Volume 4 Nomor 1 Januari 2022 **JJEEE** e-ISSN : 2715-0887
p-ISSN : 2654-7813

Jambura Journal of Electrical and Electronics Engineering

Typo Checking Menggunakan Algoritma Rabin-Karp

Irma Surya Kumala Idris
Prodi Teknik Informatika
Universitas Ichsan Gorontalo
Gorontalo, Indonesia
mhaladp@gmail.com

Yasin Aril Mustofa
Prodi Teknik Informatika
Universitas Ichsan Gorontalo
Gorontalo Indonesia
arieldcc@gmail.com

Diterima : November 2021
Disetujui : Desember 2021
Dipublikasi : Januari 2022

***Abstrak*—Kesalahan pengetikan merupakan hal yang biasa terjadi ketika membuat tulisan, misalnya ketika membuat karya ilmiah, buku maupun lainnya. Kesalahan penulisan kata memang hal yang biasa terjadi tetapi dapat berakibat buruk sehingga perlu dilakukan pemeriksaan kata terhadap tulisan pada dokumen yang dibuat. *Typo checking* merupakan proses pemeriksaan kata yang dilakukan untuk mendeteksi kesalahan penulisan kata dan memberikan kandidat kata yang benar. Pemeriksaan kesalahan penulisan membutuhkan waktu lama jika dilakukan secara manual, sehingga dibuat aplikasi untuk mendeteksi kesalahan penulisan kata menggunakan Algoritma *Rabin-Karp,* dengan mencocokkan *string* berdasarkan nilai *hash* pada *teks* dan *pattern*. Proses Pengecekan Penulisan Kata dilakukan dengan menghitung sampai indeks akhir dan didapatkan hasil seperti kata dan nilai *hash*. Proses *hashing* menggunakan modulo (sisa bagi) sebesar 107 dengan nilai k-gram k=3 pada setiap kata asal dan kata hasil. Proses *hashing* dilakukan dengan cara mengkonversi *string* menjadi nilai ASCII, sehingga diperoleh nilai *hash* a-z = 79-122. Berdasarkan hasil perhitungan manual yang telah dilakukan, jika terdapat kesalahan pengetikan akan diperoleh nilai *hashing* yang berbeda antara kata asal dan kata yang dihasilkan.**

Kata Kunci—Kesalahan Pengetikan; *Rabin-Karp*; *Hash*; *Pattern*; *String; Typo Checking*.

Abstract*—Typing errors are common when writing, for example, when writing scientific papers, books and others. Word writing errors are common but can have bad consequences, so it is necessary to check the words on the writing in the document that is made. Typo checking is a word checking process that is carried out to detect word writing errors and provide the correct word candidate. Checking writing errors takes a long time if done manually, so an application is made to detect word writing errors using the Rabin-Karp Algorithm, by matching strings based on hash values in text and patterns. The process of Checking Word Writing is done by counting to the final index and getting results such as words and hash values. The hashing process uses a modulo (remaining for) of 107 with a value of k-gram k=3 for each word of origin and word of result. The hashing process is done by converting the string into an ASCII value, so that the hash value a-z = 79-122. Based on the results of manual calculations that have been carried out, if there are typing errors, a different hashing value will be obtained between the original word and the resulting word.

***Keyword*—Typing Error; Rabin-Karp; Hashing; Patterns; String; Typo Checking.**

I. PENDAHULUAN

Saat ini penulisan dokumen seperti skripsi, artikel ilmiah, buku maupun dokumen lainnya banyak dilakukan dengan menggunakan komputer, salah satu kebiasaan dalam penulisan dokumen adalah dengan melakukan pengetikan teks. Sementara itu, pengetikan teks ini bisa saja mengakibatkan terjadinya kesalahan sehingga kata yang dihasilkan menjadi salah dan makna kata bisa saja berubah [1].

