Perbaikan Kesalahan Kata Menggunakan Kombinasi Jaro-Winkler & Jaccard Similarity

(Spelling Checker Correction Using Combination Jaro-Winkler & Jaccard Similarity)

I Made Agus Tresna[1]*, Ramaditia Dwiyansaputra[1], Halil Akhyar[1]
[1]Dept Informatics Engineering, University Of Mataram
Jl. Majapahit 62, Mataram, Lombok NTB, INDONESIA
Email: agus.tresna88@gmail.com, rama@unram.ac.id, halil.akhyar@staff.unram.ac.id

Abstract

Word error correction is challenging due to the variety of errors. This research proposes a combination of two similarity algorithms to improve accuracy. The objective is to evaluate how each algorithm responds to different types of spelling errors and to assess the effectiveness of their combined performance. Jaro-Winkler determines initial similarity by assigning more weight to word prefixes, effectively addressing errors due to transposition and character omission. This algorithm excels in scenarios where the beginning of the word is critical to identifying the correct candidate. In contrast, Jaccard similarity filters candidates based on character set similarity, which helps assess the overall composition similarity of the word but does not consider character order. The test results show that Jaro-Winkler is more dominant in providing relevant correction candidates, with higher accuracy in the 1-best (68.94%) and 5-best (90.78%) scenarios compared to Jaccard (55.25% and 78.42%). This performance difference suggests that Jaro-Winkler is more suitable for the initial screening of candidates. The combination of the two algorithms proved to be more effective in handling different types of word errors than when used separately, resulting in a more robust overall correction mechanism.

Keywords: *Natural Language Processing, Jaro-Winkler, Jaccard Similarity*

**Correspondence Author*

1. PENDAHULUAN

Kesalahan tipografi merupakan salah satu kesalahan umum dalam penulisan, yang kerap muncul karena gagalnya mekanisme atau jari yang licin saat mengetik [1]. Walaupun perangkat lunak pengolah kata memiliki fitur perbaikan otomatis, pemanfaatannya belum optimal sehingga kesalahan masih sering terjadi[1]. Para penulis kerap tidak menyadari kesalahan ini karena terlalu fokus pada isi. Ini dapat mempengaruhi kualitas karya tulis ilmiah, yang memerlukan pemeriksaan manual terhadap teks meskipun proses ini memakan banyak waktu[1]. Oleh sebab itu, diperlukan teknik khusus untuk memperbaiki kesalahan penulisan, seperti memanfaatkan algoritma pemrosesan bahasa alami (*Natural Language Processing*, NLP)[1].

NLP adalah bidang ilmu komputer yang bertujuan untuk memahami dan memproses bahasa manusia. NLP mengintegrasikan kecerdasan buatan dan linguistik untuk membantu komputer memahami serta menafsirkan bahasa alami yang digunakan oleh manusia. Pada zaman digital ini, informasi tersebar luas melalui berbagai platform daring, termasuk artikel berita yang diterbitkan secara elektronik[2]. Artikel berita online menjadi sumber daya informasi yang sangat penting bagi masyarakat karena kemampuannya menyajikan berita dengan cepat serta mudah diakses oleh siapa saja. Namun, seringkali artikel-artikel tersebut mengandung kesalahan penulisan (*typos*) yang dapat mengurangi kredibilitas informasi yang disampaikan. Kesalahan ejaan dalam artikel berita dapat terjadi karena berbagai sebab, seperti kesalahan ketik, otomatisasi koreksi yang tidak tepat, atau kesalahan manusia selama proses pengetikan.

Dalam era digital saat ini, penggunaan teks sebagai media komunikasi dan penyimpanan informasi menjadi sangat penting, terutama dalam konteks akademik dan profesional [3]. Namun, seiring dengan meningkatnya jumlah dokumen teks, muncul tantangan baru dalam menjaga kualitas tulisan, terutama terkait dengan kesalahan penulisan dan kesalahan ejaan [3]. Kesalahan dalam penulisan tidak hanya mengurangi kredibilitas sebuah dokumen, tetapi juga dapat mempengaruhi pemahaman pembaca terhadap konten yang disampaikan[4]. Oleh karena itu, diperlukan sistem yang mampu mendeteksi dan

memperbaiki kesalahan penulisan secara efektif dan efisien[4].

Dalam beberapa tahun terakhir, penelitian mengenai sistem koreksi kata kunci dan pendeteksian kesalahan penulisan telah meningkat, dengan algoritma seperti Jaro-Winkler, N-Gram, Jaccard Similarity, dan Damerau-Levenshtein. Penelitian pertama menunjukkan bahwa Jaro-Winkler mampu mencapai precision 82% dan recall 100% [1] .Selanjutnya, kombinasi Jaro-Winkler dan N-Gram mencapai akurasi 85.7% dalam prediksi perbaikan kata, yang menekankan pentingnya kuantitas dan kualitas korpus [2]. Selain itu, penggunaan Jaccard Index dan fitur N-Gram menunjukkan bahwa Bi-Gram lebih efektif dibandingkan Tri-Gram dan Quad-Gram dalam meningkatkan performa koreksi kata [5]. Namun, penelitian dengan Cosine Similarity dan Jaccard Similarity masih kurang memuaskan dalam penilaian otomatis jawaban pendek, menunjukkan perlunya metode yang lebih canggih [6]. Terakhir, algoritma Damerau-Levenshtein menunjukkan sensitivitas 85% dan *coverage* 74% dalam mendeteksi kesalahan ejaan, khususnya kesalahan transposisi [7]. Meskipun algoritma-algoritma ini menunjukkan hasil yang menjanjikan, tantangan tetap ada dalam menangani kesalahan yang lebih kompleks dan variasi semantik, sehingga penelitian lanjutan harus difokuskan pada pengembangan algoritma yang lebih efisien serta penggunaan kamus yang lebih lengkap untuk meningkatkan akurasi.

Kombinasi algoritma Jaro-Winkler Distance dan Jaccard Similarity menawarkan solusi yang diharapkan lebih baik dalam mendeteksi dan memperbaiki kesalahan penulisan. Jaro-Winkler sangat sensitif terhadap perbedaan kecil dalam urutan karakter, terutama pada bagian awal kata, sehingga cocok untuk mendeteksi kesalahan ketik yang melibatkan urutan huruf [2]. Sementara itu, Jaccard Similarity lebih unggul dalam mengukur kemiripan berdasarkan himpunan token, yang menjadikannya efektif untuk mengenali variasi yang lebih besar dalam ejaan atau struktur kata [5].

Namun, kedua algoritma ini memiliki kekurangan masing-masing. Jaro-Winkler cenderung kurang efektif dalam menangani kesalahan yang terjadi di bagian tengah atau akhir kata, karena algoritma ini lebih fokus pada awal kata. Selain itu, ia kurang responsif terhadap perubahan yang lebih kompleks pada kata atau frasa [8]. Di sisi lain, Jaccard Similarity, meskipun kuat dalam mengukur kesamaan berdasarkan token, dapat kurang akurat ketika berhadapan dengan kata-kata pendek atau yang memiliki sedikit tumpang tindih dalam n-gram [9].

Namun Kombinasi kedua algoritma ini diharapkan dapat menutupi sebagian kekurangan masing-masing. Sensitivitas Jaro-Winkler terhadap perbedaan di awal kata dapat membantu menyeimbangkan kelemahan Jaccard pada kata-kata pendek, sementara kemampuan Jaccard untuk mengukur kemiripan lebih holistik berdasarkan himpunan karakter dapat mengatasi keterbatasan Jaro-Winkler pada bagian tengah atau akhir kata [1]. Meskipun demikian, tidak semua kesalahan dapat diperbaiki secara sempurna dengan kombinasi ini, terutama jika kesalahan melibatkan perubahan struktural yang lebih besar atau konteks linguistik yang rumit [1].

