ANALISA KOREKSI KATA SOAL UJIAN SEMESTER DENGAN ALGORITMA LEVENSHTEIN DISTANCE

Aida Indriani[1*], Mussalimah[2], Suprianto[3]
[1,2]Program Studi Teknik Informatika, STMIK PPKIA Tarakanita Rahmawati
Jl. Yos Sudarso No. 8, Tarakan 77121
[3]Program Studi Magister Teknik Informatika, Universitas Ahmad Dahlan
Jl. Prof. Dr. Soepomo, S.H. Janturan, Yogyakarta 55164
[*]Email: aida@ppkia.ac.id

Abstrak

Setiap Perguruan Tinggi pasti mempunyai aturan yang diterapkan dalam pengelolaan akademik, salah satunya adalah untuk mengevaluasi hasil belajar setiap mahasiswa untuk setiap semester. Proses UTS maupun UAS diserahkan sepenuhnya oleh bagian akademik. Soal ujian semester yang dikumpulkan dalam bentuk soft copy (file) sering kali terdapat kesalahan pengetikan (typo). Untuk meminimalkan kata typo pada soal ujian semester, penulis melakukan analisa koreksi kata pada soal ujian semester sebelum diperbanyak dan didistribusikan. Analisa koreksi kata yang dilakukan adalah dengan menggunakan teknik text mining yaitu mengubah teks menjadi kata (term) dengan proses pre-processing dan menggunakan algoritma Levenshtein Distance sebagai persamaan dalam memperoleh nlai jarak kata typo dengan kata pada kamus besar bahasa Indonesia (KBBI) dengan membuat sebuah matrix yang akan menghasilkan nilai edit distance. Selanjutnya dilakukan perhitungan nilai similarity antara kata typo dengan kata pembanding. Nilai similarity yang tertinggi merupakan kata yang tepat dijadikan kata koreksi. Langkah terakhir algoritma Levenshtein Distance adalah melakukan 3 (tiga) operasi utama yaitu menambah, mengubah atau menghapus karakter. Hasil analisa dari penelitian dapat disimpulkan bahwa penggunaan algoritma Levenshtein Distance untuk melakukan koreksi kata dapat dilakukan. Tetapi kata koreksi yang diperoleh menghasilkan lebih dari 1 (satu) kata koreksi dikarenakan banyaknya kata dalam KBBI yang mempunyai karakter yang hampir sama.

***Kata kunci**: koreksi kata; levenshtein distance; text mining; similarity*

1. PENDAHULUAN

Ujian semester adalah ujian yang dilakukan oleh Perguruan Tinggi kepada Mahasiswa sebanyak 2 (dua) periode yaitu setiap semester pada tengah dan akhir. Adapun fungsi dari dilakukannya Ujian semester adalah untuk mengevaluasi hasil proses belajar mengajar selama 1 (satu) semester dengan hasil akhir berupa grade nilai dari A sampai dengan E. STMIK PPKIA Tarakanita Rahmawati, dalam mengevaluasi hasil belajar setiap Mahasiswa dengan melakukan pengolahan nilai-nilai yang diserahkan langsung kepada Dosen pembina Mata Kuliah yang terdiri dari nilai tugas, nilai quiz dan nilai ujian semester. Ujian semester dilakukan pada tengah dan akhir semester, dimana untuk tengah semester disebut sebagai UTS dan akhir semester disebut dengan UAS. Pada STMIK PPKIA Tarakanita Rahmawati Tarakan, sebelum melakukan ujian semester bagian Akademik mempersiapkan beberapa poin penting yang terkait dengan pelaksanaan ujian antara lain mempersiapkan kartu ujian untuk Mahasiswa yang memenuhi syarat untuk mengikuti ujian semester, soal ujian yang harus diperbanyak sesuai dengan jumlah Mahasiswa yang mengikuti ujian, mengatur jadwal dosen pengawas yang bertugas mengawasi Mahasiswa pada saat ujian berjalan, dan sebagainya.