Penulisan buku maupun karya ilmiah menuntut kita harus menggunakan kata dan tata tulis yang benar, seringnya terjadi kesalahan kata seperti typo mengakibatkan panjangnya proses terselesainya sebuah naskah buku maupun karya ilmiah, mengingat banyaknya waktu yang dibutuhkan dalam melakukan revisi dan pengecekan kembali penulisan kata yang digunakan. Kesalahan penulisan kata bisa saja ditemukan kembali setelah tahap pengecekan dilakukan, kurangnya pengetahuan terhadap penggunaan ejaan kata dapat mempengaruhi informasi yang akan disampaikan penulis. Kesalahan penggunaan ejaan serta kesalahan penulisan kata dapat berakibat buruk karena makna pengetahuan yang ingin disampaikan dapat berubah sehingga pembaca dapat menerima informasi yang tidak sesuai.

Pendeteksian kesalahan penulisan saat ini telah banyak dilakukan, proses pengecekan kata untuk deteksi kata yang salah pengeejaan dan memberikan saran kata yang benar atau biasa dikenal dengan istilah *Typo Checking* [15]. Penelitian tersebut diantaranya tentang Sistem Koreksi Kesalahan Pengetikan Menggunakan *Levenshtein Distance* Pada *Layout Qwerty*, penelitian ini menggunakan keyboard dengan *layout Qwerty* dalam penulisan, sehingga posisi karakter yang tidak berurutan membuat kesalahan dalam pengetikan kata-kata. Sistem untuk memeriksa penulisan ini dibuat menggunakan *Levenshtein Distance* yang digunakan untuk meminimalisir kesalahan penulisan. Penelitian ini dimulai dengan tahap pengambilan sampel, selanjutkan dilanjutkan ke tahap Filtering, Segmentasi Kata, Segmentasi Huruf hingga ke

tahap seleksi *Qwerty*. Pada penelitian ini menggunakan *Levenshtein Distance* dengan bantuan karakteristik *qwerty* [2].

Penelitian selanjutnya tentang Identifikasi kesalahan penulisan kata (*Typographical Error*) pada Dokumen Berbahasa Indonesia menggunakan metode *N-gram* dan *Levenshtein Distance*. Penelitian ini digunakan untuk memeriksa penulisan Tugas Akhir mahasiswa. Menggunakan *Levenshtein Distance* untuk deteksi jumlah kandidat kata yang terdeteksi tidak sesuai, selanjutnya akan diurutkan menggunakan metode N-gram. Pada penelitian ini, N-gram menggunakan nilai N = 2, dan setiap 2 karakter kata yang terdeteksi dilakukan pemisahan beserta kandidat katanya, dan diperoleh hasil presisi 0.97 serta hasil *recall* terbaik sebesar 1 pada uji coba *substitution* [3]. Penelitian lain mengenai Analisis Algoritma *Rabin-Karp* Pada Kamus Umum Berbasis Android. Pada Penelitian ini Algoritma *Rabin-Karp* digunakan di bagian "cari kata", berfungsi untuk menelusuri hasil terjemahan dari bahasa Indonesia ke Bahasa lain maupun sebaliknya. Selain itu akan ditampilkan jendela pesan yang menampilkan kata yang dicari, ditemukan, dan lama waktu pencarian. Rata-rata *running time* yang dibutuhkan pada pencarian kata yaitu sebesar 14.9 ms untuk 10 kali percobaan [4]. Berdasarkan penelitian diatas Algoritma *Rabin-Karp* digunakan hanya sebatas untuk mencari dan menghasilkan terjemahan kata. Sedangkan pada penelitian ini penulis menerapkan Algoritma *Rabin-Karp* untuk mendeteksi kesalahan penulisan kata, serta menampilkan saran perbaikan dari hasil kesalahan penulisan yang ditemukan.

Pentingnya melakukan pengecekan kembali terhadap penulisan kata yang digunakan dalam penyusunan karya ilmiah dan untuk meminimalisir penggunaan waktu pengecekan, sehingga penelitian ini bertujuan untuk merancang perangkat lunak yang dapat membantu mengidentifikasi kesalahan penulisan kata yang akan dilakukan dengan bantuan Algoritma *Rabin-Karp*, yaitu salah satu Algoritma Pencocokan *String* dengan memanfaatkan fungsi nilai *hash*. Penelitian ini diharapkan bisa mempermudah dalam melakukan pemeriksaan kembali dokumen karya ilmiah yang dibuat.