Penelitian ini bertujuan untuk mengevaluasi performa algoritma Jaro-Winkler dan Jaccard Similarity dalam mendeteksi dan mengoreksi kesalahan penulisan pada teks berbahasa Indonesia. Jaro-Winkler efektif dalam mengukur kemiripan antara dua string dengan bobot lebih tinggi pada kesamaan di bagian awal kata [2]. sedangkan Jaccard Similarity menilai kesamaan berdasarkan himpunan karakter atau n-gram [5]. Kedua algoritma diuji melalui skenario yang mensimulasikan kesalahan penulisan umum, menggunakan data uji yang dimodifikasi untuk menilai kinerja model. Hasil evaluasi diukur dengan metrik seperti akurasi, yang menunjukkan kemampuan algoritma dalam mendeteksi kesalahan dan memberikan koreksi yang tepat [2]. Penelitian ini diharapkan dapat memberikan panduan untuk memilih skenario pengembangan model terbaik dalam sistem koreksi otomatis, serta menjadi dasar untuk pengembangan sistem yang lebih canggih dan sesuai kebutuhan pengguna dalam dunia nyata seperti aplikasi perbaikan kesalahan kata yang mudah digunakan .Dengan demikian, penelitian ini tidak hanya fokus pada performa teknis algoritma, tetapi juga menawarkan solusi praktis untuk meningkatkan efisiensi dan akurasi dalam penulisan teks.

2. TINJAUAN PUSTAKA

Terdapat penelitian terdahulu yang menggunakan jaro-winkler untuk memproses data teks sebagai berikut.

Penelitian[1] sebelumnya terkait pengoreksian kata kunci menggunakan algoritma Jaro-Winkler menunjukkan bahwa algoritma ini efektif dalam menangani kesalahan ejaan pada pencarian artikel. Hasil pengujian menunjukkan precision yang bervariasi antara 61% hingga 82%, tergantung jenis kesalahan, sementara recall mencapai 100%, yang menandakan

kemampuannya dalam menemukan semua kandidat yang relevan. Penelitian[2] yang mengkombinasikan Jaro-Winkler Distance dan N-Gram dalam perbaikan kata bahasa Indonesia menghasilkan akurasi yang bervariasi antara 45% hingga 85,7%. Sistem ini mampu mendeteksi dan memperbaiki kesalahan penulisan, tetapi kinerjanya dipengaruhi oleh ukuran dan kualitas korpus yang digunakan. Kombinasi algoritma ini efektif, namun keterbatasan korpus dapat memengaruhi hasil deteksi kesalahan. Penelitian [10] Levenshtein Distance terbukti paling unggul di antara Edit Distance, Hamming Distance, dan Jaccard Similarity dalam perbaikan ejaan. Algoritma ini memiliki waktu MAP terbaik (13,125 ms) untuk penghapusan, penggantian, penambahan, dan penukaran huruf, serta paling cepat dalam memproses operasi perbaikan.

Penelitian[11] Aplikasi pencarian berbasis Jaro-Winkler efektif dalam mengoreksi kesalahan ketik, terutama pada string pendek. Namun, waktu pencarian meningkat seiring bertambahnya data dan ketidakakuratan karakter. Pengujian menunjukkan sistem berfungsi baik, tapi kepuasan pengguna belum diukur. Disarankan untuk menguji kepuasan pengguna dan mengombinasikan Jaro-Winkler dengan algoritma lain. Penelitian[8] Penerapan algoritma Jaro-Winkler Distance pada fitur autocorrect dan spelling suggestion di BMS TV efektif menangani 49 dari 60 kata dalam naskah bahasa Indonesia. Namun, beberapa kesalahan masih terjadi. Pengembangan lebih lanjut disarankan untuk menambah pembobotan kata dan membatasi korpus pada kata yang sering digunakan. Juga direkomendasikan penyempurnaan Porter Stemmer, pengujian metode lain, serta pengembangan sistem menjadi plugin untuk office dan browser.

Penelitian[6] Penggunaan Cosine Similarity dan Jaccard Similarity untuk penilaian otomatis jawaban pendek kurang memuaskan karena hanya menilai susunan leksikal, bukan makna semantik. Penelitian selanjutnya disarankan untuk menggunakan metode yang menangani aspek semantik. Namun, kedua metode ini masih berguna dalam penilaian berbasis keyword, dengan rekomendasi menggunakan dataset lebih besar dan pertanyaan yang lebih beragam. Penelitian[12] Penerapan metode n-gram dalam pencarian hadis Riwayat Bukhari efektif dalam memperbaiki ejaan dan menampilkan dokumen relevan. Hasil menunjukkan precision rata-rata bervariasi: 55% untuk kekurangan huruf, 100% untuk kelebihan huruf, 75% untuk huruf salah, dan 40% untuk huruf tidak tepat, dengan recall 100% untuk semua jenis kesalahan. Precision menurun seiring bertambahnya kesalahan ketik dan meningkat jika kesalahan berkurang. Penelitian [9] Algoritma Jaccard dengan pendekatan SWR2 efektif dalam mengidentifikasi kesamaan antar judul disertasi. Penghilangan stop word menjadi kunci sukses dalam perhitungannya. Hasil analisis ini dapat digunakan oleh program studi untuk pengambilan keputusan dan menilai kontribusi penelitian. Penelitian[5] ini menyimpulkan bahwa aplikasi yang menggunakan Jaccard index dan fitur N-gram efektif untuk membantu tim editor dalam koreksi kata. Hasil analisis menunjukkan bahwa fitur Bi-gram memberikan nilai kesamaan kata tertinggi dibandingkan dengan Trigram dan Quad-gram, menjadikannya pilihan yang lebih tepat untuk proses koreksi.

Penelitian[13] Penggunaan n-gram dan Jaccard similarity efektif dalam mendeteksi kesamaan dokumen, dengan n-gram kecil menghasilkan similarity lebih tinggi. Algoritma winnowing menunjukkan potensi baik untuk deteksi plagiarisme. Penelitian selanjutnya disarankan untuk mengembangkan metode ini pada karya ilmiah yang diterjemahkan. Penelitian[14] menyimpulkan bahwa metode Extended Jaccard Similarity lebih unggul dibandingkan metode Jaccard Similarity dalam hal pengukuran tingkat kemiripan teks. Dengan tingkat similaritas yang lebih tinggi, metode ini menawarkan akurasi yang lebih baik dalam menganalisis dan membandingkan dokumen, menjadikannya pilihan yang lebih efektif dalam konteks deteksi plagiarisme dan tugas-tugas serupa.

Penelitian-penelitian terkait penggunaan algoritma Damerau-Levenshtein dan N-gram dalam deteksi serta koreksi kesalahan ejaan pada dokumen bahasa Indonesia menunjukkan hasil yang beragam [3] [4] [7]. Algoritma Damerau-Levenshtein memiliki sensitivitas 85% dengan cakupan 74% [7].Sementara metode Dictionary Lookup yang dikombinasikan dengan N-gram dan Levenshtein Distance mencapai presisi hingga 0,97 dan recall 1 untuk kesalahan substitusi [4].Hasil analisis menunjukkan bahwa kinerja sistem sangat bergantung pada kelengkapan kamus dan jenis kesalahan ejaan, dengan recall yang lebih tinggi menunjukkan efektivitas dalam menangkap koreksi yang diharapkan [3].

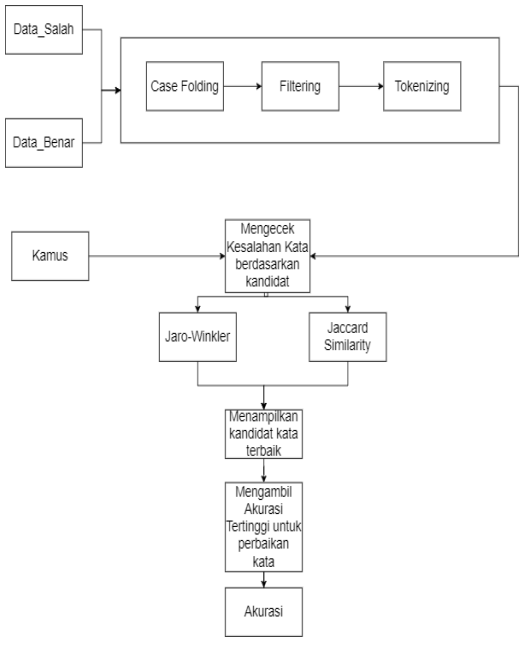
penelitian[15] memberikan dasar yang kuat untuk pengembangan sistem pendeteksi kesamaan dokumen, dengan penekanan pada penggunaan algoritma yang tepat untuk meningkatkan efektivitas dan efisiensi dalam mencegah plagiarisme di bidang akademik. penelitian[16] , dapat disimpulkan bahwa nilai k-gram yang lebih rendah (seperti 2-gram) memberikan hasil kemiripan yang lebih tinggi, sedangkan peningkatan

nilai k-gram cenderung menurunkan kemiripan. Ini menunjukkan bahwa pemilihan parameter dan tahapan preprocessing yang tepat berperan penting dalam mendeteksi plagiarisme dalam dokumen akademik. penelitian[17] bahwa sistem pendeteksi plagiarisme berbasis web yang menggunakan algoritma Rabin-Karp mampu mengidentifikasi tingkat kemiripan dokumen dengan baik. Namun, hasil akurasi yang bervariasi menunjukkan perlunya pengembangan lebih lanjut untuk meningkatkan keandalan deteksi plagiarisme dalam dokumen akademik.

