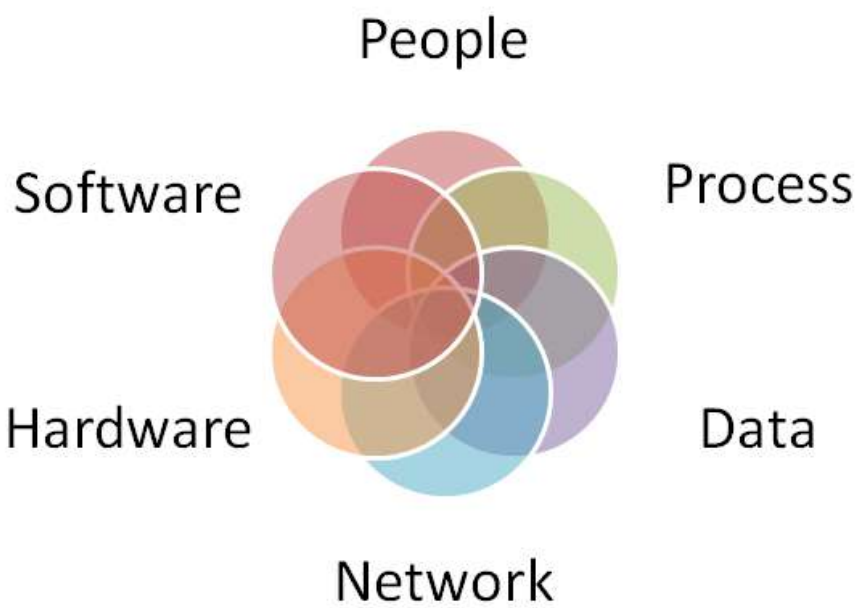
OAJIS
Open Access
Journal of
Information
Systems
is.its.ac.id/pubs/oajis/

ISSN 1979-3979

SISFO

Inspirasi Profesional Sistem Informasi



Vol. 08 No. 01 September 2018

OAJIS
Open Access
Journal of
Information
Systems
is.its.ac.id/pubs/oajis/

SISFO
Inspirasi Profesional Sistem Informasi

Jurnal Sisfo Vol. 08 No. 01 (2018) i-ii

Sekretariat

Departemen Sistem Informasi – Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember (ITS) – Surabaya
Telp. 031-5999944 Fax. 031-5964965
Email: editor@jurnalsisfo.org
Website: http://jurnalsisfo.org

Jurnal SISFO juga dipublikasikan di *Open Access Journal of Information Systems* (OAJIS)

Website: http://is.its.ac.id/pubs/oajis/index.php

OAJIS
Open Access
Journal of
Information
Systems
is.its.ac.id/pubs/oajis/

SISFO
Inspirasi Profesional Sistem Informasi

Jurnal Sisfo Vol. 08 No. 01 (2018) i-ii

Mitra Bestari

Anisah Herdiyanti, S.Kom., M.Sc., ITILF. (Institut Teknologi Sepuluh Nopember)

Hatma Suryotrisongko, S.Kom., M.Eng. (Institut Teknologi Sepuluh Nopember)

Mahendrawathi ER, S.T., M.T., Ph.D. (Institut Teknologi Sepuluh Nopember)

Nur Aini Rakhmawati, Ph.D. (Institut Teknologi Sepuluh Nopember)

Riska Asriana Sutrisnowati, Ph.D. (Pusan National University, Korea)

Satria Fadil Persada, S.Kom., M.BA., Ph.D. (Institut Teknologi Sepuluh Nopember)

OAJIS
Open Access
Journal of
Information
Systems
is.its.ac.id/pubs/oajis/

SISFO
Inspirasi Profesional Sistem Informasi
Jurnal Sisfo Vol. 08 No. 01 (2018) iii

Daftar Isi

Halaman ini sengaja dikosongkan

OAJIS
Open Access Journal of Information Systems
is.its.ac.id/pubs/oajis/

SISFO
Inspirasi Profesional Sistem Informasi
Jurnal SISFO Vol. 08 No. 01 (2018) 19–32

Perbandingan Algoritma Kemiripan Teks untuk Perbaikan dan Saran Penulisan Frasa dalam Bahasa Alami