II. METODE

Penelitian ini berdasar pada metode penelitian deskriptif, merupakan metode yang melakukan pemecahan masalah yang diamati dengan menelaah kondisi pokok atau tujuan pada penelitian baik itu lembaga, masyarakat, orang, serta lainnya berdasarkan fakta yang ada.

A. *Analisis Sistem*

Analisis sistem yang digunakan dengan pendekatan berorientasi objek yang digambarkan dalam bentuk :

a. *Function Modelling*, menggunakan alat bantu UML, berupa : Diagram *Use Case*, dan Diagram *Activity*

b. *Behavioral Modelling*, menggunakan alat bantu UML, dalam bentuk : *Sequence Diagram*.

B. *Spelling Correction*

Spelling Correction merupakan sistem yang bisa digunakan dalam pengoreksian kesalahan pengetikan. Seperti penulisan huruf yang lebih, kurang, atau penempatan huruf yang tidak sesuai. Berikut contoh kesalahan pengetikan yang biasa terjadi :

TABEL 1. PENULISAN KATA YANG SALAH DAN KEMUNGKINANNYA

Kata yang salah	Kemungkinan kata	Penyebab Kesalahan
Kiash	Kisah, Kasih	Kesalahan urutan 'i,a,s'
	Kias	Huruf lebih terdapat huruf 'h'
Temn	Teman	Huruf kurang Kurang huruf 'a'
Keuar	Tema, Temu	Salah Huruf Menghapus huruf 'n' dan menggantinya dengan huruf 'a' atau 'u'
	Keluar	Huruf kurang Kurang huruf 'l'

Pengecekan penulisan kata secara otomatis digunakan untuk memeriksa dan melakukan perbaiakan terhadap kata yang salah berdasarkan kumpulan kata yang dianggap mendekati dengan kata yang salah [5].

C. *String Matching*

String Matching merupakan sebuah proses pencarian pola susunan *string* dengan *string* lainnya pada isi teks, atau lebih dikenal dengan istilah pencocokan string [13]. Pencarian string ini akan mencari semua hasil string dengan nilai *pattern* yang kecil [0….n-1] dan pattern string lebih besar teks [0…m-1] [14].

Pencocokan String memiliki kerangka kerja seperti dibawah [13]:

1. *Text* adalah sebuah (*long*) yang panjangnya adalah *n*
2. *Pattern*, merupakan suatu string yang panjangnya adalah *m.*

Nilai karakter (m < n) akan dilakukan pencarian pada teks, diasumsikan berada di memori, sehingga bila dilakukan pencarian string, sehingga seluruh bagian arsip akan dibaca dan disimpan. Hasil yang dimunculkan berupa pattern pada teks, dan memunculkan tempat awal pattern ditemukan jika pattern muncul lebih dari satu kali.

D. *Text Mining*

Text mining merupakan bidang dalam *data mining* [9], yang melakukan proses penggalian informasi dari kumpulan dokumen menggunakan *tools* analisis pada *data mining* [10]. *Teks Mining* adalah suatu bagian pada *Artificial Intelligence* yang menggunakan konsep *data mining*, dan pada prosesnya dilakukan penggalian data berupa teks, yang sumber datanya

diperolah dari dokumen, dan bertujuan untuk menelusuri kata serta menampilkan informasi dari dokumen [11].

Teks Mining mengadopsi beberapa teknik pada bidang lain, diantaranya : Pengambilan Informasi, Penambangan Data, Statistik dan Matematik, Pembelajaran Mesin, Pemrosesan Bahasan Alami, Linguistik, serta Visualisasi. *Teks Mining* memiliki tahapan proses pada kegiatan riset yaitu proses pemisahan atau ekstraksi dan dokumentasi teks, praproses data, penghimpunan data statistik serta *indexing*, dan analisis isi [12].