Berdasarkan hasil tinjauan pustaka, Algoritma Jaro-Winkler efektif untuk memperbaiki kesalahan ketik pada kumpulan karakter yang pendek dengan ketepatan tinggi pada ejaan sederhana, sedangkan Jaccard Similarity unggul dalam mendeteksi kesamaan struktur kata pada variasi kesalahan karakter berbasis n-gram. Kombinasi keduanya dianggap optimal, karena menggabungkan kemampuan Jaro-Winkler dalam menangani kesamaan karakter berurutan dan fleksibilitas Jaccard terhadap kesalahan n-gram, yang diharapkan lebih unggul dalam menangani berbagai kesalahan ejaan bahasa Indonesia dibandingkan algoritma lain seperti Damerau-Levenshtein atau Cosine Similarity.

3. METODE PENELITIAN

3.1 Implementasi Model



Gambar 1. Alur Implementasi Model

Model implementasi pada gambar 1 menunjukkan proses koreksi kata yang salah melalui beberapa tahap. Data kata yang salah dan benar diproses menggunakan *case folding*, *filtering*, dan *tokenizing*. Setelah itu, sistem memeriksa kesamaan kata yang salah dengan kandidat dari kamus menggunakan dua algoritma Jaro-Winkler dan Jaccard Similarity. Berdasarkan hasil kedua algoritma ini, sistem menampilkan kandidat kata terbaik, memilih kata yang paling sesuai, dan kemudian menghitung akurasi untuk mengevaluasi performa koreksi kata.

3.2 Data Set

Dataset penelitian ini terdiri dari 10 data yang diambil dari artikel berita online Universitas Mataram sebagai contoh teks yang digunakan, mencakup data benar, data salah, dan referensi kamus KBBI.Untuk perincian data set yang lebih lengkap dapat dilihat pada Tabel 1.

Tabel 1.Rincian Kata Dataset

Data	Jumlah Kata	Jumlah Kata Salah
Dok-1	390	24
Dok -2	321	17
Dok -3	286	14
Dok -4	420	24
Dok -5	562	17
Dok -6	305	20
Dok -7	377	16
Dok -8	416	25
Dok -9	365	23
Dok -10	409	20

a. Data Benar

Data Benar adalah sekumpulan teks yang diambil dari artikel berita online Universitas Mataram, yang sudah benar secara tata bahasa dan ejaan. Data ini digunakan sebagai data uji untuk mengevaluasi kinerja model dalam kondisi normal, sekaligus sebagai referensi untuk mengambil kata-kata yang nantinya akan disalahkan dalam proses evaluasi model.Data benar memiliki jumlah kata yang bervariasi pada tiap datanya yang dapat dilihat pada Tabel 1.

b. Data Salah

Data Salah adalah sekumpulan teks yang diolah dari Data Benar, di mana kesalahan tata bahasa atau ejaan sengaja dimasukkan. Data ini dibuat secara sengaja dari Data Benar untuk menguji kemampuan model dalam mendeteksi dan memperbaiki kesalahan kata yang ada pada dokumen tersebut. Pada dataset ini, kesalahan dalam teks dibagi ke dalam beberapa tipe untuk membantu proses pendeteksian dan perbaikan kesalahan kata. Kesalahan teks dalam dataset ini dibagi menjadi beberapa tipe: *Penghapusan Karakter* (contoh: *hubngan* menjadi *hubungan*), *Penambahan Karakter* (contoh: *ddappat* menjadi *dapat*), *Transposisi* (contoh: *pealksaan* menjadi *pelaksanaan*), *Penghapusan + Penambahan Karakter* (contoh: *dlaam*

menjadi *dalam*), dan *Perubahan Karakter* (contoh: *perahaman* menjadi *pemahaman*).Yang dimana dari semua tipe kesalahan kata telah diterapkan untuk menyalahkan kata dari 10 dokumen yang dipilih secara acak yang jumlahnya dapat dilihat pada Tabel 1.

c. Kamus

Kamus adalah daftar kata-kata yang digunakan sebagai referensi untuk mengecek kesalahan ejaan dan menyediakan kata-kata koreksi dalam teks. Kamus berisi kata-kata yang benar secara tata bahasa dan ejaan yang dimana pada kamus yang berbasiskan file txt ini didapatkan dari website wikipedia yang didalamnya terdiri dari 14.500 lebih kata yang dimananya nanti akan menjadi pemilihan perbaikan kata berdasarkan skenario pengujian yang akan dilakukan.

3.3 Preprocessing Text

a. Case Folding

Pada semua karakter dalam teks diubah menjadi huruf kecil. Ini berguna untuk memastikan konsistensi, sehingga kata-kata yang sama dalam kapitalisasi yang berbeda dianggap sebagai kata yang sama dengan contoh sebagai berikut:

Input : Contoh Text Processing 101 !
Output : contoh text processing 101 !

b. Tokenizing

Teks yang dibagi menjadi unit-unit yang lebih kecil yang disebut "token." Biasanya, token ini adalah kata-kata individu atau tanda baca dengan contoh sebagai berikut:

Input : contoh text processing 101 !
Ouput : ['!', 'contoh', 'text', 'processing', '101']

c. Filtering

Setelah tokenisasi, hanya token yang berisi huruf alfabet (A-Z) yang disimpan. Token yang mengandung angka atau tanda baca dihilangkan dengan contoh sebagai berikut:

Input : ['!', 'contoh', 'text', 'processing', '101']
Output : ['contoh', 'text', 'processing']

3.4 Jaro-Winkler Distance

Algoritma Jaro-Winkler Distance digunakan untuk menilai seberapa mirip dua string atau kata. Ini adalah varian dari Jaro Distance, yang dikembangkan oleh William E. Winkler dan Thibaudeau, dengan penekanan lebih pada awalan yang cocok di kedua string. Algoritma ini menghitung tingkat kemiripan antara dua string, di mana nilai yang lebih tinggi menunjukkan kemiripan yang lebih besar. Jaro-Winkler Distance sangat efektif untuk mengukur kesamaan kata pada string yang lebih pendek dan sering kali lebih cepat dibandingkan dengan algoritma pengukuran kesamaan lainnya.

a. Hitung Panjang String
b. Temukan jumlah karakter yang sama dalam dua string
c. Temukan jumlah transposisi

Untuk menghitung jarak kedekatan antara dua string, algoritma jaro-winkler distancce menggunakan rumus sebagai berikut:

$$d_j = \frac{1}{3} \times \left(\frac{m}{s_1} + \frac{m}{s_2} + \frac{m-t}{m}\right) \dots (1)$$

Dimana :

$m = Jumlah\ karakter\ huruf\ yang\ sama\ persis$
$|s_1| = Panjang\ string\ kata1$
$|s_2| = panjang\ string\ kata2$
$t = Jumlah\ Tranporsisi$

