowing)				Presentase *Similarity* (Algoritma Ratcliff/Obershelp)	Waktu pemrosesan (Algoritma Ratcliff/Obershelp)
	k = 2	k = 3	k = 5	k = 7	k = 2	k = 3	k = 5	k = 7		
Judul	77,82%	72,32%	68,39%	65,29%	0,45 detik	0,45 detik	0,45 detik	0,45 detik	86,87%	0,45 detik
Deskripsi	79,11%	59,9%	53,55%	51,23%	12,11 detik	12,11 detik	12,11 detik	12,11 detik	47,29%	12,10 detik
Judul+Deskripsi	79,51%	59,61%	52,89%	48,06%	12,53 detik	12,53 detik	12,54 detik	12,54 detik	41,08%	12,52 detik

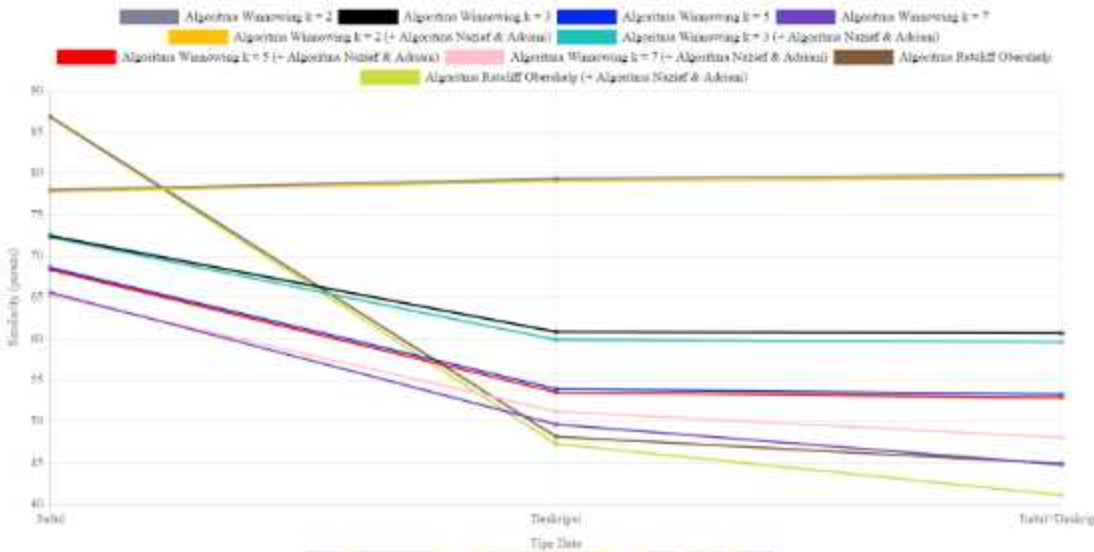
Perbandingan rata-rata hasil *similarity* pada data indikasi duplikat yang berdasarkan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani mendapatkan hasil nilai terbesar yaitu pada pemrosesan algoritma Ratcliff/Obershelp pada tipe data judul dengan nilai rata-rata *similarity* 86,87%. Sedangkan untuk waktu pemrosesan tercepat didapatkan pada algoritma Winnowing dengan nilai k = 2, 3, 5 dan 7 serta algoritma Ratcliff/Obershelp pada tipe data judul dengan rata-rata nilai yang sama yaitu 0,45 detik. Berdasarkan sebaran hasil pengujian dapat divisualisasikan pada grafik untuk mengetahui perbandingan antara hasil rata-rata nilai *similarity* dan rata-rata waktu pemrosesan. Berikut grafik rata-rata hasil *similarity* menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp dengan perbandingan antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani pada Indikasi produk tidak duplikat ditampilkan pada Gambar 4.34.



Gambar 4.34. Grafik Perbandingan Rata-Rata Hasil *Similarity* Berdasarkan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Indikasi Produk Tidak Duplikat

Grafik perbandingan rata-rata hasil *similarity* pada Gambar 4.34 dengan data indikasi tidak duplikat serta data yang berdasarkan Tabel 4.9 dan Tabel 4.10 pada pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani memiliki hasil rata-rata nilai terkecil yaitu pada pemrosesan algoritma Winnowing dengan nilai $k = 7$ pada data tipe deskripsi yang berdasarkan pemrosesan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani dengan nilai rata-rata *similarity* 8,12% sehingga mendapatkan proporsi plagiarisme ringan. Selanjutnya berikut sebaran grafik rata-rata hasil *similarity* menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp dengan perbandingan antara *text preprocessing* tanpa *stemming*

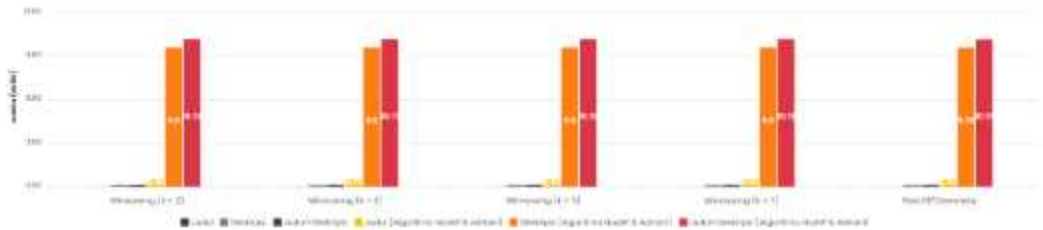
dan *text preprocessing* dengan algoritma Nazief & Adriani pada Indikasi produk duplikat ditampilkan pada Gambar 4.35.



Gambar 4.35. Grafik Rata-Rata Hasil *Similarity* Berdasarkan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Indikasi Produk Duplikat

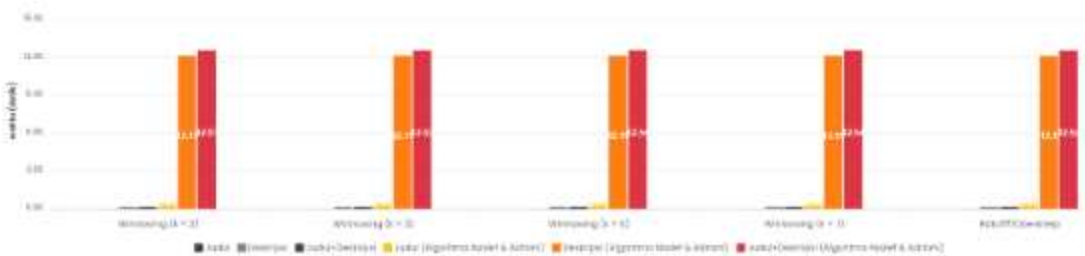
Berdasarkan Gambar 4.35 pada perbandingan rata-rata hasil *similarity* pada indikasi duplikat dengan data yang berdasarkan Tabel 4.11 dan Tabel 4.12 pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani didapatkan nilai rata-rata *similarity* dengan nilai terbesar yaitu pada pemrosesan algoritma Ratcliff/Obershelp pada tipe data judul dengan nilai rata-rata *similarity* 86,91% sehingga mendapatkan proporsi plagiarisme berat atau total. Setelah divisualisasikan grafik rata-rata hasil *similarity* selanjutnya ditampilkan grafik rata-rata waktu pemrosesan menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp dengan perbandingan

antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani yang ditampilkan pada Gambar 4.36.



Gambar 4.36. Grafik Perbandingan Waktu Pemrosesan Berdasarkan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Indikasi Produk Tidak Duplikat

Perbandingan waktu pemrosesan Gambar 4.36 pada produk tidak duplikat memperoleh hasil pemrosesan tercepat yaitu pada pemrosesan algoritma Winnowing dengan nilai k = 2, 3, 5 dan 7 serta algoritma Ratcliff/Obershelp pada tipe data judul dan berdasarkan pemrosesan dari *text preprocessing* tanpa *stemming* yaitu dengan rata-rata waktu pemrosesan yang sama yaitu 0,01 detik. Berikut sebaran grafik waktu pemrosesan menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp dengan perbandingan antara *text preprocessing* tanpa *stemming* dan *text preprocessing* dengan algoritma Nazief & Adriani pada Indikasi produk duplikat ditampilkan pada Gambar 4.37.