E. Rabin-Karp

Algoritma *Rabin Karp* merupakan Algoritma yang digunakan untuk menangani masalah pencocokan *string* dengan menggunakan fungsi hash [6].

Algoritma *Rabin Karp* bekerja dengan fungsi *hash* untuk membandingkan antara *substring* pada teks (n) dengan *string* yang dicari (m). Jika ada kesamaan pada *hash* kedua, maka akan dilakukan lagi proses perbandingan. Tetapi jika kedua *hash* berbeda, maka *substring* akan bergerak ke sebelah kanan sebesar (n-m) kali. Performa dari algoritma dipengaruhi oleh hasil perhitungan nilai *hash* [7].

Rumus matematis Algoritma *Rabin-Karp* [8] :

$$T_{s+1} = (d(t_s - T[s+1]h) + T[s+m+1] \bmod q \quad (1)$$

dimana :

t_s = nilai desimal dengan panjang m dari *substring* T $[s+1 \ldots s+m]$, untuk $s = 0, 1, \ldots, n-m$

t_{s+1} = nilai desimal selanjutnya yang dihitung dari t_s

d = *radix* desimal (bilangan basis 10)

h = d^{m-1}

n = panjang teks

m = panjang pola

q = nilai *modulo*

Pseudocode dan rumus yang digunakan pada algoritma *Rabin-Karp* adalah sebagai berikut :

```
RABIN-KARP-MATCHER (T, P, d, q)
n       = T.length
m       = P.length
h       = dm-1 mod q
p       = 0
t0      = 0
for     i = 1 to m                      // preprocessing
        p = (dp + P[i]) mod q
        t0 = (dt0 + T[i]) mod q
for s = 0 to n – m
        if p == ts                      //matching
        if P[1..m] == T[s + 1..s + m]
                print "Pattern occurs with shift" s
        if s < n – m
                ts + 1 = (d(ts – T[s + 1] h) + T[s + m + 1])
mod q
```

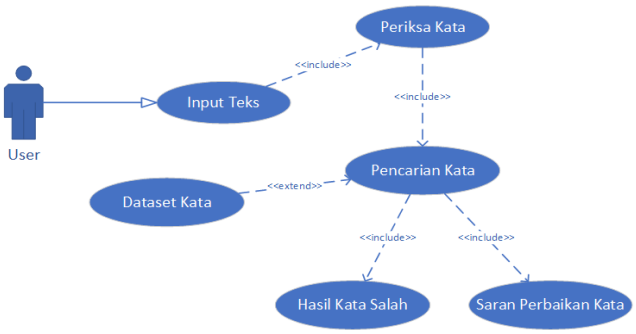
III. HASIL DAN PEMBAHASAN

A. Pemodelan Sistem

Pemodelan sistem digunakan sebagai penggambaran keadaan sistem yang dirancang. Pemodelan sistem dibuat menggunakan Diagram *Use Case*, Diagram *Activity*, dan Diagram *Sequence*

A.1 Use-Case Diagram

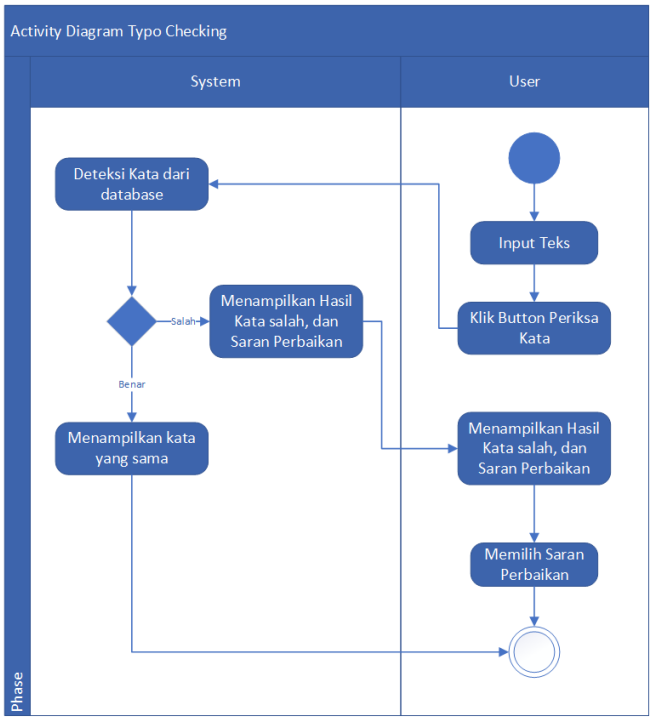
Diagram *Use Case* pada gambar 1 menerangkan proses oleh user dalam melakukan pengecekan kesalahan pengetikan.