Salah satu poin penting adalah pengelolaan soal ujian. Sesuai dengan prosedur yang ada, setiap Dosen wajib mengumpulkan soal ujian dalam bentuk soft copy (file), dan kemudian diperbanyak oleh bagian Akademik sesuai dengan jumlah Mahasiswa yang mengikuti ujian. Dalam pendistibusian soal ke Mahasiswa terdapat beberapa kata dalam soal yang mengalami kesalahan karakter (typo) dan berakibat membuat bingung Mahasiswa

dalam mengartikan makna soal. Untuk meminimalkan terjadinya kesalahan kata pada soal ujian yang diberikan oleh Dosen sebelum dibagikan ke Mahasiswa maka diperlukan analisa koreksi kata yang terdapat pada soal ujian semester. Untuk melakukan analisa koreksi kata diperlukan teknik text mining.

Text mining adalah sering disebut pemrosesan teks merupakan salah satu bidang pengetahuan pada *Artificial Intelligence* yang menerapkan konsep dan teknik *data mining* untuk mencari pola dalam teks dengan proses *ekstraksi* pola yang berupa informasi dan pengetahuan berguna dari sejumlah besar sumber data yang tidak terstruktur. Dalam *text mining* dilakukan penambangan data yang berupa teks dimana sumber data biasanya didapatkan dari dokumen, dan tujuannya adalah mencari kata yang memberikan penjelasan isi dari dokumen sehingga dapat dilakukan analisa keterkaitan dokumen (A.I Fahma dkk., 1998).

Dalam penelitian ini, penulis menggunakan teknik *text mining* untuk melakukan analisa koreksi kata pada soal ujian semester dengan *algoritma Levenshtein Distance* sebagai penghitung jarak antara kata yang terdapat pada soal dengan kata yang terdapat pada kamus besar bahasa Indonesia. Selanjutnya diikuti dengan mencari nilai *similarity* yang tertinggi untuk kemudian dilakukan langkah terakhir yaitu 3 (tiga) operasi utama menambah, mengubah dan menghapus karakter.

2. METODOLOGI

Pada penelitian ini, penulis melakukan metodologi penelitian terkait dengan beberapa teori yang digunakan untuk melakukan analisa koreksi kata pada soal ujian semester.

2.1 *Text Mining*

Text mining merupakan proses pencarian pola atau penggalian informasi dari data teks untuk menghasilkan informasi baru. Tujuan dari *text mining* adalah menemukan informasi yang penting dari teks dengan mengubah teks menjadi data yang dapat digunakan untuk analisis yang lebih lanjut (T.N. Mahgfira dkk., 2017). Pengasosiasian satu bagian teks dengan yang lainnya berdasarkan aturan tertentu merupakan teknik *text mining* dalam menganalisa sebagian atau keseluruhan *unstructured text* (I. Adiwijaya, 2006). Diperlukan beberapa tahap awal dalam teknik text mining yaitu melakukan persiapan perubahan pada teks agar lebih terstruktur. Tahapan awal yang dilakukan pada *text mining* yaitu *pre-processing* diantaranya *case folding, tokenizing, filtering* dan *stemming* (T.N. Mahgfira dkk., 2017). Untuk analisa koreksi kata, penulis hanya menggunakan *case folding* dan *tokenizing*.

Proses pertama dari rangkaian *pre-processing* dokumen adalah *case folding*. Dalam proses ini dilakukan perubahan pada kata menjadi huruf kecil (a sampai z) dalam sebuah dokumen. *Tokenizing* merupakan tahapan dimana dilakukannya pemotongan terhadap *string input* berdasarkan atas *delimiter* yang telah ditentukan. Karakter selain huruf akan dianggap sebagai delimiter dan akan dihilangkan atau dihapus untuk proses mendapat kata-kata penyusun teks. Dari proses ini akan dihasilkan kata-kata penyusun string atau teks atau yang sering disebut *token* atau *term* (A. Yudhana dkk., 2017).