Gambar 4.37. Grafik Perbandingan Waktu Pemrosesan Berdasarkan Algoritma Winnowing dan Algoritma Ratcliff/Obershelp pada Indikasi Produk Duplikat

Berdasarkan perbandingan waktu pemrosesan pada Gambar 4.37 pada produk duplikat memperoleh hasil pemrosesan tercepat yaitu pada pemrosesan algoritma Winnowing dengan nilai k = 2, 3, 5 dan 7 serta algoritma Ratcliff/Obershelp pada tipe data judul dan berdasarkan pemrosesan dari *text preprocessing* tanpa *stemming* yaitu dengan rata-rata waktu pemrosesan yang sama yaitu 0,01 detik.

4.8.7 Evaluasi Skenario 1

Berdasarkan skenario 1 yaitu menggunakan algoritma Winnowing sebagai pendeteksi *similarity* yang menggunakan data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani. Didapatkan hasil pada rata-rata berdasarkan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani pada indikasi tidak duplikat Tabel 4.1 dan Tabel 4.2 bahwa penerapan nilai k yang bervariasi dari nilai k = 2, 3, 5 dan 7 menghasilkan pengaruh terhadap perbedaan hasil rata-rata *similarity* yaitu jika nilai k semakin besar maka *similarity* semakin, pengaruh tersebut sama dengan hasil pada data indikasi duplikat pada Tabel 4.3 dan Tabel 4.4.

Perbandingan penerapan pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani pada data indikasi tidak duplikat pada Gambar 4.26 dengan hasil dari Tabel 4.1 dan Tabel 4.2 didapatkan hasil bahwa penerapan algoritma *stemming* Nazief & Adriani cenderung menurunkan hasil *similarity* pada setiap pengujian menggunakan nilai nilai k = 2, 3, 5 dan 7. Berdasarkan tipe data judul menurunkan hasil rata-rata nilai

similarity pada nilai $k = 2$ sebesar 0,74%, $k = 3$ sebesar 0,16%, $k = 5$ sebesar 0,27% dan $k = 7$ sebesar 0,44%. Sedangkan untuk data tipe deskripsi menurunkan hasil rata-rata nilai *similarity* pada nilai $k = 2$ sebesar 2,05%, $k = 3$ sebesar 1,37%, $k = 5$ sebesar 0,34% dan $k = 7$ sebesar 0,29%. Serta pada tipe data judul + deskripsi menurunkan hasil rata-rata nilai *similarity* pada nilai $k = 2$ sebesar 1,98%, $k = 3$ sebesar 1,35%, $k = 5$ sebesar 0,33% dan $k = 7$ sebesar 0,29%. Sehingga hasil dari data yang berdasarkan pemrosesan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani memiliki hasil rata-rata dibawah rata-rata hasil *similarity* berdasarkan data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* untuk indikasi produk tidak duplikat.

Berdasarkan indikasi data duplikat pada Gambar 4.27 penerapan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani juga membuat hasil nilai rata-rata *similarity* menurun pada setiap nilai *k* 2, 3, 5 dan sebagian 7. Berdasarkan tipe data judul menurunkan hasil rata-rata nilai *similarity* pada nilai $k = 2$ sebesar 0,25%, $k = 3$ sebesar 0,16%, $k = 5$ sebesar 0,25% dan $k = 7$ sebesar 0,34%. Sedangkan untuk data tipe deskripsi menurunkan hasil rata-rata nilai *similarity* pada $k = 2$ sebesar 0,30%, $k = 3$ sebesar 0,97%, $k = 5$ sebesar 0,39%. Serta pada tipe data judul + deskripsi menurunkan hasil rata-rata nilai *similarity* pada nilai $k = 2$ sebesar 0,38%, $k = 3$ sebesar 1,09%, $k = 5$ sebesar 0,41%. Sedangkan untuk nilai $k = 7$ menambah nilai *similarity* pada tipe data deskripsi dan judul+deskripsi, yaitu pada tipe data deskripsi sebesar 6,43% sedangkan untuk tipe data judul+deskripsi sebesar 3,26%. Sehingga hasil dari data yang berdasarkan pemrosesan *text preprocessing* menggunakan algoritma *stemming* Nazief &

Adriani memiliki sebagian besar hasil rata-rata dibawah rata-rata hasil *similarity* berdasarkan data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* untuk indikasi produk duplikat terkecuali pada nilai k = 7 pada tipe deskripsi dan judul+deskripsi yang mengalami kenaikan nilai rata-rata *similarity*.

Berdasarkan perbandingan waktu pemrosesan pada Gambar 4.28 dan 4.29 menampilkan bahwa penerapan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani secara penuh pada nilai k 2, 3, 5 dan 7 serta pada tipe data judul, deskripsi dan judul+deskripsi mengalami kenaikan waktu pemrosesan. Berdasarkan Gambar 4.28 kenaikan pemrosesan waktu didapatkan selisih kenaikan pada tipe data judul dengan nilai k = 2, 3, 5 dan 7 sebesar 0,47 detik, sedangkan pada tipe data deskripsi dengan nilai k = 2, 3, 5 dan 7 sebesar 0,81 detik, sedangkan pada tipe data judul+deskripsi dengan nilai k = 2 sebesar 10,03 detik, pada nilai k = 3 sebesar 10,02 detik, pada nilai k = 5 dan 7 sebesar 10.03 detik. Berdasarkan Gambar 4.29 pada indikasi produk duplikat didapatkan kenaikan pemrosesan waktu yaitu pada tipe data judul dengan nilai k = 2, 3, 5 dan 7 sebesar 0,44 detik, sedangkan pada tipe data deskripsi dengan nilai k = 2 dan 3 sebesar 12,35 detik, untuk nilai k = 5 dan 7 sebesar 12,36 detik. Sehingga pengaruh penerapan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani secara menyeluruh pada setiap nilai k menambah waktu pemrosesan baik itu pada data indikasi duplikat atau tidak duplikat dan pada setiap tipe data judul, deskripsi dan judul+deskripsi.

4.8.8 Evaluasi Skenario 2

Berdasarkan skenario 2 yaitu menggunakan algoritma Ratcliff/Obershelp sebagai pendeteksi *similarity* yang menggunakan data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani. Didapatkan hasil pada rata-rata berdasarkan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani pada indikasi tidak duplikat pada perbandingan pada Tabel 4.5 dan Tabel 4.6, bahwa penerapan algoritma *stemming* Nazief & Adriani membuat sebagian hasil nilai rata-rata *similarity* menjadi menurun. Penurunan tersebut tidak terjadi pada semua tipe data, ditampilkan pada Gambar 4.30 yang menjelaskan bahwa perbandingan antara data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani memiliki selisih penurunan jika menggunakan algoritma *stemming* Nazief & Adriani yang terdapat pada tipe data judul yaitu sebesar 0,23%, dan pada tipe data deskripsi sebesar 0,17% sedangkan untuk tipe data judul+deskripsi mengalami kenaikan nilai rata-rata *similarity* sebesar 0,19%. Sehingga pada produk indikasi tidak duplikat memiliki hasil rata-rata nilai *similarity* yang paling kecil yaitu jika menggunakan algoritma *stemming* Nazief & Adriani tepatnya pada tipe data deskripsi dengan nilai rata-rata *similarity* 9,69%.

Hasil rata-rata *similarity* pada indikasi duplikat Tabel 4.7 dan Tabel 4.8, bahwa penerapan algoritma *stemming* Nazief & Adriani membuat keseluruhan hasil nilai rata-rata *similarity* menjadi menurun yaitu pada semua tipe data judul,

deskripsi dan judul+deskripsi. Grafik perbandingan hasil rata-rata nilai *similarity* pada Gambar 4.31 yang menjelaskan bahwa perbandingan antara data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani memiliki selisih penurunan jika menggunakan algoritma *stemming* Nazief & Adriani yang terdapat pada semua tipe data, yakni pada tipe data judul mengalami penurunan sebesar 0,04%, pada tipe data deskripsi sebesar 0,88%, pada tipe data judul+deskripsi sebesar 3,81%.