Gambar 1. *Use Case Diagram Typo Checking*

A.2 Activity Diagram

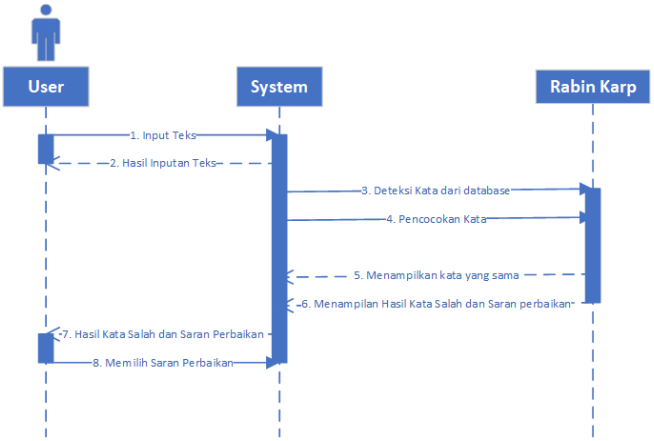
Diagram *Activity* pada gambar 2 menerangkan proses oleh user dalam melakukan pengecekan kesalahan pengetikan.



Gambar 2. *Activity Diagram Typo Checking*

A.3 Sequence Diagram

Sequence Diagram dibawah menunjukkan proses yang berada pada bagian user dalam pengecekan kesalahan pengetikan.



Gambar 3. *Sequence Diagram Typo Checking*

B. *Pembahasan*

TABEL 2. DAFTAR KATA YANG AKAN DIPROSES

Kata Asal	Kata Hasil	Perhitungan
kiash	kisah	91 dan 92
temn	teman	91 dan 23
keuar	keluar	12 dan 36

Setelah dilakukan perhitungan sampai indeks akhir didapatkan hasil seperti kata dan nilai *hash*.

TABEL 3. HASIL ANALISIS

Kata	Hash
kisah	92
teman	23
keluar	36

kiash = $((107*3^4) + (105*3^3) + (97*3^2) + (115*3^1) + (104*3^0))$ MOD_{107}
= $(8667 + 2835 + 873 + 245 + 104)$ MOD_{107}
= 12824 MOD_{107}
= 91

kisah = $((107*3^4) + (105*3^3) + (115*3^2) + (97*3^1) + (104*3^0))$ MOD_{107}
= $(8667 + 2835 + 1035 + 291 + 104)$ MOD_{107}
= 12932 MOD_{107}
= 92

temn = $((116*3^3) + (101*3^2) + (109*3^1) + (110*3^0))$ MOD_{107}
= $(3132 + 909 + 327 + 110)$ MOD_{107}
= 4478 MOD_{107}
= 91

teman = $((116*3^4) + (101*3^3) + (109*3^2) + (97*3^1) + (110*3^0))$ MOD_{107}
= $(9396 + 2727 + 981 + 291 + 110)$ MOD_{107}
= 13505 MOD_{107}
= 23

keuar = $((107*3^4) + (101*3^3) + (117*3^2) + (97*3^1) + (114*3^0))$ MOD_{107}
= $(8667 + 2727 + 1053 + 291 + 114)$ MOD_{107}
= 12852 MOD_{107}
= 12

keuar = $((107*3^5) + (101*3^4) + (108*3^3) + (117*3^2) + (97*3^1) + (114*3^0))$ MOD_{107}
= $(26001 + 8181 + 2916 + 1053 + 291 + 114)$ MOD_{107}
= 38556 MOD_{107}
= 36