2.2 *Levenshtein Distance*

Tahun 1965, Vladimir Levenshtein mengembangkan *algoritma Levenshtein distance*. *Levenshtein Distance* merupakan salah satu metode untuk menghitung jarak perbedaan antar kata dari beberapa metode perhitungan jarak lainnya. *Levenshtein Distance* dilakukan dengan bantuan matrix yang digunakan untuk menghitung nilai hasil modifikasi dari satu kata terhadap kata lainnya (A.L. Adiasto dkk., 2016). Perhtungan *edit distance* diperoleh dari menghitung jumlah perbedaan *string* antara dua *string* dengan bantuan sebuah *matriks*. Operasi perubahan untuk membuat *string* A menjadi *string* B ditentukan dari jumlah minimum. Pada *algoritma Levenshtein Distance* terdapat 3 (tiga) operasi utama yang dilakukan yaitu: pengubahan karakter, penambahan karakter dan penghapusan karakter.

a) Operasi pengubahan karakter merupakan operasi menukar sebuah karakter dengan karakter lain, contohnya penulis menuliskan *string* ‘zaya’ menjadi ‘saya’. Dalam kasus ini karakter ‘z’ diganti dengan huruf ‘s’.
b) Operasi penambahan karakter berarti menambahkan karakter ke dalam suatu *string*. Contohnya *string* ‘ketem’ menjadi *string* ‘ketemu’, dilakukan penambahan karakter ‘u’ di akhir *string*. Karakter yang ditambahkan tidak hanya dilakukan pada akhir kata, tapi dapat ditambahkan pada awal maupun pada tengah kata dengan cara disisipkan.
c) Operasi penghapusan karakter dilakukan untuk menghilangkan karakter dari suatu kata. Contohnya kata ‘makang’ karakter terakhir dihilangkan sehingga menjadi kata ‘makan’. Pada operasi ini dilakukan penghapusan karakter ‘g’ (N.M.M. Adriyani dkk., 2012).

Matrix (m,n) digunakan untuk membandingkan kata dimana M mewakili kata yang dibandingkan, dan N mewakili kata pembanding yang masing-masing mewakili setiap huruf agar mempermudah dalam menentukan operasi apa yang perlu dilakukan untuk kata tersebut. Nilai yang akan didapat adalah seberapa banyak langkah yang diselesaikan untuk mendapatkan kemiripan kata. Total angka untuk setiap operasinya mengacu kepada jarak, dimana semakin kecil jarak semakin besar kemungkinan kata sesuai dengan target kata. Persamaan yang digunakan untuk mencari jarak, seperti pada (1).

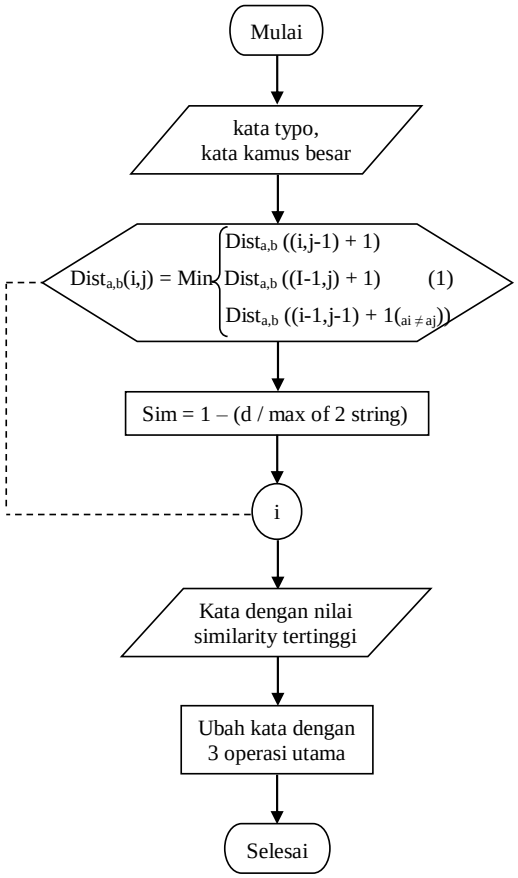
$$\begin{aligned} & Dist_{a,b}((i,j-1)+1) \\ Dist_{a,b}(i,j) = \text{Min} \quad & Dist_{a,b}((i-1,j)+1) \\ & Dist_{a,b}((i-1,j-1)+1_{(a_i \neq a_j)}) \end{aligned} \tag{1}$$