Perbandingan waktu pemrosesan pada penerapan algoritma *stemming* Nazief & Adriani pada indikasi data duplikat dan tidak duplikat secara keseluruhan pada Gambar 4.32 dan Gambar 4.33 mengalami penambahan / kenaikan waktu pemrosesan, pada indikasi produk tidak duplikat mengalami kenaikan pada tipe data judul sebesar 0,47 detik, pada tipe data deskripsi sebesar 9,45 detik, pada tipe data judul+deskripsi sebesar 10,02 detik. Sedangkan untuk indikasi produk duplikat mengalami kenaikan pada tipe data judul sebesar 0,44 detik, pada tipe data deskripsi sebesar 11,93 detik, pada tipe data judul+deskripsi sebesar 12,35 detik. Sehingga pemrosesan waktu tercepat jika menggunakan data berdasarkan pemrosesan *text preprocessing* tanpa *stemming*.

4.8.8 Evaluasi Skenario 3

Berdasarkan skenario 3 yaitu menggunakan algoritma Winnowing dan algoritma Ratcliff/Obershelp sebagai pendeteksi *similarity* yang menggunakan data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani. Perbandingan Tabel 4.9 dan Tabel 4.10 memperoleh hasil bahwa dalam penerapan *text*

preprocessing menggunakan algoritma *stemming* Nazief & Adriani memberikan hasil rata-rata nilai *similarity* pada indikasi produk tidak duplikat secara garis besar mengalami penurunan pada setiap pengujian kecuali pada satu pengujian yaitu pada penerapannya di algoritma Ratcliff/Obershelp dengan tipe data judul+deskripsi yang mengalami kenaikan 0,19% meski demikian pada seluruh pengujian di algoritma Winnowing mengalami penurunan sedangkan pada algoritma Ratcliff/Obershelp pada 3 pengujian terdapat 2 pengujian mendapatkan nilai penurunan dan 1 pengujian mendapatkan nilai kenaikan pada nilai rata-rata *similarity*, grafik perbandingan tersebut ditampilkan pada Gambar 4.34. Sehingga hasil akurasi terbaik untuk indikasi produk tidak duplikat didapatkan hasil nilai terkecil yaitu dari algoritma Winnowing dengan menerapkan *stemming* algoritma Nazief & Adriani pada nilai $k = 7$ dan menggunakan tipe data deskripsi dengan nilai *similarity* 8,12%.

Sedangkan untuk data dengan indikasi produk duplikat yang ditampilkan pada Tabel 4.11 dan Tabel 4.12 memperoleh hasil bahwa dalam penerapan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani memberikan hasil rata-rata nilai *similarity* secara garis besar mengalami penurunan pada setiap pengujian kecuali pada satu pengujian yaitu pada penerapannya di algoritma Winnowing dengan nilai $k = 7$ pada tipe data deskripsi dan tipe data judul+deskripsi mengalami kenaikan pada tipe deskripsi sebesar 1,58% dan pada tipe judul+deskripsi sebesar 3,26% sedangkan untuk pengujian lainnya pada algoritma Winnowing mengalami penurunan sedangkan pengujian pada algoritma Ratcliff/Obershelp dengan menerapkan *text preprocessing* menggunakan algoritma

stemming Nazief & Adriani secara keseluruhan memberikan penurunan pada hasil nilai rata-rata *similarity*, grafik perbandingan tersebut ditampilkan pada Gambar 4.35. Sehingga hasil akurasi terbaik untuk indikasi produk duplikat didapatkan hasil nilai terbesar dari algoritma Ratcliff/Obershelp dengan menerapkan *text preprocessing* tanpa *stemming* Nazief & Adriani pada tipe data judul dengan nilai *similarity* 86,91%.

Berdasarkan grafik perbandingan kecepatan waktu pemrosesan pada Gambar 4.36 dan Gambar 4.37 pada indikasi produk tidak duplikat dan duplikat dengan menggunakan data yang berdasarkan pemrosesan *text preprocessing* tanpa *stemming* dan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani. Memberikan hasil bahwa secara keseluruhan penerapan algoritma *stemming* Nazief & Adriani menambah waktu pemrosesan ketika menganalisa tingkat kesamaan produk baik itu menggunakan algoritma Winnowing ataupun algoritma Ratcliff/Obershelp. Namun secara garis besar pada Gambar 4.36 pemrosesan waktu algoritma Ratcliff/Obershelp pada indikasi data tidak duplikat memiliki waktu pemrosesan lebih cepat daripada algoritma Winnowing pada rata2 dengan nilai $k = 2, 3, 5,$ dan 7 yaitu dengan selisih 0,01 detik pada tipe data deskripsi dan 0,0075 detik dan pada tipe data judul+deskripsi sedangkan untuk tipe data judul memiliki waktu pemrosesan yang sama diantara kedua algoritma tersebut. Hal tersebut sama dengan penerapan algoritma Nazief & Adriani, yaitu untuk tipe data judul memiliki waktu pemrosesan yang sama namun untuk tipe data deskripsi memiliki selisih 0,01 detik dan tipe data judul+deskripsi memiliki selisih 0,015 detik . Gambar 4.37 pada indikasi produk duplikat pemrosesan waktu algoritma

Ratcliff/Obershelp lebih cepat daripada algoritma Winnowing dengan nilai k = 2, 3, 5, dan 7 pada tipe data deskripsi dan judul+deskripsi sedangkan untuk tipe data judul memiliki nilai kecepatan yang sama. Berdasarkan data yang berasal dari pemrosesan *text preprocessing* tanpa *stemming* pada tipe data deskripsi memiliki selisih nilai kecepatan yaitu 0,015 detik dan pada tipe data judul+deskripsi memiliki selisih 0,01 detik. Sedangkan pada data yang berasal dari pemrosesan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani mendapatkan hasil yang sama yaitu memiliki kecepatan pemrosesan yang sama pada tipe data judul dan memiliki selisih pada tipe data deskripsi dan judul+deskripsi. Berdasarkan pada tipe data deskripsi memiliki selisih nilai kecepatan yaitu 0,01 detik dan pada tipe data judul+deskripsi memiliki selisih nilai kecepatan yaitu 0,015 detik.

4.8.9 Evaluasi Keseluruhan

Berdasarkan Skenario 1 pada penerapan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani pada algoritma Winnowing Gambar 4.26 dengan hasil dari Tabel 4.1 dan Tabel 4.2 diindikasi produk tidak duplikat memberikan penurunan rata-rata hasil tingkat *similarity* secara keseluruhan pada algoritma Winnowing, terjadinya penurunan tersebut didapatkan pada semua jenis data dengan tipe judul, deskripsi dan judul+deskripsi. Penurunan yang terjadi pada nilai k = 2 dengan tipe data judul didapatkan hasil yang dari tanpa menggunakan *text preprocessing* sebesar 38,85% (plagiarisme sedang) sedangkan pada penerapan *stemming* Nazief & Adriani sebesar 38,11% (plagiarisme sedang) sehingga selisihnya sebesar 0,74%, pada data dengan tipe deskripsi memiliki selisih 2,04%

dari 51,89% menjadi 49,84% (plagiarisme sedang), pada data dengan tipe judul+deskripsi sebesar 1,98% dari 53,12% menjadi 51,14% (plagiarisme sedang). Sedangkan untuk nilai $k = 3$ dengan tipe data judul didapatkan hasil selisih sebesar 0,16% dari 28,57% menjadi 28,41% (plagiarisme ringan), pada data dengan tipe deskripsi memiliki selisih 1,37% dari 23,7% menjadi 22,33% (plagiarisme ringan), pada data dengan tipe judul+deskripsi 1,35% dari 24,61% menjadi 23,26% (plagiarisme ringan). Selanjutnya untuk nilai $k = 5$ dengan tipe data judul didapatkan hasil selisih sebesar 0,27% dari 19,53% menjadi 19,26% (plagiarisme ringan), pada data dengan tipe deskripsi memiliki selisih 0,34% dari 10,55% menjadi 10,21% (plagiarisme ringan), pada data dengan tipe judul+deskripsi 0,33% dari 10,89% menjadi 10,56% (plagiarisme ringan). Pada nilai $k = 7$ dengan tipe data judul didapatkan hasil selisih sebesar 0,44% dari 13,01% menjadi 12,57% (plagiarisme ringan), pada data dengan tipe deskripsi memiliki selisih 0,29% dari 8,41% menjadi 8,12% (plagiarisme ringan), pada data dengan tipe judul+deskripsi 0,29% dari 8,55% menjadi 8,26% (plagiarisme ringan). Penurunan tersebut terjadi karena adanya perbedaan nilai *hash* pada rolling hash ketika ada kata yang telah diubah menjadi kata dasar, sehingga pada penentuan nilai *fingerprint* atau nilai terkecil pada window yang dibuat dipilih nilai terkecil yang berbeda karena walaupun pada kedua data yang dibandingkan sama-sama diterapkan perubahan kata dasar dengan algoritma *stemming* Nazief & Adriani namun kata yang disederhanakan memiliki letak yang berbeda sehingga ketika dibuatkan window akan ditemukan *fingerprint* yang berbeda sehingga ketika dilakukan perhitungan Jaccard Similarity jumlah *fingerprint* yang sama akan berkurang.