Proses hashing menggunakan modulo (sisa bagi) 107 dengan k-gram k=3 pada setiap kata asal dan kata hasil. Proses hashing dilakukan dengan cara mengkonversi string menjadi nilai ASCCI. a-z = 79-122

C. *Implementasi Sistem*

Dataset yang digunakan diperoleh dari kamus besar Bahasa Indonesia yang berjumlah 30879 daftar kata yang akan digunakan dalam proses pengecekan kata atau kalimat yang diinput di sistem.

Pengujian dilakukan dengan menggunakan nilai K-gram = 3, dengan memasukkan data uji dan data pembanding, setelah itu dilakukan proses Analisis dan menghasilkan hasil pengelompokan K-Gram Uji dan nilai K-Gram Pembanding. Hasil analisis juga akan menampilkan Nilai Hash Uji serta Nilai Hash Pembanding. Hasil akhir pengujian akan menampilkan Jumlah Hash Kalimat, serta menampilkan nilai *Similarity*.

Berikut tampilan proses analisis sistem yang dilakukan :

Gambar 5. Form Input Data dan Nilai K-Gram

Gambar 6. Hasil Pengelompokan K-Gram Uji dan Pembanding

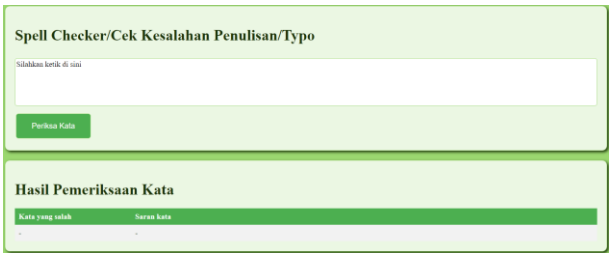
Gambar 7. Hasil Nilai *Hash* Uji dan Nilai *Hash* Pembanding

Gambar 8. Hasil Nilai *Similarity*

Berdasarkan hasil analisis sistem diatas, diperoleh hasil Nilai *similarity* yaitu sebesar 85,71 %.

Untuk tampilan sistem deteksi kesalahan penulisan kata yang dibuat, dapat dilihat pada gambar berikut:

Tampilan halaman pengecekan kesalahan penulisan kata dalam kalimat

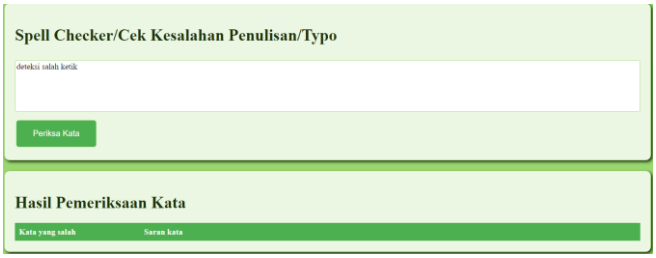


Gambar 9. Form *Typo Checking*

Tampilan halaman hasil pengecekan kesalahan penulisan dan saran perbaikan

Gambar 10. Hasil *Typo Checking* Jika Ada Kesalahan

Tampilan halaman hasil pengecekan jika tidak ditemukan adanya kesalahan penulisan kata



Gambar 11. Hasil *Typo Checking* Jika Tidak Ada Kesalahan

IV. KESIMPULAN

Berdasarkan hasil pengujian dan pembahasan metode yang dilakukan menghasilkan nilai similarity rata-rata di atas 80%, sehingga dapat disimpulkan bahwa perangkat lunak yang direkayasa dapat berfungsi sesuai dengan harapan serta telah cukup memenuhi tujuan awal pembangunan dan Algoritma *rabin-karp* cendrung lebih cepat dan efisien dalam pencarian kata. *Rabin-Karp* mampu melakukan pencarian *String* dan memberikan solusi saran perbaikan dari kesalahan pengetikan yang ditemukan serta dapat menambah kosa kata baru pada aplikasi *typo checking*. Algoritma *Rabin-Karp* juga mampu menyimpan sebuah informasi yang dapat digunakan untuk mendapatkan informasi pada aplikasi.