Jika jarak teoritis antara dua karakter yang sama,dapat dibenarkan bila tidak melebihi:

$$\left(\frac{\max(|s_1|, |s_2|)}{2}\right) - 1 \dots (2)$$

Algoritma Jaro-Winkler Distance memiliki nilai maksimum 1, yang menunjukkan bahwa dua string yang dibandingkan sangat mirip atau identik dengan kemiripan 100%. Algoritma ini biasanya menggunakan string pertama, s_1 sebagai referensi untuk mencari transposisi, yaitu situasi di mana karakter yang sama muncul di kedua string, tetapi urutannya berbeda. Dalam Jaro-Winkler Distance, terdapat skala prefiks (p) dengan nilai tetap 0,1, serta panjang prefiks (l) yang mengacu pada jumlah karakter awal yang cocok hingga ditemukan ketidakcocokan, dengan maksimal 4 karakter yang dihitung. Rumus ini digunakan untuk menghitung jarak Jaro-Winkler (dw).

$$d_w = d_j \times p\left(1 - d_j\right) \dots (3)$$

dimana:
Tentu, berikut teks lengkap dengan rumus yang sama persis:
d_j = Jaro Distance S_1 dan S_2
l = panjang prefix / panjang karakter yang sama sebelum ditemukannya ketidaksamaan dengan nilai maksimal 4 karakter.
p = konstanta scalling factor, dengan nilai standar 0.1

3.5 Jaccard Similarity

Jaccard Similarity, atau dikenal juga sebagai Jaccard Coefficient, adalah sebuah algoritma yang digunakan untuk membandingkan dua dokumen dengan menganalisis kesamaan kata yang ada di antara keduanya. Algoritma ini biasanya diterapkan

untuk menilai kemiripan antara dua objek atau dokumen dengan menghitung nilai kesamaan yang dimilikinya. Rumus untuk menghitung Jaccard Similarity dapat ditemukan dalam persamaan terkait

$$\text{Similarity}(X, Y) = \frac{|X \cap Y|}{|X \cup Y|} \dots (4)$$

merupakan dari rumus *Jaccard Similarity* yang digunakan untuk mencari persamaan dan perbedaan pada dua sampel.

3.6 Skenario Pengujian

Pada pengujian kesalahan kata, digunakan dua pendekatan skenario evaluasi, yaitu 1-Terbaik dan 5-Terbaik Pendekatan ini seperti yang dilakukan pada penelitian [7]. Beberapa skenario pengujian yang akan dilakukan adalah sebagai berikut. Pada pendekatan 1-Terbaik, sistem memilih satu kandidat teratas sebagai hasil koreksi, dan kandidat tersebut digunakan untuk perbaikan kesalahan kata. Sedangkan dalam pendekatan 5-Terbaik, jika kata yang benar terdapat di antara lima kandidat teratas, maka sistem menganggapnya sebagai koreksi yang benar. Pendekatan ini seperti yang dilakukan pada penelitian[7]. Beberapa skenario pengujian yang akan dilakukan adalah sebagai berikut:

a. Menghitung akurasi ketepatan perbaikan menggunakan algoritma Jaro-Winkler Distance & Jaccard Similarity pendekatan 1-Terbaik & 5-Terbaik

Pengujian skenario ini bertujuan mengevaluasi akurasi perbaikan kata menggunakan algoritma Jaro-Winkler Distance dan Jaccard Similarity. Proses pengujian dimulai dengan menyediakan kumpulan data kata yang benar dan salah, kemudian menghitung akurasi masing-masing pendekatan dengan membandingkan hasil algoritma dengan kata yang benar. Hasil pengujian diharapkan memberikan wawasan mengenai efektivitas kedua algoritma dalam meningkatkan akurasi perbaikan kata.

b. Menghitung akurasi ketepatan perbaikan menggunakan kombinasi jaro-winkler & Jaccard-Similarity 1-Terbaik & 5-Terbaik

Pengujian ini bertujuan untuk mengevaluasi akurasi perbaikan kata menggunakan kombinasi algoritma Jaro-Winkler Distance dan Jaccard Similarity. Dengan menggabungkan kedua algoritma, diharapkan akurasi perbaikan dapat meningkat dibandingkan dengan pengujian sebelumnya yang hanya menggunakan satu algoritma. Proses ini melibatkan pengujian terhadap kumpulan data yang terdiri dari kata-kata benar dan salah, serta menghitung akurasi berdasarkan hasil kombinasi algoritma. Hasil pengujian diharapkan bisa meningkatkan akurasi perbaikan kata.

c. Menghitung akurasi ketepatan perbaikan kata menggunakan kombinasi Algoritma Jaro-Winkler Distance & Jaccard Similarity dengan pembobotan nilai menggunakan pendekatan 1-Terbaik & 5-Terbaik

Pengujian ini bertujuan mengevaluasi akurasi perbaikan kata dengan mengombinasikan algoritma Jaro-Winkler Distance dan Jaccard Similarity melalui pembobotan linier. Pembobotan ini berfungsi meningkatkan akurasi dengan mempertimbangkan kesamaan serta pentingnya masing-masing kandidat. Bobot lebih besar diberikan pada algoritma (Jaro atau jaccard) yang lebih efektif dalam menangani kesalahan kata yang melibatkan perubahan urutan karakter, sehingga cocok untuk koreksi ejaan.Hal ini akan didasarkan pada hasil pengujian skenario subab 3.6.a.

d. Menghitung akurasi ketepatan kata menggunakan algoritma Jaro-Winkler Filter Jaccard Similarity 1-Terbaik & 5-Terbaik

Pengujian skenario ini bertujuan untuk menghitung akurasi ketepatan perbaikan kata dengan menggunakan algoritma Jaro-Winkler yang difilter oleh Jaccard Similarity. Dalam pendekatan ini, algoritma Jaro-Winkler pertama-tama menghasilkan kandidat perbaikan untuk setiap kata yang salah. Kemudian, Jaccard Similarity digunakan untuk menyaring kandidat tersebut, memastikan bahwa hanya kandidat yang memiliki kesamaan yang signifikan dengan kata yang benar yang dipertimbangkan.

e. Menghitung akurasi ketepatan kata menggunakan algoritma Jaccard Similarity Filter Jaro-Winkler Distance 1-Terbaik & 5-Terbaik

Pengujian skenario ini bertujuan untuk menghitung akurasi ketepatan perbaikan kata dengan menggunakan algoritma Jaccard Similarity yang difilter oleh Jaro-Winkler Distance. Dalam pendekatan ini, algoritma Jaccard Similarity pertama-tama menghasilkan kandidat perbaikan untuk setiap kata yang salah, berdasarkan kesamaan dengan kata-kata lain dalam kumpulan data. Selanjutnya, Jaro-Winkler Distance digunakan untuk menyaring kandidat tersebut, memastikan bahwa hanya kandidat yang memiliki kesamaan tertinggi dengan kata yang benar yang dipertimbangkan.

3.7 Evaluasi Metrik

Pada pengujian ketepatan kata untuk mendapatkan nilai akurasi dari masing-masing algoritma yang digunakan menggunakan perhitungan sebagai berikut:

$$\text{Akurasi} = \frac{\text{Total Kandidat Yang benar}}{\text{Total Kandidat}} \dots (5)$$

4. HASIL DAN PEMBAHASAN

Pengujian perbaikan kesalahan kata telah dilakukan menggunakan algoritma Jaro-Winkler dan Jaccard Similarity. Berikut ini adalah hasil yang menunjukkan performa kedua algoritma dalam mendeteksi dan memperbaiki kesalahan pada dataset yang digunakan dengan berbagai macam skenario pengujian sebagai berikut:

4.1 Evaluasi Matrik

4.1.1 Pengujian Jaro-Winkler Distance & Jaccard Similarity Pendekatan 1 Terbaik

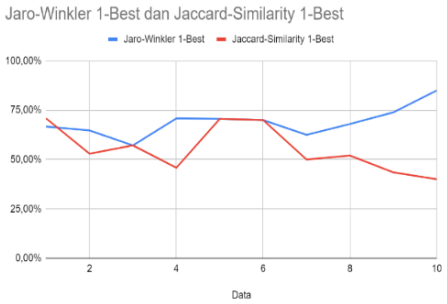
Tabel 2. Jaro-Winkler & Jaccard 1-Terbaik

Data	Jaro-Winkler	Jaccard Similarity
Dok-1	66,67%	70,83%
Dok-2	64,71%	52,94%
Dok-3	57,14%	57,14%
Dok-4	70,83%	45,83%
Dok-5	70,59%	70,59%
Dok-6	70,00%	70,00%
Dok-7	62,50%	50,00%
Dok-8	68,00%	52,00%
Dok-9	73,91%	43,48%
Dok-10	85%	40%
Rata-Rata	68,94%	55,25%

Berdasarkan Tabel 2 menunjukkan perbandingan bahwa akurasi antara dua algoritma koreksi kesalahan kata, yaitu Jaro-Winkler dan Jaccard Similarity, dengan pendekatan 1-Terbaik Dalam pendekatan ini, akurasi dihitung berdasarkan apakah kandidat pertama yang dihasilkan oleh masing-masing algoritma sesuai dengan kata yang benar.