Gambar 4.27 pada grafik rata-rata hasil *similarity* dari Tabel 4.3 dan Tabel 4.4 dengan indikasi produk duplikat pada pengujian algoritma Winnowing secara keseluruhan pada nilai k = 2, 3 dan 5 mengalami penurunan sedangkan dengan nilai k = 7 mengalami mengalami 1 penurunan yaitu pada tipe data judul sebesar 0,34% dari 65,63% menjadi 65,29% (plagiarisme sedang) dan mengalami 2 kenaikan yaitu pada tipe data deskripsi dan tipe data judul+deskripsi mengalami kenaikan pada tipe deskripsi sebesar 1,58% dari 49,65% menjadi 51,23% (plagiarisme sedang) dan pada tipe judul+deskripsi sebesar 3,26% dari 44,8% menjadi 48,06% (plagiarisme sedang). Kenaikan tersebut terjadi pada nilai $k = 7$ pada beberapa analisa sehingga ketika dirata-rata terbentuk grafik pada nilai $k = 7$ pada tipe data deskripsi dan judul+deskripsi mengalami kenaikan karena ketika ada kata yang diubah menjadi kata dasar menggunakan algoritma *stemming* Nazief & Adriani dan dilakukan pemotongan *string* dengan jumlah $k = 7$ hal tersebut mengakibatkan jumlah total *fingerprint* setiap kedua data yang dibandingkan akan mengalami penurunan yang lebih besar, begitu juga terjadi penurunan nilai fingerpint yang sama namun penurunannya tidak sebesar dari penurunan total *fingerprint* sehingga ketika dihitung menggunakan Jaccard Similarity nilai pembagi akan lebih banyak berkurang daripada nilai kesamaan *fingerprint* yang menyebabkan terjadinya kenaikan nilai *similarity* pada tipe data deskripsi dan judul+deskripsi.

Berdasarkan skenario 2 pada pengujian algoritma Ratcliff/Obershelp didapakan hasil rata-rata pada Gambar 4.30 dengan berdasarkan hasil dari Tabel 4.5 dan Tabel 4.6 pada skenario pengujian dengan indikasi data tidak duplikat mengalami 1 kenaikan dan 2 penurunan pada 3 pengujian, penurunan terjadi pada

tipe data judul yang mengalami penurunan sebesar 0,23% dari 54,32% menjadi 54,09% (plagiarisme sedang), dan pada tipe data deskripsi yang mengalami penurunan sebesar 0,17% dari 9,86% menjadi 9,69% (plagiarisme ringan). Kenaikan yang terjadi pada tipe data judul+deskripsi yang mengalami kenaikan 0,19% dari 10,98% menjadi 11,17% (plagiarisme ringan). Kenaikan tersebut terjadi dibeberapa analisa dan diakibatkan karena ketika sebuah kata dirubah menjadi kata dasar menggunakan algoritma *stemming* Nazief & Adriani maka ketika dilakukan proses pada algoritma Ratcliff/Obershelp yang menggabungkan semua kata menjadi baris dan dihilangkan spasinya yang akan membuat deretan baris karakter akan ditemukan anchor atau jumlah karakter yang sama dengan susunan terpanjang selanjutnya akan dicari karakter yang sama lagi diantara potongan anchor yang terbentuk dan akan menyisakan karakter abstrak bukan berupa sebuah kata seperti per-karakter abjad sehingga dari potongan per-karakter itu akan menambah jumlah karakter yang sama sehingga akan menaikkan total jumlah karakter yang sama dan akan memberikan nilai presentase *similarity* yang lebih tinggi. Namun dibeberapa analisa lainnya terjadi penurunan secara konsisten pada semua skenario tipe data judul, deskripsi dan judul+deskripsi pada Gambar 4.31 berdasarkan hasil dari Tabel 4.7 dan Tabel 4.8 dengan skenario pengujian indikasi data duplikat memberikan penurunan rata-rata hasil tingkat *similarity* secara keseluruhan pada algoritma Ratcliff/Obershelp, pada tipe data judul penurunan terjadi sebesar 0,04% dari 86,91% menjadi 86,87 (plagiarisme berat atau total), pada tipe data deskripsi penurunan terjadi sebesar 0,88% yaitu dari 48,17% menjadi 47,29% (plagiarisme sedang) dan pada tipe data judul+deskripsi sebesar 3,81% dari 44,89% menjadi

41,08%(plagiarisme sedang) hal tersebut terjadi karena ketika menerapkan algoritma *stemming* Nazief & Adriani menyebabkan sebuah kata akan dirubah menjadi kata dasarnya dan hal tersebut mempengaruhi urutan dari deretan karakter pada proses algoritma Ratcliff/Obershelp sehingga ketika ditemukan pemotongan baris karakter berdasarkan anchor akan mendapatkan hasil deretan karakter yang berbeda antara kiri anchor dan kanan anchor.

Berdasarkan evaluasi skenario 3 dengan penerapan algoritma *stemming* Nazief & Adriani pada perbandingan algoritma Winnowing dan algoritma Ratcliff/Obershelp didapatkan hasil rata-rata perbandingan Tabel 4.9 dan Tabel 4.10 memperoleh hasil bahwa dalam penerapan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani memberikan hasil rata-rata nilai *similarity* pada indikasi produk tidak duplikat dengan algoritma Winnowing secara garis besar mengalami penurunan sedangkan pada algoritma Ratcliff/Obershelp pada 3 pengujian terdapat 2 pengujian mendapatkan nilai penurunan dan 1 pengujian mendapatkan nilai kenaikan pada nilai rata-rata *similarity*. Berdasarkan perbandingannya didapatkan hasil akurasi terbaik untuk indikasi produk tidak duplikat didapatkan hasil nilai terkecil yaitu dari algoritma Winnowing dengan menerapkan *stemming* algoritma Nazief & Adriani pada nilai k = 7 dan menggunakan tipe data deskripsi dengan nilai *similarity* 8,12% (plagiarisme ringan). Sedangkan untuk data dengan indikasi produk duplikat yang ditampilkan pada Tabel 4.11 dan Tabel 4.12 penerapan *text preprocessing* menggunakan algoritma *stemming* Nazief & Adriani memberikan hasil rata-rata nilai *similarity* pada algoritma Ratcliff/Obershelp secara keseluruhan mengalami penurunan

namun pada algoritma Winnowing secara garis besar mengalami penurunan kecuali pada skenario dengan nilai k = 7 pada tipe data deskripsi dan tipe data judul+deskripsi mengalami kenaikan pada tipe deskripsi sebesar 1,58% dari 49,65% menjadi 51,23% (plagiarisme sedang) dan pada tipe judul+deskripsi sebesar 3,26% dari 44,8% menjadi 48,06% (plagiarisme sedang). Berdasarkan perbandingannya didapatkan hasil akurasi terbaik untuk indikasi produk duplikat didapatkan hasil nilai terbesar dari algoritma Ratcliff/Obershelp dengan menerapkan *text preprocessing* tanpa *stemming* Nazief & Adriani pada tipe data judul dengan nilai *similarity* 86,91% (plagiarisme berat atau total).