keterangan:
a : kata pertama
b : kata kedua
i : *iterasi* kata pertama
j : *iterasi* kata kedua
Dist : jarak (A.L. Adiasto dkk., 2016).

Persamaan (2) digunakan untuk mendapatkan nilai kemiripan antara kedua kata yang dibandingkan.

$$Sim = 1 - (d / \text{max of 2 string}) \tag{2}$$

keterangan:
Sim = *similarity*/kemiripan
d = nilai *edit distance*
max of 2 *string* = jumlah maksimal karakter dari kedua kata (K.N.M. Ngafidin dkk., 2015).
Untuk alur algoritma Levenshtein Distance ditunjukkan pada Gambar 1.



Gambar 1. Alur Algoritma Levenshtein Distance

Tahapan alur alogirtma Levenshtein Distance seperti yang ditunjukkan pada gambar 2 yaitu dengan cara menghitung jarak antara kata typo dan kata yang terdapat dalam kamus besar Bahasa Indonesia dengan membentuk sebuah matrix dan mencari nilai minimal dengan menggunakan persamaan (1) dan dilanjutkan dengan mencari nilai similarity yaitu nilai kemiripan kata typo dengan kata yang terdapat pada KBBI dengan perhitungan yang dijelaskan pada persamaan (2). Kata pada KBBI yang memiliki nilai similarity tertinggi diambil untuk kemudian dilakukan perbaikan kata dengan 3 operasi utama yaitu mengubah, menambah dan menghapus karakter.

2.3 Kamus Besar Bahasa Indonesia (KBBI)

Pada KBBI, ejaan yang digunakan adalah ejaan Bahasa Indonesia yang berdasarkan pada pedoman umum ejaan Bahasa Indonesia yang telah disempurnakan dan pedoman umum untuk pembentukan istilah (Kamus Besar Bahasa Indonesia, 2008). Beberapa contoh kata yang terdapat dalam KBBI ditunjukkan pada Tabel 1.

Tabel 1. Contoh kata dalam kamus bahasa indonesia

No	Kata
1	Aksi
2	Aktif
3	Buih
4	Salai
5	Salah
6	Salam
7	Buas
8	Buah
9	Bunga

3. HASIL DAN PEMBAHASAN

3.1. Tahapan *Pre-processing*

Tahapan *pre-processing* yang dilakukan dalam penelitian ini yaitu tahapan *case folding* dan *tokenizing*. Berikut adalah contoh teks yang terdapat pada soal ujian semester "Pesan yang akan tampil di layar adalah Salh, ini Buzh". Dari potongan teks yang terdapat pada soal ujian semester dilakukan *pre-processing* dengan mendapatkan hasil seperti yang ditunjukkan pada Tabel 2.

Tabel 2. Hasil tahapan proses *pre-processing*

No	Kata dalam potongan teks soal ujian semester
1	Pesan
2	Yang
3	Akan
4	Tampil
5	Di
6	Layar
7	Adalah
8	Salh
9	Ini
10	Buzh

3.2. *Algorithma Levenshtein Distance*

Setelah teks melewati proses *pre-processing* sehingga menghasilkan kata seperti yang ditunjukkan pada Tabel 2. Langkah selanjutnya adalah menghitung nilai *edit distance* dengan menggunakan matriks terhadap kata yang mengalami typo dengan kata pada kamus besar Bahasa Indonesia seperti yang ditunjukkan pada Tabel 1. Kata yang digunakan untuk perhitungan edit distance dengan menggunakan algoritma *Levenshtein Distance* dengan perbandingan kata dalam kamus besar Bahasa Indonesia ditunjukkan pada Tabel 3.