Tabel 2 menunjukkan akurasi dua algoritma koreksi kata, Jaro-Winkler dan Jaccard Similarity, dalam skenario 1-Terbaik pada 10 dokumen. Pada dokumen pertama, Jaccard lebih akurat (70,83%) dibandingkan Jaro-Winkler (66,67%). Namun, secara keseluruhan, Jaro-Winkler memiliki akurasi rata-rata lebih tinggi, yaitu 68,94%, dibandingkan Jaccard yang hanya 55,28%. Perbedaan ini disebabkan oleh cara kerja kedua algoritma, di mana Jaro-Winkler lebih baik menangani kesalahan ketik perbedaan karakter, sementara Jaccard kurang efektif jika perbedaan karakter.

Berdasarkan Gambar 2 menunjukan bahwa grafik tersebut memiliki perbandingan akurasi antara algoritma Jaro-Winkler dan Jaccard Similarity dalam skenario 1-Terbaik pada 10 dokumen. Secara keseluruhan, Jaro-Winkler memiliki performa yang lebih stabil dan akurat, dengan akurasi yang lebih tinggi pada sebagian besar dokumen. Sementara itu, Jaccard-Similarity cenderung menunjukkan ketikdaktetapan yang lebih tajam dan akurasi yang lebih rendah dibandingkan Jaro-Winkler.



Gambar 2. Jaro-Winkler & Jaccard 1-Terbaik

4.1.2 Pengujian Jaro-Winkler Distance & Jaccard Similarity Pendekatan 5 Terbaik

Tabel 3. Jaro-Winkler & Jaccard 5-Terbaik

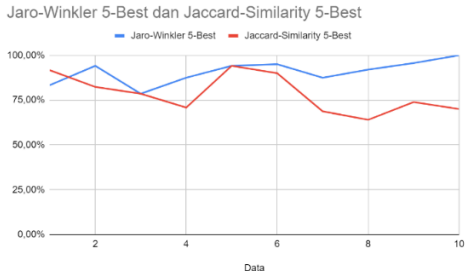
Data	Jaro-Winkler	Jaccard Similarity
Dok-1	83,33%	91,67%
Dok-2	94,12%	82,35%
Dok-3	78,57%	78,57%
Dok-4	87,50%	70,83%
Dok-5	94,12%	94,12%
Dok-6	95,00%	90,00%
Dok-7	87,50%	68,75%
Dok-8	92,00%	64,00%
Dok-9	95,65%	73,91%
Dok-10	100%	70,00%
Rata-Rata	90,78%	78,42%

Berdasarkan Tabel 3 tersebut menunjukkan akurasi algoritma Jaro-Winkler dan Jaccard Similarity dalam skenario 5-Terbaik, di mana kata yang benar dianggap teridentifikasi jika muncul di antara 5 kandidat teratas. Pada 10 dokumen yang diuji, Jaro-Winkler konsisten menunjukkan akurasi yang lebih tinggi dengan rata-rata 90,78%, sementara Jaccard memiliki rata-rata akurasi 78,42%.

Dibandingkan dengan skenario 1-Terbaik sebelumnya, terdapat peningkatan signifikan pada kedua algoritma. Pada pendekatan 5-best, Jaccard Similarity meningkat dari 55,28**%** menjadi 78,42**%**, sedangkan Jaro-Winkler juga mengalami peningkatan dari 68,94**%** menjadi 90,78**%**. Peningkatan ini menunjukkan bahwa kedua algoritma lebih efektif dalam mengidentifikasi kata yang benar saat diberi kesempatan memilih dari 5 kandidat teratas.

Berdasarkan Gambar 3 menunjukan bahwa grafik tersebut memiliki performa algoritma Jaro-Winkler dan Jaccard Similarity dalam koreksi kesalahan kata

dengan pendekatan 5-Terbaik. Jaro-Winkler cenderung memiliki performa lebih konsisten dan tinggi dibandingkan Jaccard Similarity yang menunjukkan perubahan ketikdaktetapan yang lebih besar dengan penurunan pada beberapa titik. Secara keseluruhan, Jaro-Winkler lebih unggul dalam menangani data uji.



Gambar 3. Jaro-Winkler & Jaccard 5-Terbaik

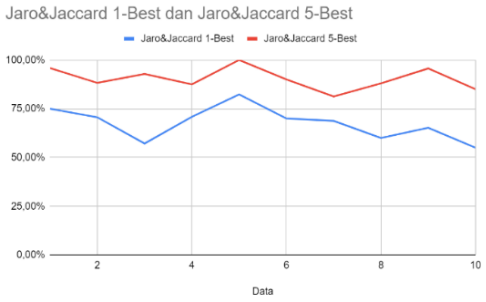
4.1.3 Pengujian Kombinasi Jaro-Winkler Distance & Jaccard Similarity Pendekatan 1 & 5 Terbaik

Tabel 4. Kombinasi Jaro-Winkler Distance & Jaccard Similarity Pendekatan 1 & 5 Terbaik

Data	Kombinasi 1-Terbaik	Kombinasi 5-Terbaik
Dok-1	75,00%	95,83%
Dok-2	70,59%	88,24%
Dok-3	57,14%	92,86%
Dok-4	70,83%	87,50%
Dok-5	82,35%	100%
Dok-6	70,00%	90,00%
Dok-7	68,75%	81,25%
Dok-8	60,00%	88,00%
Dok-9	65,22%	95,65%
Dok-10	55,00%	85,00%
Rata-Rata	67,49%	90,43%

Berdasarkan Tabel 4 menunjukkan hasil uji perbaikan kesalahan kata menggunakan kombinasi algoritma Jaro-Winkler dan Jaccard Similarity dengan dua pendekatan 1-Terbaik dan 5-Terbaik. Pada pendekatan 1-Terbaik, rata-rata akurasi hanya mencapai 67,49%, sementara pada pendekatan 5-Terbaik, akurasi meningkat signifikan hingga 90,43%. Hal ini menunjukkan bahwa pendekatan 5-Terbaik, yang mempertimbangkan 5 pilihan teratas dalam koreksi kata, lebih efektif dalam menangani kesalahan kata dibandingkan hanya menggunakan satu pilihan teratas.

Berdasarkan Gambar 4 menunjukan bahwa grafik menunjukkan perbandingan performa kombinasi algoritma Jaro-Winkler dan Jaccard Similarity dengan pendekatan 1-Terbaik dan 5-Terbaik.Pada pendekatan 5-Terbaik dengan secara konsisten menunjukkan akurasi yang lebih tinggi, mendekati atau mencapai 100% pada beberapa titik.Namun sebaliknya, pendekatan 1-Terbaik akurasinya lebih rendah, berkisar di antara 50%-75%. Ini menegaskan bahwa mempertimbangkan lima pilihan teratas dalam koreksi kata 5-Terbaik lebih baik dan efektif dibandingkan hanya menggunakan satu pilihan terbaik.