Perbandingan efisiensi waktu pada Gambar 4.36 dan Gambar 4.37 secara keseluruhan penerapan algoritma *stemming* Nazief & Adriani menambah waktu pemrosesan pada setiap analisa yang dilakukan hal tersebut terjadi karena pada saat dilakukan penerapan algoritma *stemming* Nazief & Adriani dilakukan proses pada beberapa aturan yang ada serta kata akan dicocokkan pada kamus dasar Bahasa Indonesia. Detail pada pemrosesan tanpa menerapkan *stemming* mendapatkan nilai yang seimbang antara algoritma Winnowing dan algoritma Ratcliff/Obershelp dengan nilai rata-rata waktu pemrosesan yaitu 0,01 detik untuk kedua algoritma. Sedangkan pada tipe data deskripsi algoritma Winnowing menghasilkan 0,15 detik untuk produk tidak duplikat dan 0,18 detik untuk produk duplikat sehingga memiliki rata-rata waktu pemrosesan 0,17 detik. Sedangkan untuk algoritma Ratcliff/Obershelp pada tipe data deskripsi menghasilkan 0,14 detik untuk produk tidak duplikat dan 0,17 detik untuk produk duplikat sehingga memiliki rata-rata waktu pemrosesan 0,16 detik sehingga pada tipe data deskripsi algoritma

Ratcliff/Obershelp lebih efisien dengan selisih 0,01 detik. Tipe data judul+deskripsi pada algoritma Winnowing menghasilkan 0,15 detik untuk produk tidak duplikat dan 0,18 detik untuk produk duplikat sehingga memiliki rata-rata waktu pemrosesan 0,17 detik. Sedangkan pada algoritma Ratcliff/Obershelp menghasilkan 0,14 detik untuk produk tidak duplikat dan 0,17 detik untuk produk duplikat sehingga memiliki rata-rata waktu pemrosesan 0,16 detik sehingga algoritma Ratcliff/Obershelp lebih efisien dengan selisih 0,01 detik.

Setelah menerapkan *stemming* Nazief & Adriani pada algoritma Winnowing dan algoritma Ratcliff/Obershelp dari awal proses *text preprocessing* sampai ditemukan hasil nilai *similarity* didapatkan hasil penambahan waktu pemrosesan pada setiap skenario yang dilakukan yaitu mendapatkan hasil yang seimbang antara kedua algoritma dengan tipe data judul dengan hasil 0,48 detik pada produk tidak duplikat dan 0,45 detik untuk produk duplikat sehingga mendapatkan rata-rata waktu pemrosesan 0,47 detik, hasil ini lebih besar daripada sebelumnya yang tanpa menggunakan *stemming* yaitu 0,01 detik sehingga selisihnya 0,46 detik. Tipe data deskripsi pada algoritma Winnowing menghasilkan 9,60% untuk produk tidak duplikat dan 12,11 detik untu produk duplikat sehingga memiliki rata-rata waktu pemrosesan 10,86 detik, hasil tersebut lebih besar daripada tanpa menggunakan *stemming* yaitu 0,17 detik sehingga selisihnya 10,69 detik. Sedangkan pada algoritma Ratcliff/Obershelp menghasilkan 9,59 detik untuk produk tidak duplikat dan 12,10 detik untuk produk duplikat sehingga memiliki rata-rata waktu pemrosesan 10,85 detik. Sehingga pada tipe data deskripsi algoritma Ratcliff/Obershelp lebih efisien dengan hasil rata-rata waktu pemrosesan

10,85 dengan selisih 0,01 terhadap algoritma Winnowing, namun nilai hasil 10,85 tersebut lebih besar daripada tanpa menggunakan *stemming* yaitu 0,16 detik sehingga selisihnya 10,69 detik. Tipe data judul+deskripsi pada algoritma Winnowing mendapatkan hasil 10,18 detik untuk produk tidak duplikat dan 12,54 detik untuk produk duplikat sehingga memiliki hasil rata-rata waktu pemrosesan 11,36 detik, hasil ini lebih besar dibandingkan tanpa menggunakan *stemming* yaitu 0,17 detik sehingga selisihnya 11,19 detik. Sedangkan untuk Ratcliff/Obershelp menghasilkan 10,16 detik untuk produk tidak duplikat dan 12,52 detik untuk produk duplikat sehingga memiliki rata-rata waktu pemrosesan 11,34 detik. Sehingga pada tipe data judul+deskripsi algoritma Ratcliff/Obershelp lebih efisien dengan hasil rata-rata waktu pemrosesan 11,34 dengan selisih 0,02 terhadap algoritma Winnowing, namun nilai hasil 11,34 tersebut lebih besar daripada tanpa menggunakan *stemming* yaitu 0,16 detik sehingga selisihnya 11,18 detik.

Sehingga dapat diketahui total selisih penambahan waktu pemrosesan terhadap penerapan *stemming* Nazief & Adriani pada algoritma Winnowing yaitu pada tipe data judul dengan rata-rata selisih 0,46 detik, tipe data deskripsi 10,69 detik dan tipe data judul+deskripsi 11,19 detik, sehingga didapatkan rata-rata selisih keseluruhan pada algoritma Winnowing setelah menggunakan penerapan *stemming* Nazief & Adriani yaitu menambah waktu pemrosesan dengan rata-rata selisih 7,45 detik. Sedangkan pada algoritma Ratcliff/Obershelp dengan tipe data judul memiliki rata-rata selisih 0,46 detik, tipe data deskirpsi 10,69 detik dan tipe data judul+deskripsi 11,18 detik, sehingga didapatkan rata-rata selisih keseluruhan pada

algoritma Ratcliff/Obershelp setelah menggunakan penerapan *stemming* Nazief & Adriani yaitu menambah waktu pemrosesan dengan rata-rata selisih 7,44 detik.

4.8.10 Evaluasi Perbandingan Terhadap Studi Literatur

Berdasarkan hasil evaluasi yang telah dilaksanakan selanjutnya dilakukan perbandingan terhadap hasil evaluasi penelitian terhadap studi literatur yang ada. Penelitian yang dilakukan oleh (Agusta, 2009) mengenai perbandingan *stemming* dari algoritma Porter dan algoritma Nazief & Adriani dengan menggunakan uji coba 30 data teks dengan ukuran bervariasi yang menyimpulkan bahwa algoritma Nazief & Adriani membutuhkan waktu yang lebih lama dibandingkan algoritma Porter hal tersebut memiliki hasil yang sama apabila dihadapkan dengan penelitian yang dilakukan karena dengan menerapkan algoritma Nazief & Adriani menambah waktu pemrosesan karena terdapat proses dari aturan yang telah ditentukan beserta pencocokan kata menggunakan kamus Bahasa Indonesia.

Perbandingan penelitian algoritma *stemming* (Wahyudi, dkk, 2017) yang membandingkan algoritma Porter dengan algoritma Nazief & Adriani dengan menggunakan data kata sebanyak 2132 kata menyimpulkan bahwa penerapan algoritma Nazief & Adriani memiliki akurasi 95,26% yang lebih baik daripada algoritma Porter yaitu 79,13% namun waktu pemrosesan yang paling efisien didapatkan algoritma Porter dengan waktu proses 12,38 detik sedangkan algoritma Nazief & Adriani menghasilkan waktu proses 22,16 detik sehingga waktu pemrosesan untuk algoritma Nazief & Adriani lebih lama. Hal tersebut memiliki hasil yang sama apabila dihadapkan dengan penelitian yang dilakukan karena dengan menerapkan algoritma Nazief & Adriani menambah waktu pemrosesan.