Tabel 3. Kata yang digunakan untuk Algoritma *levenshtein distance*

No	Kata typo pada soal ujian semester	Kata dalam kamus besar Bahasa Indonesia	
	A		B
1	Salh	1	salai
		2	salah
		3	salam
2	Buzh	1	buas
		2	buah
		3	bunga
		4	buih

Diketahui a1 adalah kata 'salh' dan b1 adalah kata 'salai'. Kemudian dibuatkan *matriks* untuk nilai jarak antara kata a1 dan b1 yang ditunjukkan pada Tabel 4.

Tabel 4. Matrix edit distance antara kata a1 dan b1

		s	a	l	a	I
	0	1	2	3	4	5
s	1	0	1	2	3	4
a	2	1	0	1	1	2
l	3	2	1	0	1	2
h	4	3	2	1	1	2

Nilai yang diperoleh pada *matriks edit distance* dihitung dengan menggunakan persamaan (1), proses perhitungan dengan menggunakan bantuan penempatan karakter pada *array* yang ditunjukkan pada Tabel 5.

Tabel 5. Penempatan karakter pada *array*

		s	a	l	a	i
	0 (index 0)	1 (index 1)	2	3	4	5
S	1 (index 2)	? (index 3)				
A	2					
L	3					
H	4					

Index menunjukkan lokasi cell yang dibutuhkan untuk perhitungan jarak. *Index* mewakili *index* pada *two-dimensional array* yang terdiri dari: *index* 0 (0,0), *index* 1 (0,1), *index* 2 (1,0) dan *index* 3 (1,1). *Index* 3 adalah *cell distance* untuk menghitung jarak karakter pertama. Keempat *cell* ini akan bergeser untuk memberi nilai jarak pada masing-masing karakter. Pengisian nilai *index* 3 berdasarkan persamaan (1), dengan mengambil nilai minimum dari *index* 0, *index* 1 dan *index* 2. Nilai minimum setelah *index* 1 dan *index* 2 dijumlahkan dengan nilai 1, sedangkan *index* 0 dijumlahkan dengan nilai 0, bila kedua karakter *input* dan kamus sama. Bila tidak maka *index* 0 akan dijumlahkan dengan nilai 1 (A. Yudhana dkk., 2017). Untuk mendapatkan nilai *index* 3 pada Tabel 5 dengan perhitungan sebagai berikut:

Dista,b ((i,j-1) + 1) = Dista,b ((1, 1-1) + 1) = Dista,b ((1,0) + 1) = (1 + 1) = 2
Dista,b ((i-1,j) + 1) = Dista,b ((1-1, 1) + 1) = Dista,b ((0,1) + 1) = (1 + 1) = 2
Dista,b ((i-1,j-1) + 1) = Dista,b ((1-1, 1-1) + 1) = Dista,b ((0,0) + 0) = (0 + 0) = 0

Nilai minimal dari perhitungan jarak di atas adalah 0, maka nilai *index* 3 adalah 0. Lakukan perhitungan yang sama untuk mendapatkan nila *index* berikutnya sehingga hasil yang diperoleh seperti ditunjukkan pada Tabel 4. Dengan hasil jarak adalah 2. Nilai jarak yang telah dilakukan perhitungan untuk kata A1 dan A2 dengan seluruh contoh kata yang ada di kolom B ditunjukkan pada Tabel 6.