Nambi Sembilu*, Febriliyan Samopa, Mahendrawathi Er

Departemen Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember

Abstract

Natural Language Processing (NLP) is a branch of science Aritificial Intelligence (AI) that focuses on natural language processing. Natural language is the language generally used by humans to communicate with each other. One of the problems in the use of natural language that is write error the word nor the structure of the word in a sentence. It also results in natural language processing into a computer is not able to get maximum results. So, to solve that problem in this research applies the Algorithm Oliver, LCS Algorithm (Longest Common Subsequences) of the Levenshtein Distance Algorithm and for improvement of writing as well as advising the writing. The results of the application of the Algorithm is shown in the third a simple application with a natural language interface as stuffed. The third test of the Algorithm shows good performance with the average ROC curve above 0.80. Then by comparing the performance of the Algorithm, the third is obtained that the LCS Algorithm has better performance in overcoming problems repair write error and the giving of advice with the indicated value of the ROC curve 0948.

Keywords: Text Prediction, Text Similarity Algorithms, Natural Language Processing

Abstrak

Natural Language Processing (NLP) merupakan salah satu cabang ilmu *Artificial Intelligence* (AI) yang fokus pada pengolahan bahasa alami. Bahasa alami adalah bahasa yang umum digunakan oleh manusia dalam berkomunikasi. Salah satu masalah dalam penggunaan bahasa alami yaitu kesalahan penulisan kata maupun struktur kata dalam sebuah kalimat. Hal tersebut menyebabkan pemrosesan bahasa alami dengan komputer tidak mendapatkan hasil yang maksimal. Penelitian ini menerapkan Algoritma Oliver, Algoritma LCS (*Longest Common Subsequences*) dan *Algoritma Levenshtein Distance* untuk memperbaiki dan memberikan saran penulisan. Sebuah aplikasi sederhana dengan antarmuka bahasa alami sebagai masukannya dirancang untuk mendemonstrasikan hasil penerapan ketiga algoritma tersebut. Hasil pengujian menunjukkan kinerja yang baik dengan nilai rata-rata kurva ROC diatas 0.80. Hasil perbandingan kinerja ketiga algoritma tersebut menunjukkan bahwa Algoritma LCS memiliki kinerja yang lebih baik dalam mengatasi permasalahan perbaikan kesalahan penulisan dan pemberian saran dengan nilai kurva ROC 0.948.

Kata kunci: Prediksi Text, Algoritma Kemiripan Teks, Pemrosesan Bahasa Alami

Histori Artikel: Disubmit 01-08-2018; Direvisi 28-08-2018; Diterima 06-09-2018; Tersedia online 26-09-2018

*Corresponding Author
Email address: nambisembilu@gmail.com (Nambi Sembilu)
https://doi.org/10.24089/j.sisfo.2018.09.004

1. Pendahuluan

Natural Language Processing (NLP) merupakan salah satu cabang ilmu *Artificial Intelligence* (AI) yang berfokus pada pengolahan bahasa alami. Bahasa alami sendiri adalah bahasa yang secara umum digunakan oleh manusia dalam berkomunikasi satu sama lain. Bahasa yang diterima oleh komputer harus diproses dan dipahami terlebih dahulu agar maksud dari pengguna bisa dipahami dengan baik oleh komputer. Penggunaan teknik dalam NLP sudah banyak berhasil untuk mengatasi permasalahan pada berbagai bidang diantaranya: Psikologi [1], keperawatan [2], keselamatan kerja [3], biomedis [4][5], dan masih banyak lagi. Pengembangan teknik NLP ditujukan agar komputer dapat memahami bahasa alami manusia. Bahasa alami yang digunakan manusia dari berbagai negara akan mengalami perbedaan dalam bentuk penulisan dan pengucapan. NLP juga dapat digunakan untuk melakukan pengambilan kembali informasi (*information retreival*) [6]. Pada dasarnya NLP dimanfaatkan untuk mengotomasikan pekerjaan yang bersifat manual dan berasal dari bahasa natural manusia lalu diterjemahkan untuk dalam bahasa komputer agar dapat diproses lagi untuk keperluan selanjutnya.