Penelitian mengenai algoritma Nazief & Adriani lainnya (Simarangkir, 2017) dengan membandingkan algoritma Nazief & Adriani, algoritma Arifin & Setiono, algoritma Vega dan algoritma Tala dengan menggunakan dataset pengujian menggunakan 100 dokumen teks berBahasa Indonesia menghasilkan kesimpulan bahwa algoritma Nazief & Adriani memiliki akurasi yang paling baik daripada algoritma lainnya dengan nilai akurasi Nazief & Adriani 97,93% dengan waktu proses 5,14 detik sedangkan untuk Arifin & Setiono memiliki nilai akurasi 92,09% dan waktu pemrosesan 15,20 detik, algoritma Vega dengan akurasi 63,48% dan waktu pemrosesan 0,08 detik, sedangkan untuk Tala memiliki nilai akurasi 78,27% dan waktu pemrosesan 0,22 detik sehingga waktu pemrosesan tercepat adalah algoritma Vega. Hasil penelitian tersebut menghasilkan pemrosesan waktu algoritma Nazief & Adriani jika dibandingkan dengan algoritma Arifin & Setiono lebih efisien namun jika dibandingkan dengan algoritma lainnya menambah waktu. Hal tersebut mempunyai karakteristik yang sama dengan penelitian yang dilakukan, karena penerapan algoritma Nazief & Adriani menambah pemrosesan.

Perbandingan algoritma *similarity* (Wibowo, dkk, 2013) dengan mambandingkan metode Fingerprint dan algoritma Winnowing yang melakukan pengujian dengan 80 data uji, didapatkan bahwa hasil akurasi pada pendeteksi *similarity* untuk metode *fingerprint* yaitu 92,8% sedangkan untuk algoritma Winnowing memiliki akurasi 91,8%. Karena yang dicari pada penelitian tersebut adalah nilai *similarity* terkecil maka disimpulkan bahwa algoritma Winnowing memiliki kinerja dan performa yang lebih baik berdasarkan hasil pendeteksi nilai *similarity* yang terkecil. Hal tersebut memiliki hasil yang sama terhadap penelitian

yang dilakukan karena pada semua pengujian algoritma Winnowing memiliki hasil nilai terkecil pada k = 2, 3, 5 dan 7 pada indikasi produk duplikat dan tidak duplikat pada tipe data judul jika dihadapkan dengan pemrosesan menggunakan algoritma Ratcliff/Obershelp pada tipe data yang sama, serta pada indikasi produk tidak duplikat diperoleh nilai terkecil yaitu dari algoritma Winnowing dengan menerapkan *stemming* algoritma Nazief & Adriani pada nilai k = 7 dan menggunakan tipe data deskripsi dengan nilai *similarity* 8,12% pada skenario 3. Namun berbeda jika dihadapkan dengan data deskripsi dan judul+deskripsi karena secara garis besar algoritma Winnowing dengan nilai k = 2, 3, 5 dan 7 memiliki hasil nilai *similarity* yang tinggi jika dibandingkan dengan penerapan algoritma Ratcliff/Obershelp, namun pada nilai k = 7 dengan indikasi produk tidak duplikat memiliki nilai *similarity* yang lebih rendah.

Analisis mengenai perbandingan algoritma *similarity* lainnya (Ilyankou, 2014) yaitu membandingkan algoritma Ratcliff/Obershelp dengan algoritma Jaro-Winkler dengan menggunakan dataset kata dalam bahasa Inggris sebanyak 58000 dan 236000 entri untuk mengidentifikasi ejaan dan didapatkan hasil bahwa penerapan algoritma Ratcliff/Obershelp memiliki hasil yang lebih baik dengan nilai 4,0% hingga 18,6% daripada algoritma Jaro-Winkler. Hal tersebut memiliki hasil karakteristik yang sama jika dihadapkan dengan penelitian yang dilakukan karena penerapan algoritma Ratcliff/Obershelp dalam mendeteksi kesamaan judul yang memiliki karakteristik karakter yang sedikit sehingga memiliki nilai *similarity* lebih tinggi daripada algoritma Winnowing pada pengujian data duplikat ataupun data tidak duplikat.

Penelitian lainnya mengenai algoritma Winnowing (Alamsyah, 2017) yang melakukan penelitian dengan variasi n-gram denan nilai n = 2, 3, 4, 5, 6, 7, 8, 9, 10 pada penerapannya dalam algoritma Winnowing (dengan nilai window 3, 5 dan 7) dibandingkan dengan algoritma Rabin-Karp, hasil dari penelitian tersebut mencari nilai terkecil dari tingkat *similarity* sehingga didapatkan kesimpulan bahwa semakin besar nilai n-gram akan menghasilkan nilai *similarity* yang kecil sehingga ditemukan bahwa penerapan algoritma Winnowing dengan nilai n = 9 dan 10 serta nilai w = 3 mendapatkan nilai terbaik dengan nilai terkecil. Hasil analisa tersebut sama dengan hasil analisis yang dilakukan yaitu menggunakan variasi k-gram dengan nilai k jika semakin besar maka akan memperkecil nilai *similarity*.

Penerapan *stemming* pada algoritma *similarity* (Sukmana, dkk, 2018) yang menerapkan *stemming* Nazief & Adriani pada algoritma Rabin-Karp dengan dataset 100 kata, 75 kata, 50 kata dan 25 kata menghasilkan kesimpulan bahwa penerapan *stemming* Nazief & Adriani menurunkan hasil *similarity* secara keseluruhan jika data tidak sama 100% dengan perbandingan antara Rabin-Karp murni dengan Rabin-Karp dengan *stemming* yaitu 100%:100%, 74,87%:72,09%, 48,39%:45,69%, 23,60%:20:59%, sedangkan penerapan *stemming* Nazief & Adriani mempercepat proses dalam algoritma Rabin-Karp namun bukan proses keseluruhan. Hasil tersebut mempunyai kesamaan dengan penelitian yang dilakukan karena terjadi penurunan hasil nilai *similarity* secara garis besar pada skenario yang dilakukan, dan untuk pemrosesan waktu pada penelitian ini diukur dari awal text preprocessing sampai hasil nilai *similarity* ditemukan sehingga tidak diukur secara spesifik untuk pemrosesan pada algoritma.

Analisis yang menggunakan algoritma Ratcliff/Obershelp (Yusuf, dkk, 2019) yang membandingkan algoritma Ratcliff/Obershelp dengan algoritma Rabin-Karp dalam mendeteksi *similarity* pada 10 data uji yang memiliki jumlah karakter yang berbeda-beda menghasilkan kesimpulan bahwa algoritma Ratcliff/Obershelp menghasilkan persentase tingkat *similarity* yang lebih mendekati persentase data masukan daripada algoritma Rabin-Karp sehingga algoritma Ratcliff/Obershelp lebih unggul kurang lebih 3% dari total pengujian yang dilakukan serta kesimpulan lainnya penelitian tersebut menyimpulkan bahwa Ratcliff/Obershelp memiliki waktu proses yang lebih cepat dibandingkan algoritma Rabin-Karp dan tata letak kata mempengaruhi hasil dari algoritma Rabin-Karp namun tidak mempengaruhi hasil dari Ratcliff/Obershelp. Hasil tersebut jika dibandingkan dengan penelitian yang dilakukan memiliki kesamaan yaitu penerapan algoritma Ratcliff/Obershelp lebih unggul untuk mendeteksi tingkat *similarity* yang mendekati presentase yang tinggi serta pemrosesan algoritma Ratcliff/Obershelp lebih efisien namun pada penelitian yang dilakukan disimpulkan bahwa penempatan karakter ataupun kata akan mempengaruhi hasil *similarity* dari algoritma Ratcliff/Obershelp karena pemrosesan pada algoritma Ratcliff/Obershelp menggabungkan semua karakter dan tidak dipecah berdasarkan kata menggunakan spasi sehingga jika letak karaktemya berbeda maka terjadi perbedaan ketika dipotong berdasarkan anchor dan *string* yang sama.