Gambar 4. Kombinasi Jaro-Winkler Distance & Jaccard Similarity Pendekatan 1 & 5 Terbaik

4.1.4 Pengujian Kombinasi Jaro-Winkler Distance & Jaccard Similarity dengan Pembobotan Nilai Perdeketan 1-Terbaik

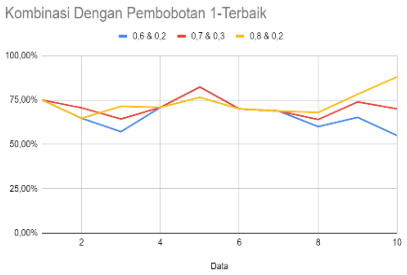
Tabel 5. Kombinasi Jaro-Winkler Distance & Jaccard Similarity dengan pembobotan -1 Terbaik

Data	0,6&0,4	0,7&0,3	0,8&0,2
Dok-1	75,00%	75,00%	75,00%
Dok-2	64,71%	70,59%	64,71%
Dok-3	57,14%	64,29%	71,43%
Dok-4	70,83%	70,83%	70,83%
Dok-5	82,35%	82,35%	76,47%
Dok-6	70,00%	70,00%	70,00%
Dok-7	68,75%	68,75%	68,75%
Dok-8	60,00%	64,00%	68,00%
Dok-9	65,22%	73,91%	78,26%
Dok-10	55,00%	70,00%	88,00%
Rata-Rata	66,90%	70,97%	73,15%

Berdasarkan tabel 5 hasil akurasi perbaikan kata menggunakan algoritma Jaro-Winkler dan Jaccard Similarity dengan pembobotan tertentu, terlihat adanya variasi akurasi di setiap dokumen dan kombinasi bobot. Pada pembobotan 0,6 & 0,4, akurasi rata-rata yang dihasilkan adalah 66,90%, sedangkan pada pembobotan 0,7 & 0,3 akurasi meningkat menjadi 70,97%. Pembobotan tertinggi pada Jaro-Winkler, yaitu 0,8 & 0,4, menghasilkan rata-rata akurasi yang tertinggi, yaitu 73,15%.

Secara keseluruhan, terlihat bahwa Peningkatan bobot Jaro-Winkler dari 0,6 ke 0,8 berdampak positif pada akurasi perbaikan kata di semua dokumen,

menunjukkan dominasi Jaro-Winkler dibanding Jaccard Similarity dalam meningkatkan akurasi. Akurasi tertinggi dicapai di Dok-10 (88% pada bobot 0,8 & 0,4), sementara akurasi terendah ada di Dok-10 (55,00% pada bobot 0,6 & 0,4).



Gambar 5.Kombinasi Pembobotan 1-Terbaik

Berdasarkan gambar 5 pada grafik di atas, yang menunjukkan performa akurasi perbaikan kata dengan tiga kombinasi pembobotan (0,6 & 0,2; 0,7 & 0,3; 0,8 & 0,2), terlihat bahwa pembobotan 0,8 & 0,2 secara konsisten menghasilkan performa yang lebih tinggi di sebagian besar data dibandingkan dua kombinasi lainnya. Puncak performa pada bobot 0,8 & 0,2 terjadi pada data ke-10 dengan akurasi mendekati 90%. Sementara itu, kombinasi 0,7 & 0,3 juga menunjukkan performa yang stabil dan lebih tinggi pada beberapa titik, terutama pada data ke-5 dan ke-9. Di sisi lain, pembobotan 0,6 & 0,2 cenderung memiliki akurasi lebih rendah dan ketidakstabilan performa dibandingkan dua kombinasi lainnya. Hal ini mengindikasikan bahwa peningkatan bobot Jaro-Winkler berbanding lurus dengan peningkatan akurasi.

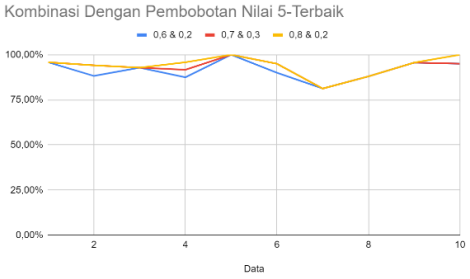
4.1.5 Pengujian Kombinasi Jaro-Winkler Distance & Jaccard Similarity dengan Pembobotan Nilai Perdeketan 5-Terbaik

Tabel 6. Kombinasi Jaro-Winkler Distance & Jaccard Similarity dengan pembobotan nilai 5-Terbaik

Data	0,6&0,4	0,7&0,3	0,8&0,2
Dok-1	95,83%	95,83%	95,83%
Dok-2	88,24%	94,12%	94,12%
Dok-3	92,86%	92,86%	92,86%
Dok-4	87,50%	91,67%	95,83%
Dok-5	100%	100%%	100%
Dok-6	90,00%	95,00%	95,00%
Dok-7	81,25%	81,25%	81,25%
Dok-8	88,00%	88,00%	88,00%
Dok-9	95,65%	95,65%	95,65%
Dok-10	95,00%	95,00%	100%
Rata-Rata	91,43%	92,94%	93,85%

Berdasarkan tabel 6 dapat dilihat bahwa akurasi perbaikan kata menggunakan Jaro-Winkler dan Jaccard Similarity dengan pendekatan 5-best, rata-rata akurasi untuk setiap pembobotan lebih tinggi dibandingkan dengan pendekatan 1-Terbaik pada tabel 5. Pada pembobotan 0,6 & 0,4, akurasi rata-rata mencapai 91,43%, sedangkan pada pendekatan 1-Terbaik hanya 66,90%. Peningkatan yang sama juga terjadi pada pembobotan 0,7 & 0,3 (dari 70,97% menjadi 92,94%) dan pada pembobotan 0,8 & 0,4 (dari 73,15% menjadi 93,85%).

Secara umum, pendekatan 5-best memberikan peningkatan akurasi yang signifikan, dengan performa tertinggi terlihat pada Dok-5 yang mencapai 100% di semua kombinasi bobot. Pendekatan ini menunjukkan bahwa menggunakan lebih banyak kandidat kata yang mirip (5-Terbaik) dibandingkan hanya (1-Terbaik) memberikan hasil perbaikan kata yang mendekati kemiripan dari kata yang salah.



Gambar 6. Kombinasi Pembobotan 5-Terbaik

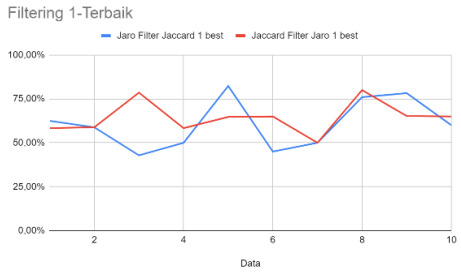
Berdasarkan Gambar 6 menunjukan grafik pengujian 5-terbaik, terlihat adanya peningkatan performa yang signifikan dibandingkan dengan pengujian 1-terbaik. Pada pengujian 5-terbaik, nilai akurasi cenderung lebih stabil dan konsisten mendekati 100%, yang menunjukkan bahwa dengan mempertimbangkan 5 kandidat terbaik, kemungkinan menemukan kata yang benar menjadi lebih tinggi. Begitu juga sebaliknya, pada pengujian 1-terbaik, performa terlihat lebih tidak stabil dengan beberapa titik yang memiliki akurasi lebih rendah. Hal ini menunjukkan bahwa dengan hanya mempertimbangkan 1 kandidat terbaik, kemungkinan untuk salah dalam memilih kata yang benar lebih tinggi, karena kandidat terbaik yang dipilih mungkin bukan yang benar secara keseluruhan. Kombinasi pembobotan juga berkontribusi pada peningkatan performa, namun pengaruh terbesar terlihat pada peningkatan jumlah kandidat yang dipertimbangkan dalam pengujian 5-terbaik, yang secara umum menghasilkan hasil yang lebih baik dibandingkan 1-terbaik.