Salah satu masalah dalam penggunaan bahasa alami yaitu kesalahan penulisan kata maupun struktur kata dalam sebuah kalimat. Hal tersebut menyebabkan pemrosesan bahasa alami kedalam komputer tidak berjalan dengan baik. Salah satu contoh penelitian yang telah dilakukan adalah penerapan bahasa alami untuk kebutuhan ektraksi data yang dapat mengakomodir pertanyaan dan konversi satuan [7] . Saat penulisan kalimat "*When were the books' publish years*" terjadi error saat proses translasi ke dalam bahasa komputer yang terjadi karena kesalahan struktur kalimat yang tidak sesuai dengan kaidah yang sudah ditentukan sebelumnya. Sementara itu, Jurafsky dan Martin [8] memberikan perkiraan frekuensi kesalahan ejaan dalam teks diketik manusia bervariasi dari 1-2% untuk penulisan ulang teks yang sudah ditulis dan 10-15% untuk pencarian di *web*. Beberapa penelitian lain yang berfokus pada kesalahan ejaan kata melaporkan bahwa terdapat satu kesalahan ejaan di setiap lima kalimat pada korpus medis [9]. Mereka juga menemukan bahwa tingkat kesalahan mencapai 10% pada tiap catatan medis. Selain itu pada penelitian dalam Bahasa Indonesia yang menggunakan antarmuka bahasa alami merujuk bahwa kesalahan penulisan maupun kesalahan struktur kata dalam sebuah kalimat tidak akan diproses ke dalam bahasa komputer.

Beberapa penelitian telah dilakukan untuk proses perbaikan penulisan kata maupun ejaan secara otomatis. Dalam dunia nyata hal ini telah diterapkan pada mesin pencari yaitu Google, Yahoo dan Bing yang memiliki Algoritma tersendiri dalam meningkatkan cara penulisan kata kunci sehingga dapat memberikan hasil yang sesuai dengan harapan. Kata kunci dalam mesin pencari bisa memiliki lebih dari satu satu kata atau dalam Bahasa Indonesia disebut frasa. Makna dari sebuah kata atau frasa dapat memiliki arti yang berbeda, termasuk dalam penulisan frasa yang mengandung kata kerja seperti "verifikasi dokumen", "pembelian barang", "pembayaran tagihan" dan lain sebagainya. Penggabungan lebih dari satu kata tersebut akan memiliki makna yang berbeda dengan makna dari tiap kata. Penelitian dalam perbaikan penulisan ejaan kata telah banyak dilakukan, namun untuk perbaikan penulisan frasa yang mengandung kata kerja masih jarang dilakukan. Oleh karena itu dalam penelitian ini akan membahas tentang perbaikan teks yang berbasis frasa yang mengandung kata kerja.

Untuk melakukan perbaikan frasa yang mengandung kata kerja diperlukan kumpulan frasa yang mengandung kata kerja yang digunakan dalam dunia nyata. Korpus tersebut dapat diambil dari tiap aktivitas dalam sebuah proses bisnis dalam sebuah sistem. Contohnya pada proses bisnis "Pendaftaran Seminar", aktivitasnya terdiri dari "Registrasi diri", "Aktivasi akun", "Upload berkas" dan "Verifikasi data". Algoritma kemiripan teks digunakan untuk membandingkan data tersebut dengan teks yang ditulis. Beberapa algoritma telah digunakan untuk menyelesaikan berbagai macam permasalahan diantaranya deteksi plagiasi, pengelompokan dokumen, klasifikasi teks berita, mesin penjawab otomatis dan aplikasi penerjemahan bahasa. Penelitian ini akan berfokus pada penggunaan tiga algoritma kemiripan yaitu Algoritma Oliver [10], Algoritma LCS (*Longest Common Subsequences*) [11] dan Algoritma *Levenshtein Distance* [12] untuk perbaikan ejaan frasa yang mengandung kata kerja dalam antarmuka bahasa alami. Ketiga Algoritma tersebut terbukti memberikan hasil yang baik dalam menyelesaikan berbagai macam permasalahan.

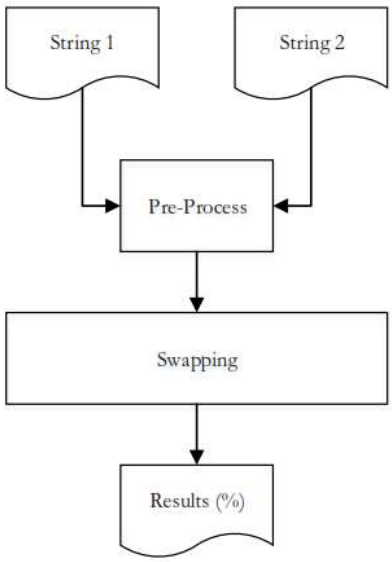
Penelitian ini bertujuan untuk membandingkan algoritma kemiripan teks tersebut untuk mengetahui algoritma terbaik dalam menyelesaikan permasalahan perbaikan penulisan frasa yang mengandung kata kerja.