REFFERENSI

[1] Agustin Sendy, Kenny, Suryaningrum Kristien. "Aplikasi Koreksi Kesalahan Penulisan Kata Dalam Bahasa Inggris dengan Menggunakan Algoritma Rabin-Karp". Jurnal Ilmiah Informatika Komputer, pp. 105-115, Januari 2019.

[2] Aldiasto A.L, Witanti Wina, Yuniarti Rezki. "Sistem Koreksi Kesalahan Pengetikan Menggunakan Levenshtein Distance Pada Layout Qwerty", Seminar Nasional Telekomunikasi dan Informatika (SELISIK) Bandung, pp. 171-176, Mei 2016.

[3] Fahma, A., Cholissodin, I., & Perdana, R. "Identifikasi Kesalahan Penulisan Kata (Typographical Error) pada Dokumen Berbahasa Indonesia Menggunakan Metode N-gram dan Levenshtein Distance". Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. 2, no. 1, p. 53-62, Agustus 2017. ISSN 2548-964X.

[4] Herriyance, Handrizal, dan S. D. Faradilla, "Analisis algoritma Rabin-Karp pada kamus umum berbasis Android", Jurnal Riset Sistem Informasi dan Teknik Informatika, vol. 2, no.1, hal. 64 – 74, Juli 2017. ISSN 2527-5771

[5] M.O. Braddley, M. Fachrurrozi, Novi.Y, "Pengoreksian Ejaan Kata Berbahasa Indonesia Menggunakan Algoritma Levenshtein Distance", Prosiding Annual Research Seminar, Computer Science and ICT, vol.3, no. 1, hal. 167-171, 2017.

[6] Nugroho, E., "Perancangan Sistem Deteksi Plagiarisme Dokumen Teks Dengan Menggunakan Algoritma Rabin-Karp", Skripsi Jurusan Ilmu Komputer, Universitas Muhammadiyah Malang, 2011.

[7] Viny C.M, Bagus M, Desi A., "Implementasi Spelling Correction dengan D-LD dan Rabin Karp Plagiarism Checking Pada Aplikasi Pendaftaran Skripsi", Journal of Computer Science and Information Systems, vol. 4, no. 1, hal. 78-89, April 2020.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to algorithms, USA: MIT Press, 2001.

[9] R. Feldman and J. Sanger, "The text mining handbook: advanced approaches in analyzing unstructured data, Cambridge: Cambridge University Press, 2007.

[10] Salmuasih, Andi S., "Implementasi Algoritma Rabin Karp untuk Pendeteksian Plagiat Dokumen Teks Menggunakan Konsep Similarity", Seminar Nasional Aplikasi Teknologi Informasi (SNATI), hal. F-23 – F-28, Juni 2013

[11] Baskoro, S.Y., "Pencarian Pasal pada Kitab Undang-Undang Hukum Pidana (KUHP) berdasarkan Kasus Menggunakan Metode Cosine Similarity dan Latent Semantic Indexing (LSI), 2006.

[12] C. Triawati, "Metode Pembobotan Statistical Concept Based untuk Klastering dan Kategorisasi Dokumen Berbahasa Indonesia", Institut Teknologi Telkom Bandung, 2009.

[13] Syaroni, Mokhamad, dan Munir, Rinaldi, "Pencocokan String Berdasarkan Kemiripan Ucapan (Phonetic String Matching) dalam Bahasa Inggris, Institut Teknologi Bandung, 2014.

[14] Charras, Christian., et al, "Handbook of Exact String Matching", 1997.

[15] Astawijaya, Ali Nurcahya. "Perbandingan Levenshtein, Smith-Waterman Dan Needleman-Wunsch Dalam Typo Checking. Diss. Universitas Komputer Indonesia, 2019.