4.1.6 Pengujian Jaro-Winkler Filter Jaccard Similarity & Jaccard Similarity Filter Jaro-Winkler 1-Terbaik

Tabel 7. Jaro-Winkler Filter Jaccard Similarity & Jaccard Similarity Filter Jaro-Winkler 1-Terbaik

Data	Jaro-Winkler Filter Jaccard Similarity	Jaccard Similarity Filter Jaro-Winkler
Dok-1	62,50%	58,33%
Dok-2	58,82%	58,82%
Dok-3	42,86%	50,00%
Dok-4	50,00%	58,33%
Dok-5	82,35%	64,71%
Dok-6	45,00%	65,00%
Dok-7	50,00%	50,00%
Dok-8	76,00%	80,00%
Dok-9	78,26%	65,22%
Dok-10	60,00%	65,00%
Rata-Rata	60,58%	61,54%

Berdasarkan Tabel 7 menunjukkan bahwa perbandingan akurasi antara dua metode, yaitu Jaro-Winkler Filter Jaccard Similarity dan Jaccard Similarity Filter Jaro-Winkler, yang diuji pada 10 dokumen. Hasilnya menunjukkan bahwa algoritma Jaccard Similarity Filter Jaro-Winkler memiliki rata-rata akurasi yang lebih tinggi, yaitu 61,54%, dibandingkan dengan algoritma Jaro-Winkler Filter Jaccard Similarity yang memiliki rata-rata akurasi 60,58%. Secara individual, beberapa dokumen seperti Dok-3 dan Dok-6 menunjukkan peningkatan akurasi yang signifikan dengan metode Jaccard dibandingkan Jaro-Winkler, sementara pada beberapa dokumen lainnya seperti Dok-1 dan Dok-5, metode Jaro-Winkler memberikan hasil yang lebih baik. Ini mengindikasikan bahwa filter yang digunakan pada kedua metode memiliki pengaruh yang berbeda tergantung pada karakteristik dokumen, namun secara keseluruhan metode Jaccard memberikan akurasi yang sedikit lebih tinggi.



Gambar 7 Jaro-Winkler Filter Jaccard Similarity & Jaccard Similarity Filter Jaro-Winkler 1- Terbaik

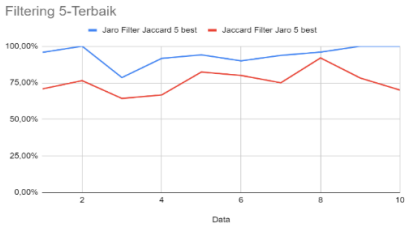
Berdasarkan Gambar 7 menunjukkan pada grafik perbandingan performa antara dua metode filtering, yaitu Jaro Filter dengan Jaccard dan Jaccard Filter dengan Jaro Dari grafik ini, terlihat bahwa performa kedua Algoritma cenderung tidak konsisten di berbagai data yang diuji. Pada beberapa titik, metode Jaro Filter Jaccard memiliki performa yang lebih baik, sedangkan pada titik lain, Jaccard Filter Jaro menunjukkan hasil yang lebih unggul. Secara keseluruhan, kedua metode memiliki hasil yang cukup seimbang tanpa perbedaan yang signifikan di antara keduanya.

4.1.7 Pengujian Jaccard Similarity Filter Jaro-Winkler Distance Pendekatan 1 & 5 Terbaik

Tabel 8. Jaro-Winkler Filter Jaccard Similarity & Jaccard Similarity Filter Jaro-Winkler 5- Terbaik

Data	Jaro-Winkler Filter Jaccard Similarity	Jaccard Similarity Filter Jaro-Winkler
Dok-1	95,83%	70,83%
Dok-2	100%	76,47%
Dok-3	78,57%	64,29%
Dok-4	91,67%	66,67%
Dok-5	94,12%	82,35%
Dok-6	90,00%	80,00%
Dok-7	93,75%	75,00%
Dok-8	86,00%	92,00%
Dok-9	100%	78,26%
Dok-10	100%	70,00%
Rata-Rata	93,99%	75,59%

Berdasarkan Tabel 8 menampilkan hasil pengujian dua algoritma pendekatan, yaitu Jaro Filter Jaccard 5 terbaik dan Jaccard Filter Jaro 5 terbaik, dalam mendeteksi kesalahan kata. Pada setiap data yang diuji, metode Jaro Filter Jaccard menunjukkan performa yang lebih baik dengan persentase keberhasilan yang lebih tinggi, memiliki rata-rata 93,99%. Sementara itu, algoritma Jaccard Filter Jaro menunjukkan performa yang lebih rendah dengan rata-rata 75,59%. Setiap data menunjukkan bahwa Jaro Filter Jaccard selalu unggul dibandingkan Jaccard Filter Jaro, terutama pada data ke-2, ke-9, dan ke-10, di mana perbedaannya cukup signifikan. Secara keseluruhan, dapat disimpulkan bahwa Jaro Filter Jaccard lebih efektif dan konsisten dalam menangani pengujian kesalahan kata dibandingkan Jaccard Filter Jaro.



Gambar 8. Jaro-Winkler Filter Jaccard Similarity & Jaccard Similarity Filter Jaro-Winkler 5- Terbaik

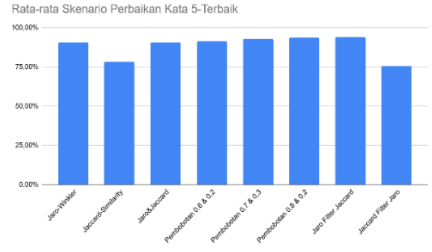
Berdasarkan Gambar 8 pada grafik di atas menunjukkan perbandingan performa antara Jaro

Filter Jaccard 5-Terbaik dan Jaccard Filter Jaro 5-Terbaik berdasarkan hasil pengujian pada beberapa data. Secara signifikan, algoritma Jaro Filter Jaccard 5 best memiliki performa yang jauh lebih baik dan konsisten dibandingkan Jaccard Filter Jaro 5 best Jaro Filter Jaccard menunjukkan hasil yang stabil dan tinggi, dengan kisaran performa antara 90% hingga 100% di hampir semua data. Sebaliknya, Jaccard Filter Jaro memiliki naik turun yang lebih besar dengan performa yang lebih rendah, berkisar antara 64% hingga 82%. Perbedaan signifikan ini menunjukkan bahwa Jaro Filter Jaccard 5-Terbaik secara keseluruhan lebih efektif dibandingkan Jaccard Filter Jaro 5-Terbaik dalam mengatasi pengujian kesalahan kata dengan pendekatan 5 terbaik.

4.2 Rangkuman Skenario Pengujian Kesalahan Kata 1-Terbaik & 5 Terbaik

Gambar 9.Performa Skenario 1-Terbaik

Berdasarkan Gambar 9 Grafik ini menunjukkan performa beberapa skenario perbaikan kesalahan kata dengan pendekatan 1-terbaik menggunakan algoritma Jaro-Winkler dan Jaccard Similarity. Jaro-Winkler mencapai akurasi tertinggi sekitar 65%, sementara Jaccard Similarity lebih rendah di sekitar 55%. Kombinasi keduanya dengan pembobotan 0.8 & 0.2 memberikan hasil terbaik, mendekati 70%, menunjukkan bahwa menekankan bobot pada Jaro-Winkler meningkatkan akurasi. Skenario dengan filter memiliki performa lebih rendah dan tidak melebihi kombinasi pembobotan.



Gambar 10.Performa Skenario 5-Terbaik

Berdasarkan Gambar 10 menunjukan bahwa pada grafik memiliki performa rata-rata dari beberapa skenario perbaikan kesalahan kata menggunakan pendekatan 5-terbaik dengan algoritma Jaro-Winkler dan Jaccard Similarity. Pada pendekatan ini, algoritma Jaro-Winkler menunjukkan performa yang tinggi, mendekati 90%, sedangkan Jaccard Similarity sedikit lebih rendah, sekitar 75%. Kombinasi kedua algoritma dengan pembobotan tertentu, terutama pembobotan 0.8 & 0.2 dan pembobotan 0.7 & 0.3, mencapai performa yang mendekati atau bahkan melebihi Jaro-Winkler. Hal ini menunjukkan bahwa memberikan bobot lebih besar pada Jaro-Winkler efektif dalam meningkatkan akurasi pada pendekatan 5-terbaik. Skenario dengan filter juga menunjukkan performa yang baik, meskipun tidak melebihi kombinasi pembobotan, dengan Jaro Filter Jaccard mendekati 85%, sedangkan Jaccard Filter Jaro sedikit lebih rendah.

5. KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian yang telah dilakukan, algoritma Jaro-Winkler dan Jaccard Similarity masing-masing memiliki kelebihan dan kekurangannya. Jaro-Winkler unggul dalam menangani kesalahan kata yang kompleks, seperti transposisi huruf atau penghapusan beberapa karakter, sehingga sangat efektif untuk memperbaiki kesalahan yang lebih parah. Di sisi lain, Jaccard Similarity memperkuat proses pemilihan kandidat dengan mengidentifikasi karakter-karakter yang sama antara kata yang salah dan kata yang benar, sehingga menghasilkan kandidat koreksi yang lebih presisi setelah melalui proses filter dari Jaro-Winkler. Faktor lain yang memengaruhi akurasi adalah variasi tingkat kesalahan kata pada setiap dokumen yang diuji, di mana setiap tipe kesalahan memiliki karakteristik yang berbeda-beda. Hal ini menyebabkan algoritma Jaro-Winkler dan Jaccard Similarity memiliki keunggulan masing-masing dalam menangani jenis kesalahan tertentu. Pada pendekatan 1-Terbaik, kombinasi algoritma dengan pembobotan 0,8 untuk Jaro-Winkler dan 0,2 untuk Jaccard menghasilkan rata-rata akurasi tertinggi sebesar 73,15%, menunjukkan bahwa prioritas yang lebih tinggi pada Jaro-Winkler memberikan hasil yang lebih akurat. Sementara itu, pada pendekatan 5-Terbaik, skenario yang menggabungkan Jaccard Similarity sebagai filter untuk kandidat yang dipilih oleh Jaro-Winkler mencapai akurasi tertinggi sebesar 93,99%. Hal ini menunjukkan bahwa Jaccard mampu lebih efektif memfilter dan menyempurnakan kandidat yang mendekati benar, setelah melalui proses awal yang dilakukan oleh Jaro-Winkler.

Berdasarkan penelitian yang telah dilakukan, terdapat beberapa saran untuk Pengembangan penelitian ini dapat dilakukan dengan memperluas skenario pengujian melalui variasi data, jenis kesalahan yang lebih luas dan membandingkan skenario

pengujian dengan metode lainnya yang dimana bisa menjadikan acuan yang lebih kuat.Kombinasi algoritma Jaro-Winkler dan Jaccard Similarity juga disarankan untuk diuji pada berbagai bahasa untuk memahami bagaimana algoritma menangani pola kesalahan dalam konteks yang berbeda. Penggunaan kamus yang lebih luas dapat membantu algoritma mengatasi variasi bentuk kata yang lebih beragam. Selain itu, penting untuk mengeksplorasi bobot optimal antara kedua algoritma dan menguji pendekatan 1-Terbaik dan 5-Terbaik untuk mendapatkan gambaran yang mencakup mengenai keunggulan algoritma dalam menangani berbagai jenis kesalahan.

DAFTAR PUSTAKA

[1] A. Retno *et al.*, "Sistem Koreksi Kesalahan Pengetikan Kata Kunci dalam Pencarian Artikel Menggunakan Algoritma Jaro-Winkler, *Seminar Informatika Aplikatif Polinema. 2019. p. 60-65."*

[2] N. C. Dewi and A. Qoiriah, "Implementasi Algoritma Jaro-Winkler Distance dan N-gram untuk Deteksi dan Prediksi Perbaikan Kesalahan Penulisan Kata Bahasa Indonesia pada Karya Tulis Ilmiah Mahasiswa," *Journal of Informatics and Computer Science*, vol. 02, 2021.

[3] A. I. Fahma, I. Cholissodin, and R. S. Perdana, "Identifikasi Kesalahan Penulisan Kata (Typographical Error) pada Dokumen Berbahasa Indonesia Menggunakan Metode N-gram dan Levenshtein Distance," 2018. [Online]. Available: http://j-ptiik.ub.ac.id

[4] T. N. Maghfira, I. Cholissodin, and A. W. Widodo, "Deteksi Kesalahan Ejaan dan Penentuan Rekomendasi Koreksi Kata yang Tepat Pada Dokumen Jurnal JTIIK Menggunakan Dictionary Lookup dan Damerau-Levenshtein Distance," 2017. [Online]. Available: http://j-ptiik.ub.ac.id

[5] A. Indriani *et al.*, "Implementasi Jaccard Index Dan N-Gram Pada Rekayasa Aplikasi Koreksi Kata Berbahasa Indonesia," *Sebatik*, pp. 95–101, 2018, Accessed: Oct. 16, 2024. [Online]. Available: https://www.neliti.com/id/publications/268949/implementasi-jaccard-index-dan-n-gram-pada-rekayasa-aplikasi-koreksi-kata-berbah

[6] U. Hasanah and D. A. Mutiara, "Perbandingan Metode Cosine Similarity Dan Jaccard Similarity Untuk Penilaian Otomatis Jawaban Pendek," *SENSITIf: Seminar Nasional Sistem Informasi dan Teknologi Informasi*, pp. 1255–1263, 2019.

[7] D. Anggreni, R. Sari, B. Irmawati, and R. Dwiyansaputra, "Spelling Error Correction In Indonesian Using Damaeru-Levenstein Distance Dan N-Gram." Jurnal Teknologi Informasi, *Komputer, dan Aplikasinya (JTIKA)*, 2024, 6.1: 257-263.

[8] A. Prasetyo, W. M. Baihaqi, and I. S. Had, "Algoritma Jaro-Winkler Distance: Fitur Autocorrect dan Spelling Suggestion pada Penulisan Naskah Bahasa Indonesia di BMS TV," *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 5, no. 4, pp. 435–444, Oct. 2018, doi: 10.25126/jtiik.201854780.

[9] L. Mayola, M. Hafizh, and D. M. Putra, "Algoritma Jaccard Similarity untuk Deteksi Kemiripan Judul Disertasi dengan Pendekatan Variasi Stop Word Removal," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 8, no. 1, p. 477, Jan. 2024, doi: 10.30865/mib.v8i1.7109.

[10] S. Rianti and R. A. Supono, "Perbandingan Algoritma Edit Distance, Levenstein Distance, Hamming Distance, Jaccard Similarity Dalam Mendeteksi String Matching," 2023.

[11] V. Sahfitri, I. Batutah Zarizal, U. Bina Darma, J. Jenderal Ahmad Yani No, and P. Sur-el, "Approximate String Matching Untuk Pencarian Kata Dalam Kamus Bahasa Indonesia Menggunakan Algoritma Jaro Winkler," *Jurnal Ilmiah MATRIK*, vol. 24, no. 3, p. 2022.

[12] R. Shalihah, I. P. Ningrum, A. M. Sajiah, and M. I. Sarita, "Penerapan Metode N-Gram Untuk Memperbaiki Kesalahan Penulisan Ejaan Kata Kunci Pada Aplikasi Pencarian Hadis," *semanTIK*, vol. 9, no. 1, p. 73, Jun. 2023, doi: 10.55679/semantik.v9i1.15344.

[13] A. Yudhana, dan Iif Alfiatul Mukaromah, P. Studi Teknik Elektro, U. Ahmad Dahlan, A. Dahlan, and U. Jl Soepomo, "Implementasi Deteksi Plagiatrisme Menggunakan Metode N-Gram Dan Jaccard Similarity Terhadap Algoritma Winnowing." *Transmisi: Jurnal Ilmiah Teknik Elektro*, 2018, 20.3: 105-110.

[14] M. Nugraheni, "Perbandingan Jaccard Similarity Dengan Extended Jaccard Similarity Pada Penalaran Berbasis Kasus." 2020, 4.2: 49-52."

[15] P. P. Putra, A. Afriansyah, and M. Syaifullah, "Pendeteksi Kesamaan Dokumen pada Sistem Informasi Pendaftaran Proposal Skripsi dengan Pendekatan Algoritma Rabin-Karp," *INTECOMS: Journal of Information Technology and Computer Science*, vol. 2, no. 1, pp. 40–47, Jun. 2019, doi: 10.31539/intecoms.v2i1.738.

[16] D. Plagiarisme *et al.*, "Techno Xplore Jurnal Ilmu Komputer dan Teknologi Informasi." *Techno Xplore: Jurnal Ilmu Komputer dan Teknologi Informasi*, 2021, 6.2: 75-81.

[17] A. Aldian and M. Mubarak, "Implementasi Algoritma Rabin-Karp Untuk Pendeteksian Plagiarisme Pada File Dokumen Berupa Text Berbasis Web," *Journal of Information System Research (JOSH)*, vol. 3, no. 3, pp. 150–154, Apr. 2022, doi: 10.47065/josh.v3i3.1404.