2. Tinjauan Pustaka/Penelitian Sebelumnya

Pada bagian ini akan dibahas tentang Algoritma kemiripan teks dan *Receiver Operating Characteristic* (ROC) yang menjadi landasan pada penelitian ini.

2.1 Algoritma Oliver (PHP similar_text)

Merupakan sebuah fungsi dalam bahasa pemrograman PHP yang diperkenalkan oleh [10] dalam buku "*Programming classics: implementing the world's best algorithms"*. Di dalam fungsi tersebut membandingkan 2 *string* dengan hasil nilai kemiripan dari ketiga *string* tersebut. Seperti pada Gambar 1 alur kerja sistem kerja fungsi *similar text* ini diawali dengan membandingkan string 1, string 2, dan seterusnya string n, yang selanjutnya dilanjutkan dengan *pre-process* yang ditandai dengan menghitung kemiripan string dalam bentuk persen. Dengan model perhitungan yaitu; membagi hasil *similar text* dengan rata-rata panjang string sehingga akan menghasilkan hasil persentase yang tertinggi 100. Hasil persentase hasil similar text juga dapat diatur sesuai dengan berapa persentase yang diambil dari string tersebut. Sebagai contoh: string 1 berisi "similarity PHP" dan string 2 berisi "PHP similarity" maka jika ingin mengambil hasil persentase similar 0 maka hasil menampilkan >70, jika hasil persentase similar 100% maka hasil akan menampilkan deteksi string 0%. Alur proses *similar text* PHP sebagaimana ditampilkan pada Gambar 1.



Gambar 1. Proses *similar text* PHP

Fungsi ini sering digunakan untuk kebutuhan menilai kemiripan antara dua masukan teks. Setelah dibandingkan fungsi tersebut akan mengembalikan nilai prosentasi kemiripan ketiga teks yang dimasukkan terbut. Fungsi tersebut digunakan pada penelitian ini untuk mengetahui nilai kemiripan dari data kamus frasa yang sudah ada dengan inputan frasa yang salah ataupun benar. Jika ada kesalahan penulisan dan memiliki nilai kemiripan yang tinggi maka akan memberikan sebuah saran untuk memperbaiki penulisan frasa tersebut.

2.2 Longest Common Subsequences (LCS)

Longest Common Subsequences adalah masalah mencari sub-rangkaian bersama (*common subsequence*) terpanjang dari beberapa (biasanya hanya 2) buah rangkaian. Subrangkaian (*subsequence*) dari sebuah string S adalah sekumpulan karakter yang ada pada S yang urutan kemunculannya sama [11]. Atau dalam definisi formalnya Rangkaian Z adalah sub-rangkaian dari $X \langle x_1, x_2, x_3 \dots x_m \rangle$ jika terdapat urutan menaik $\langle i \rangle$ yang merupakan indeks X untuk semua $J = 1,2, \dots k$, yang memenuhi $x_i = z_j$.

Misal pada S = GAATACA, beberapa sub-rangkaian yang mungkin adalah : GAT, TCA, dan GC. Sub-rangkaian bersama (common subsequence) dari 2 rangkaian adalah sub-rangkaian yang terdapat pada ketiga rangkaian tersebut. Misal S1 = GATTTAC dan S2 = AAGATC, maka GAT, GATC, maupun GAC adalah sub-rangkaian bersama dari S1 dan S2.

Sub-rangkaian bersama terpanjang (*longest common subsequences*) adalah Sub-rangkaian bersama dari 2 rangkaian yang paling panjang. Algoritma untuk masalah LCS ini berdasarkan teorema di bawah ini : Misal $X = \langle x_1, x_2, x_3 \dots x_m \rangle$ dan $Y = \langle y_1, y_2, y_3 \dots y_m \rangle$ adalah rangkaian $Z = \langle z_1, z_2, z_3 \dots z_m \rangle$ adalah suatu LCS dari X dan Y.