Tabel 6. Nilai jarak perhitungan *algoritma levenshtein distance*

No	Kata typo pada soal ujian semester	Kata dalam kamus besar Bahasa Indonesia		Nilai Jarak
	A		B	C
1	Salh	1	salai	2
		2	salah	1
		3	salam	2
2	Buzh	1	buas	2
		2	buah	1
		3	bunga	3
		4	buih	1

3.3. Nilai Similarity

Setelah mendapatkan nilai edit distance untuk setiap kata typo dan kata dalam kamus besar Bahasa Indonesia seperti yang ditunjukkan pada Tabel 6, kemudian dilakukan perhitungan nilai similarity untuk setiap perbandingan kata. Perhitungan nilai *similarity* dengan menggunakan persamaan (2). Perhitungan nilai *similarity* untuk kata ‘salh’ dengan ‘salai’ adalah sebagai berikut:

Sim = 1 – (d / max of 2 string) = 1 – (2/5) = 1 - 0.4 = 0.6

Nilai *similarity* yang telah dilakukan perhitungan untuk setiap perbandingan kata ditunjukkan pada Tabel 7.

Tabel 7. Nilai *similarity* antar kata

No	Kata typo pada soal ujian semester		Kata dalam kamus besar Bahasa Indonesia	Nilai Jarak	Nilai Similarity
	A		B	C	D
1	Salh	1	salai	2	0.6
		2	salah	1	0.8
		3	salam	2	0.6
2	Buzh	1	buas	2	0.5
		2	buah	1	0.75
		3	bunga	3	0.4
		4	buih	1	0.75

Dari Tabel 7, dapat disimpulkan kata yang mempunyai nilai *similarity* tertinggi adalah kata koreksi. Untuk kata ‘salh’ mempunyai nilai *similarity* tertinggi dengan kata ‘salah’ dengan nilai 0.8 dan kata ‘buzh’ mempunyai nilai *similarity* tertinggi dengan kata ‘buah’ dan kata ‘buih’ dengan nilai 0.75. Kemudian untuk selanjutnya dilakukan 3 operasi utama yaitu mengubah, menambah dan menghapus karakter dari kata *typo* ke kata koreksi. Berikut adalah perubahan karakter dengan menggunakan 3 operasi utama yaitu:

a) Kata ‘salh’ menjadi kata ‘salah’ dengan nilai jarak adalah 1 maka hanya mengalami 1 dari 3 operasi utama yaitu menambah karakter ‘a’ di tengah kata (salh → salah).
b) Kata ‘buzh’ menjadi kata ‘buah’ dengan nilai jarak adalah 1 maka hanya mengalami 1 dari 3 operasi utama yaitu mengubah karakter ‘z’ menjadi karakter ‘a’ (buzh → buah).
c) Kata ‘buzh’ menjadi kata ‘buih’ dengan nilai jarak adalah 1 maka hanya mengalami 1 dari 3 operasi utama yaiu mengubah karakter ‘z’ menjadi karakter ‘i’ (buzh → buih).

4. KESIMPULAN

Berdasarkan hasil penelitian yang dibuat, dapat ditarik kesimpulan yaitu :

(1) Pemanfaatan teknik *text mining* dalam analisa koreksi kata soal ujian semester dapat dilakukan dengan tahapan awal yaitu proses *pre-processing* dimana hanya menggunakan 2 tahapan awal yaitu *case folding* dan *tokenizing*.
(2) Nilai jarak yang diperoleh dari kata typo terhadap kata pembanding dengan menggunakan *algoritma Levenshtein Distance* dapat membantu mengatasi permasalahan dan dengan perhitungan *similarity* untuk masing-masing kata *typo* dan kata pembanding dapat menampilkan alternatif kata koreksi dengan mengambil nilai *similarity* yang tertinggi.
(3) Kata pembanding yang digunakan adalah kumpulan kata yang terdapat dalam kamus besar Bahasa Indonesia.
(4) Dari uji coba 2 kata *typo*, kata ‘salh’ mempunyai 1 (satu) pilihan kata koreksi dengan nilai *similarity* 0.8, sedangkan kata ‘buzh’ mempunyai 2 (dua) pilihan kata koreksi dengan nilai *similarity* 0.75.