BAB V

PENUTUP

5.1. Kesimpulan

Kesimpulan yang dihasilkan berdasarkan skenario pengujian dan evaluasi diantaranya adalah :

a. Penerapan algoritma *stemming* Nazief & Adriani pada algoritma Winnowing dengan nilai k = 2, 3, 5, dan 7 serta nilai w = 2 dan menggunakan tipe data judul, deskripsi dan judul+deskripsi menyebabkan penurunan hasil nilai rata-rata *similarity* sebanyak 22 hasil dan terdapat 2 hasil kenaikan terhadap 24 hasil pada rata-rata *similarity* pada skenario yang telah dilakukan.

b. Penerapan algoritma *stemming* Nazief & Adriani pada algoritma Ratcliff/Obershelp menyebabkan penurunan hasil nilai rata-rata *similarity* sebanyak 5 hasil dan terdapat 1 hasil kenaikan terhadap 6 hasil rata-rata *similarity* pada skenario yang telah dilakukan.

c. Secara keseluruhan algoritma *stemming* Nazief & Adriani menambah waktu pemrosesan dari proses *text preprocessing* hingga ditemukan hasil nilai *similarity* pada algoritma Winnowing dengan total rata-rata selisih penambahan waktu 7,45 detik dan algoritma Ratcliff/Obershelp dengan total rata-rata selisih penambahan waktu 7,44 detik.

d. Pemrosesan tanpa *stemming* pada tipe data judul berdasarkan algoritma Winnowing dan algoritma Ratcliff/Obershelp memiliki rata-rata waktu pemrosesan yang seimbang yaitu dengan rata-rata waktu pemrosesan 0,01 detik, sedangkan pada tipe data deskripsi dan judul+deskripsi lebih efisien algoritma Ratcliff/Obershelp dengan rata-rata selisih 0,01 detik.

e. Pemrosesan dengan *stemming* Nazief & Adriani pada tipe data judul berdasarkan algoritma Winnowing dan algoritma Ratcliff/Obershelp memiliki rata-rata waktu pemrosesan yang seimbang yaitu dengan rata-rata waktu pemrosesan 0,47 detik, sedangkan pada tipe data deskripsi dan judul+deskripsi lebih efisien algoritma Ratcliff/Obershelp dengan rata-rata selisih 0,01 detik dan 0,02 detik terhadap algoritma Winnowing.

f. Hasil akurasi terbaik pada indikasi produk tidak duplikat diperoleh nilai terkecil yaitu dari algoritma Winnowing dengan menerapkan *stemming* algoritma Nazief & Adriani pada nilai $k = 7$ dan menggunakan tipe data deskripsi dengan nilai *similarity* 8,12% (plagiarisme ringan).

g. Pengujian pada indikasi produk duplikat diperoleh nilai terbesar yaitu dari algoritma Ratcliff/Obershelp dengan menerapkan *text preprocessing* tanpa *stemming* Nazief & Adriani pada tipe data judul dengan nilai *similarity* 86,91% (plagiarisme berat atau total).

h. Penerapan nilai variasi k pada algoritma Winnowing pada semua skenario yang dilakukan memberikan hasil nilai *similarity* semakin kecil jika nilai k semakin besar.

i. Jenis data atau jumlah karakter mempengaruhi nilai *similarity* pada algoritma Ratcliff/Obershelp, jika karakter yang diproses sedikit akan memberikan hasil nilai *similarity* lebih tinggi daripada tipe data dengan jumlah karakter yang lebih besar.

5.2. Saran

Terdapat beberapa saran untuk penelitian selanjutnya, diantaranya :

a. Menambah aturan untuk mendeteksi bahasa alay pada penerapan algoritma *stemming* Nazief & Adriani.
b. Disarankan menambahkan pengecekan dalam kesalahaan ejaan dan sinonim pada teks yang diolah.
c. Perlu dilakukan perbandingan algoritma *stemming* untuk mengetahui perbandingan algoritma *stemming*.

DAFTAR PUSTAKA

PUSTAKA BUKU

Feldman, R. and Sanger, J., 2007, The Text mining Handbook: Advanced Approaches to Analyzing Unstructured Data. New York: Cambridge University Press. doi: 10.1177/0261018311403863.

Gunawan, L. H., 2019, Pengaruh Electronic Word Of Mouth Terhadap Purchase Intention pada Produk Aloevera Nature Republic di Surabaya Melalui Mediasi Brand Image dan Brand Trust.

Sugiyono., 2016, Metode Penelitian Kuantitatif, Kualitatif dan R&D. Bandung: Alfabeta.

Riduwan, and Akdon, 2013, Rumus dan Data dalam Analisis Statistika. Bandung: Alfabeta.

PUSTAKA MAJALAH, JURNAL ILMIAH ATAU PROSIDING

Abdullah, 2018, Daya Saing Ott (Over The Top) E-commerce Indonesia Dalammenghadapi Persaingan Global, Jurnal Ilmu Ekonomi Islam, 2(1), pp. 26–52.

Adriani, M., Asian, J., Nazief, B., Tahaghoghi, S. M. M., & Williams, H. E., 2005, Stemming Indonesian: A Confix-Stripping Approach. Conferences in Research and Practice in Information Technology Series, 38(4), 307–314. https://doi.org/10.1145/1316457.1316459

Agusta, L., 2009, Perbandingan Algoritma Stemming Porter Dengan Algoritma Nazief & Adriani Untuk Stemming Dokumen Teks Bahasa Indonesia, Konferensi Nasional Sistem dan Informatika 2009, (KNS&I09-036), pp. 196–201.

ALAMSYAH, N., 2017, Perbandingan Algoritma Winnowing Dengan Algoritma Rabin Karp Untuk Mendeteksi Plagiarisme Pada Kemiripan Teks Judul Skripsi, Technologia: Jurnal Ilmiah, 8(3), p. 124. doi: 10.31602/tji.v8i3.1116.

Amalia, A. et al., 2019, Automated Bahasa Indonesia essay evaluation with latent semantic analysis, Journal of Physics: Conference Series, 1235(1). doi: 10.1088/1742-6596/1235/1/012100.

Broder, A. Z., 1997, On the resemblance and containment of documents. Proceedings of the International Conference on Compression and Complexity of Sequences, 21–29. https://doi.org/10.1109/sequen.1997.666900

Diartono, D. A., Suhari, Y. and Supriyanto, A., 2015, Pengembangan Model CYBER CLUSTER E-COMMERCE Berbasis CMS dan SEO Produk UMKM, IJCCS (Indonesian Journal of Computing and Cybernetics Systems), 9(2), p. 145. doi: 10.22146/ijccs.7543.

Fauzi, R. M. and Wibawa, J. C., 2018, Implementasi Algoritma Winnowing untuk Mendeteksi Kemiripan Teks pada Artikel.

Hannisa, O. :, Hasnin, R. and Sos, S., 2016, Inovasi Produk Melalui Strategi Imitasi Dalam Menghadapi Persaingan Produk Impor (Implementasi Strategi Imitasi Pada Studi Kasus Edam Burger Di Depok), Jurnal Ilmiah Inovator, Edisi Maret, (1996).

Heidarian, A. and Dinneen, M. J., 2016, A Hybrid Geometric Approach for Measuring Similarity Level among Documents and Document Clustering, Proceedings - 2016 IEEE 2nd International Conference on Big Data Computing Service and Applications, BigDataService 2016, 1994, pp. 142–151. doi: 10.1109/BigDataService.2016.14.

Hermawan, B. I., 2015, Analisis Performansi Algoritma Winnowing dan Algoritma Manber untuk Deteksi Kesamaan Dokumen Teks Berbahasa Indonesia.

Hotana, M. S., 2018, Industri E-commerce dalam Menciptakan Pasar yang Kompetitif Berdasarkan Hukum Persaingan Usaha, Jurnal Hukum Bisnis Bonum Commune, 12, pp. 28–38.

Ilyankou, I., 2014, Comparison of Jaro-Winkler and Ratcliff / Obershelp algorithms in spell check, Ieee, (May), pp. 1–35.

Imbar, R. V. et al., 2014, Implementasi Cosine Similarity dan Algoritma Smith-Waterman untuk Mendeteksi Kemiripan Teks, Jurnal Informatika, 10(1), pp. 31–42. doi: 10.1017/CBO9781107415324.004.

Jacob, V. E., Lumenta, A. S. M. and Jacobus, A., 2019, Rancang Bangun Aplikasi Kemiripan Dokumen Dengan Sumber – Sumber Internet, Rancang Bangun Aplikasi Kemiripan Dokumen Dengan Sumber – Sumber Internet, 14(2), pp. 159–164. doi: 10.35793/jti.14.2.2019.23990.