1) Jika $x_m = y_n$ maka $z_k = x_m = y_n$ dan z_{k-1} adalah suatu LCS dari x_{m-1} dan y_{n-1}.
2) Jika $x_m \neq y_n$ maka $z_k \neq x_m$ mengimplikasi bahwa Z adalah suatu LCS dari x_{m-1} dan Y
3) Jika $x_m \neq y_n$ maka $z_k \neq y_n$ mengimplikasi bahwa Z adalah suatu LCS dari X dan y_{n-1}

Pada penelitian ini sub-rangkaian yang dimaksut dalam algoritma LCS adalah frasa yang teridiri dari susunan huruf. Dengan membandingkan antarai S1 (frasa pertama) sebagai masukkan dengan S2 (frasa ketiga) sebagai kamus frasa yang sudah disimpan akan didapatkan nilai jumlah rangkaian yang dimiliki S1 maupun S2. Semakin banyaj nilai jumlah rangkaian yang ada menunjukkan bahwa semakin tinggi tingkat kemiripan antara ketiga frasa tersebut.

2.3 Levenshtein Distance

Levenshtein distance adalah suatu matriks untuk mengukur jumlah perbedaan antara dua string [12]. *Levenshtein distance* dua buah string adalah jumlah minimum operasi yang dibutuhkan untuk mengubah satu string (*source string*) menjadi string yang lain (*target string*), dimana suatu operasi melibatkan penyisipan (*insertion*), penghapusan (*deletion*), dan penggantian (*substitution*) dari suatu karakter tunggal. *Levenshtein distance* sering digunakan pada aplikasi untuk menentukan seberapa mirip, atau berbedanya dua buah string, seperti aplikasi pengecekan suatu ejaan, atau yang biasa dikenal dengan *spell checkers.*

Levenshtein distance melibatkan penggunaan matriks berukuran $(n + 1) \times (m + 1)$ dimana n dan m adalah panjang dari dua buah string. Pada Gambar 2 terdapat *pseudocode* untuk sebuah fungsi *Levenshtein distance* yang menangani 2 string, yaitu string *s* dengan panjang *m*, dan string *t* dengan panjang *n*. Penjelasan algoritma tersebut dapat dilihat pada Tabel 1.

Tabel 1 Penjelasan algoritma *levenshtein disance*

No	Keterangan
1	Set n sebagai panjang dari s. Set m sebagai panjang dari t. Jika n = 0 maka jarak (distance) = m dan selesai Jika m = 0 maka jarak (distance) = n dan selesai Bangun matriks yang berisi 0..m baris dan 0..n kolom
2	Inisialisasi baris pertama dengan 0,1,2,…,n Inisialisasi kolom pertama dengan 0,1,2,…,m
3	Periksa tiap karakter s (dari i=1 sampai i=n)

No	Keterangan
4	Periksa tiap karakter t (dari i=1 sampai i=m)
5	Jika s[i] = t[j] maka cost = 0 Jika s[i] ≠ t[j] maka cost = 1
6	Set sel d[i,j] dari matriks dengan minimum dari : • Sel di atas, ditambah 1 □ d[i-1,j] + 1 • Sel di samping kiri, ditambah 1 □ d[i,j-1] + 1 • Sel di diagonal atas-kiri , ditambah cost □ d[i-1,j-1] + cost
7	Setelah step iterasi (3,4,5,6) selesai, maka jaral Levenshtein Distance terletak pada sel d[m,n], yaitu sel di paling pojok kanan bawah

```
int d[][]; // matrix
int n; // length of s
int m; // length of t
int i; // iterates through s
int j; // iterates through t
char s_i; // ith character of s
char t_j; // jth character of t
int cost; // cost
  // Step 1
  n = s.length ();
  m = t.length ();
  if (n == 0) { return m;}
  if (m == 0) { return n;}
  d = new int[n+1][m+1];
  // Step 2
  for (i = 0; i <= n; i++) {
    d[i][0] = i;
  }
  for (j = 0; j <= m; j++) {
    d[0][j] = j;
  }
  // Step 3
  for (i = 1; i <= n; i++) {
    s_i = s.charAt (i - 1);
    // Step 4
    for (j = 1; j <= m; j++) {
      t_j = t.charAt (j - 1);
      // Step 5
      if (s_i == t_j) {cost = 0;}
      else { cost = 1;}
      // Step 6
      d[i][j] = Minimum (d[i-1][j]+1, d[i][j-1]+1, d[i-1][j-1] + cost);
    }
  }

  // Step 7

  return d[n][m];
```