Adapun saran berdasarkan penelitian ini yaitu :

(1) Untuk mengatasi tampilnya koreksi kata lebih dari 1 (satu) alternatif dapat menggunakan fitur *N-gram* yaitu *Uni-gram, Bi-gram, Tri-gram* dan *Quad-gram.*
(2) Dapat menambahkan metode Empiris untuk mengetahui adanya kata yang ditulis tanpa spasi, agar hasil yang diperoleh lebih maksimal.Kata pembanding yang digunakan adalah kumpulan kata yang terdapat dalam kamus besar Bahasa Indonesia.
(3) Analisa koreksi kata dapat dikembangkan dengan menggunakan metode-metode *text* mining lainnya seperti metode *Damerau-Levenshtein Distance* agar bisa mendapatkan hasil yang maksimal.

UCAPAN TERIMA KASIH

Sehubungan dengan telah dirampungkannya penelitian mengenai analisa koreksi kata soal ujian semester, ijinkan penulis menyampaikan ucapan terima kasih yang sebesar-besarnya kepada orang tua penulis, suami, seluruh dosen serta rekan-rekan civitas akademika STMIK PPKIA Tarakanita Rahmawati dalam memberikan saran kepada penulis dan serta semangat sehingga penulis mampu menyelesaikan penelitian dengan tepat waktu. Terlepas dari ucapan terima kasih, penulis masih perlu masukan serta saran yang membangun demi penyempurnaan penelitian ini agar menjadi lebih baik lagi.

DAFTAR PUSTAKA

A.I. Fahma, I. Cholissodin, dan R.S. Perdana, "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, Fakultas Ilmu Komputer Universitas Brawijaya: Malang, hlm.53-62, Januari 2018.

A.L. Adiasto, W. Witanti, dan R. Yuniarti, "Sistem koreksi kesalahan pengetikan menggunakan levenshtein distance pada layout qwerty," Seminar Nasional Telekomunikasi dan Informatika, Universitas Pasundan: Bandung, hlm.171-176, Mei 2016.

A. Yudhana, A.D. Djayali, dan Sunardi, "Sistem deteksi plagiarisme dokumen karya ilmiah dengan algoritma pencocokan pola," Jurnal Rekayasa Teknologi Informasi, Vol.1 No.2, Fakultas Ilmu Komputer dan Teknologi Informasi Jurusan Teknologi Informasi dan Komunikasi Universitas Mulawarman: Samarinda, hlm.178-187, Desember 2017.

I. Adiwijaya, "Text mining dan knowledge discovery," Kolokium bersama komunitas datamining Indonesia & soft-computing Indonesia, September 2006.

"Kamus Besar Bahasa Indonesia," Pusat Bahasa Departemen Pendidikan Nasional, Jakarta 2008.

K.N.M. Ngafidin dan H. Wibawanto, "Implementasi fitur autocomplete dan algoritma levenshtein distance untuk meningkatkan efektivitas pencarian kata di kamus besar bahasa Indonesia (KBBI)," Jurnal Teknik Elektro, Vol.7 No.1, Fakultas Teknik Jurusan Teknik Elektro Universitas Negeri Semarang: Semarang, hlm.1-6, Juni 2015.

N.M.M. Adriyani, I.W. Santiyasa, dan A. Muliantara, "Implementasi algoritma levenshtein distance dan metode empiris untuk menampilkan saran perbaikan kesalahan pengetikan dokumen berbahasa Indonesia," Jurnal Elektronik Ilmu Komputer Universitas Udayana, Vol.1 No.1, Universitas Udayana: Bali, Agustus 2012.

T.N. Maghfira, I. Cholissodin, dan A.W. Widodo, "Deteksi kesalahan ejaan dan penentuan rekomendasi koreksi kata yang tepat pada dokumen jurnal JTIIK menggunakan dictionary lookup dan damerau-levenshtein distance," Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, Vol.1 No.6, Fakultas Ilmu Komputer Universitas Brawijaya: Malang, hlm.498-506, Juni 2017.