Ji, J., Li, J., Yan, S., Tian, Q., & Zhang, B., 2013, Min-max hash for jaccard similarity. Proceedings - IEEE International Conference on Data Mining, ICDM, 301–309. https://doi.org/10.1109/ICDM.2013.119

Joane, Y. L., Sinsuw, A. and Jacobus, A., 2017, Rancang Bangun Aplikasi Deteksi Kemiripan Dokumen Teks Menggunakan Algoritma Ratcliff/Obershelp, Jurnal Teknik Informatika, 11(1). doi: 10.35793/jti.11.1.2017.17654.

Leonardo, B. and Hansun, S., 2017, Text Documents Plagiarism Detection using Rabin-Karp and Jaro-Winkler Distance Algorithms, Indonesian Journal of

Electrical Engineering and Computer Science, 5(2), pp. 462–471. doi: 10.11591/ijeecs.v5.i2.pp462-471.

Maulana, S. M., Susilo, H. and Riyadi, 2015, Implementasi E-commerce Sebagai Media Penjualan Online, Jurnal Administrasi Bisnis, 29(1), pp. 1–9.

Muhammad, A. N., Bukhori, S. and Pandunata, P., 2019, Sentiment Analysis of Positive and Negative of YouTube Comments Using Naïve Bayes – Support Vector Machine (NBSVM) Classifier, 2019 International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE). IEEE, 1, pp. 199–205.

Novitasari, D., 2017, Perbandingan Algoritma Stemming Porter dengan Arifin Setiono untuk Menentukan Tingkat Ketepatan Kata Dasar, STRING (Satuan Tulisan Riset dan Inovasi Teknologi), 1(2), p. 120. doi: 10.30998/string.v1i2.1031.

Putranto, F. F. et al., 2019, Strategi Pengembangan Usaha Mikro di Kota Samarinda, Riset Inossa, 1, pp. 13–27.

Putri Ratna, A. A. et al., 2019, Investigating Parallelization of Cross-language Plagiarism Detection System Using the Winnowing Algorithm in Cloud Based Implementation, 2019 IEEE 10th International Conference on Awareness Science and Technology, iCAST 2019 - Proceedings. IEEE, pp. 1–7. doi: 10.1109/ICAwST.2019.8923539.

Purba, A. H. and Situmorang, Z., 2017, Analisis Perbandingan Algoritma Rabin-Karp Dan Levenshtein Distance Dalam Menghitung Kemiripan Teks, Jurnal Teknik Informatika Unika St. Thomas (JTIUST), 02, pp. 24–32.

Rahmatulloh, A. et al., 2019, Comparison between the stemmer porter effect and Nazief & Adriani on the performance of winnowing algorithms for measuring plagiarism, International Journal on Advanced Science, Engineering and Information Technology, 9(4), pp. 1124–1128. doi: 10.18517/ijaseit.9.4.8844.

Riki, Edy and Maryanto, 2019, Plagiarism Detection Application Uses Winnowing Algorithm with Synonym Recognition for Indonesian Text Documents, Selangor Science &Technology Review, 3(1), pp. 1–14.

Sanjaya, S., and Absar, E. A., 2015, Pengelompokan Dokumen Menggunakan Winnowing Fingerprint dengan Metode K-Nearest Neighbour. Jurnal CoreIT: Jurnal Hasil Penelitian Ilmu Komputer Dan Teknologi Informasi, 1(2), 50–56. https://doi.org/10.24014/coreit.v1i2.1229

Saputra, W. A., Nafan, M. Z. and Nurrochman, A., 2019, Implementasi Keras Library dan Convolutional Neural Network Pada Konversi Formulir

Pendaftaran Siswa, Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), 1(10), pp. 524–531.

Sastroasmoro, Sudigdo., 2007, Beberapa Catatan Tentang Plagiarisme. Maj Kedokt Indon Volum: 57,: 239–44.

Schleimer, S., Wilkerson, D. S., & Aiken, A., 2003, Winnowing: Local Algorithms for Document Fingerprinting. Proceedings of the ACM SIGMOD International Conference on Management of Data, 76–85.

Sibarani, L., Magdalena, M. and Dharma, A., 2019, Analisa Perbandingan Sistem Pendeteksian Kemiripan Judul Skripsi Menggunakan Algoritma Winnowing Dan Algoritma Rabin Karp, REMIK (Riset dan E-Jurnal Manajemen Informatika Komputer), 4(1), p. 69. doi: 10.33395/remik.v4i1.10174.

Simamora, A. A. N. and AK, M. F., 2019, Kemudahan Aplikasi dan Keragaman Produk dalam Membentuk Keputusan Pembelian Generasi Milenial Berbelanja secara Online, Jurnal Maneksi, 8(2), pp. 213–222.

Simarangkir, M. S. H., 2017, Studi Perbandingan Algoritma - Algoritma Stemming Untuk Dokumen Teks Berbahasa Indonesia, Jurnal Inkofar, 1(1), pp. 41–47.

Stein, Benno, and Sven Meyer zu Eissen., 2006, Near Similarity Search and Plagiarism Analysis. (1993): 430–37.

Sukmana, A., Kusrini and Sunyoto, A., 2018, Perbandingan Pengunaan Stemming Pada Deteksi Kemiripan Dokumen Menggunakan Metode Rabin Karp Dan Jaccard Similarity, Seminar Nasional Teknologi Informasi dan Multimedia, pp. 67–72.

Sunardi, S., Yudhana, A. and Mukaromah, I. A., 2018, Implementasi Deteksi Plagiarisme Menggunakan Metode N-Gram Dan Jaccard Similarity Terhadap Algoritma Winnowing, Transmisi, 20(3), p. 105. doi: 10.14710/transmisi.20.3.105-110.

Surahman, A., Octaviansyah, F. A. and Darwis, D., 2020, Ekstraksi Data Produk E-Marketplace Sebagai Strategi Pengolahan Segmentasi Pasar Menggunakan Web Crawler, Jurnal Sistem Informasi, 9(1), pp. 73–81.

Syahrial et al., 2020, Effect of Online Marketing (E-commerce) Activities to the Advantages of Competing Traditional Culinary Products in South Sulawesi, Journal of Tourism, Hospitality, Travel and Busines Event, 2(1), pp. 66–72.

Wahyudi, D., Susyanto, T. and Nugroho, D., 2017, Implementasi Dan Analisis Algoritma Stemming Nazief & Adriani Dan Porter Pada Dokumen Berbahasa Indonesia, Jurnal Ilmiah SINUS, 15(2), pp. 49–56. doi: 10.30646/sinus.v15i2.305.

Wibowo, A. T., Sudarmadi, K. W. and Barmawi, A. M., 2013, Comparison between fingerprint and winnowing algorithm to detect plagiarism fraud on Bahasa Indonesia documents, 2013 International Conference of Information and Communication Technology, ICoICT 2013, pp. 128–133. doi: 10.1109/ICoICT.2013.6574560.

Wicaksono, Y. A., 2012, Analisis Dan Implementasi Algoritma Rabin-Karp Dan Algoritma Stemming Nazief & Adriani Pada Sistem Pendeteksi Plagiat Dokumen.

Winarti, T., Kerami, J. and Arief, S., 2017, Determining Term on Text Document Clustering using Algorithm of Enhanced Confix Stripping Stemming, International Journal of Computer Applications, 157(9), pp. 8–13. doi: 10.5120/ijca2017912761.

Yusuf, B. et al., 2019, Analisis Perbandingan Algoritma Rabin-Karp dan Ratcliff / Obershelp untuk Menghitung Kesamaan Teks dalam Bahasa Indonesia, Seminar Nasional APTIKOM (SEMNASTIK), pp. 61–69.

Zaidi, S. A. J., Buriro, A., Riaz, M., Mahboob, A., & Riaz, M. N., 2019, Implementation and comparison of text-based image retrieval schemes. International Journal of Advanced Computer Science and Applications, 10(1), 611–618. https://doi.org/10.14569/IJACSA.2019.0100177

PUSTAKA LAPORAN PENELITIAN

Rezalina, O. 2016, Perbandingan Algoritma Stemming Nazief & Adriani, Porter dan Arifin Setiono untuk Dokumen Teks Bahasa Indonesia, Journal of Undergraduate Thesis, Universitas Muhammadiyah Jember, pp. 1–5.