Gambar 2. Pseudocode algoritma *levenshtein distance*

2.4 Receiver Operating Characteristic (ROC)

Kurva ROC adalah tool dua dimensi yang digunakan untuk menilai kinerja klasifikasi yang menggunakan dua class keputusan, masing-masing objek dipetakan ke salah satu elemen dari himpunan pasangan, positif atau negatif. Pada kurva ROC, TP rate diplot pada sumbu Y dan FP rate diplot pada sumbu X. Untuk klasifikasi data mining, nilai AUC dapat dibagi menjadi beberapa kelompok [13].

1) 0.90-1.00 = *Excellent Classification*
2) 0.80-0.90 = *Good Classification*
3) 0.70-0.80 = *Fair Classification*
4) 0.60-0.70 = *Poor Classification*
5) 0.50-0.60 = *Failure*

Kurva ini banyak digunakan untuk menilai hasil prediksi, kurva ROC adalah teknik untuk memvisualisasikan, mengatur, dan memilih pengklasifikasian berdasarkan kinerja mereka [13]. Penggunaan kurva ROC semakin popular dalam berbagai aplikasi terutama dalam bidang medis, radiologi, dan processing image. *Receiver Operating Characteristic* (ROC) adalah hasil pengukuran klasifikasi dalam bentuk 2-dimensi [14], [15]. Berikut ada empat peluang yang dapat diformulasikan dalam tabel kontingensi 2 x 2 untuk menganalisis ROC seperti pada Tabel 2.

Tabel 2 Kontingensi ROC

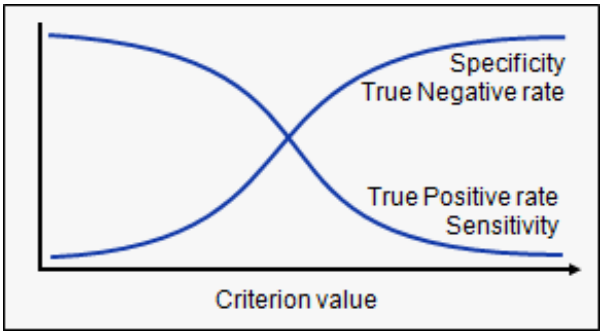
		Kelas Sebenarnya	
		Benar	Salah
Kelas Prediksi	Positif	Benar Positif	Salah Positif
	Negatif	Benar Negatif	Salah Negatif

Adapun kriteria ROC adalah sebagai berikut:

1) *True Positive Rate* disebut juga *sensitivity* (TPR)=TP/(TP+FN)
2) *True Negative Rate* disebut juga *specifity* (TNR)= TN/(TN+FP)
3) *Accuracy* = (TP+TN)/(TP+FP+TN+FN).

Dimana:

1) TP = *True Positive* yaitu klasifikasi yang dari kelas yang positif
2) FN = *False Negative* yaitu kesalahan tipe II
3) FP = *False Positive* atau kesalahan tipe I



Gambar 3. Kriteria ROC

Seperti pada Gambar 3 jika nilai kriteria yang dipilih lebih tinggi, maka bagian FP akan menurun dan specifity akan meningkat, namun TP dan sensitivity akan menurun. Sebaliknya jika nilai kriteria yang dipilih lebih rendah, maka bagian TP akan meningkat, namun bagian TN dan specificity akan menurun

Area Under Curva (AUC) adalah luas daerah di bawah kurva ROC, bila nilainya mendekati satu, maka model yang didapat lebih akurat. Berdasarkan gambar diatas maka dapat dilihat karakteristik dari AUC adalah sebagai berikut:

1) Area maksimum adalah 1
2) Jika ROC = 0,5 maka model yang dihasilkan belum terlihat optimal
3) Sedangkan jika ROC > 0,5 maka model yang dihasilkan akan lebih baik

Formula AUC berdasarkan [16] :

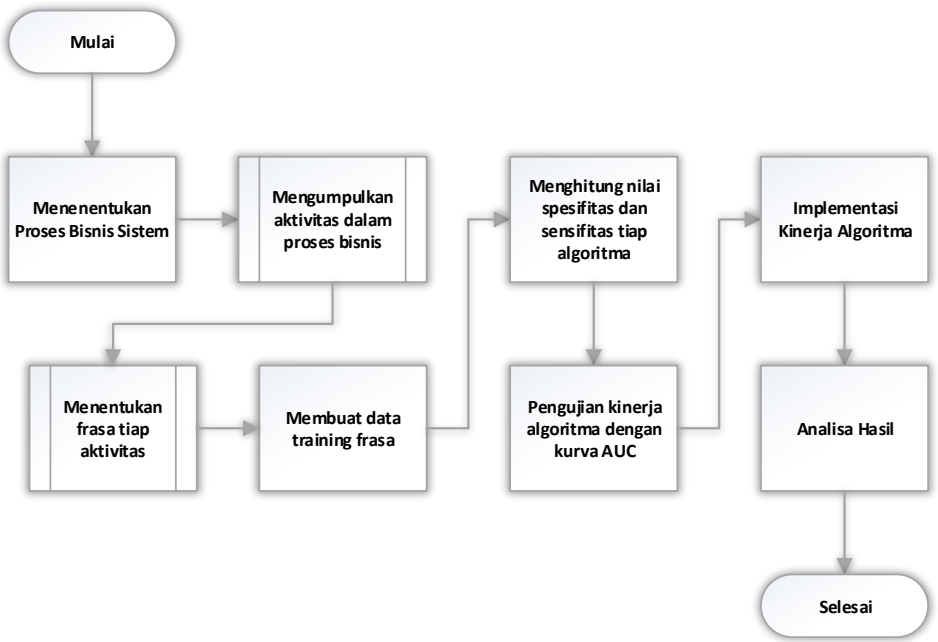
$$AUC = \frac{\sum_{i=0}^{n}\sum_{j-1}^{n} 1f(x_i^+)f\left(x_j^-\right)}{n^+n^-} \quad (1)$$

Keterangan:

1) $f(.)$ = nilai suatu fungsi
2) x^+ dan x^- = sampel positif dan negatif
3) n^+ dan n^- = jumlah sampel positif dan negatif

3. Metodologi

Penelitian ini akan membandingkan kinerja tiga algoritma kemiripan teks untuk menyelesaikan masalah pada bidang NLP. Masalah yang dimaksud adalah banyaknya kesalahan penulisan dan struktur kalimat dalam penggunaan antarmuka bahasa alami. Dengan membandingkan Algoritma kemiripan teks yang paling sering digunakan akan diketahui Algoritma mana yang paling baik kinerjanya dalam menyelesaikan permasalahan tersebut. Hasil ini diharapkan dapat membantu para peneliti kedepan dalam menyelesaikan permasalahan dibindang NLP. Gambaran proses yang dilakukan dalam mencapai tujuan yang ada pada penelitian dapat dilihat pada Gambar 4.



Gambar 4. Alur metodologi penelitian

3.1 Menentukan Proses Bisnis Sistem

Objek pada penelitian ini yaitu data frasa aktivitas dalam proses bisnis. Proses bisnis ini diambil dari Sistem Rumah Sakit dan *Open Source ERP*. Proses bisnis yang dipilih dari Sistem Rumah Sakit adalah "Permintaan Barang" yang memiliki aktivitas sebagai berikut:

1) Buat Permintaan
2) Persetujuan Permintaan
4) Penerimaan Barang

Selain itu untuk *open source* ERP proses bisnis yang dipilih adalah "Order Penjualan" yang memiliki urutan aktifitas sebagai berikut:
1) Buat Penawaran
2) Konfirmasi Penawaran
3) Buat Tagihan
4) Bayar Tagihan
5) Persiapan Pengiriman
6) Selesai Pengiriman
5) Selesai Penjualan

Tahap yang perlu dilakukan pertama kali adalah menyiapkan data masukan. Data tersebut didapat dengan menentukan beberapa frasa yang mengandung kata kerja. Frasa tersebut didapatkan dari aktivitas dalam proses bisnis dalam sebuah sistem. Kumpulan frasa yang dijadikan acuan diantaranya dapat dilihat pada Tabel 3.

Tabel 3 Kumpulan frasa kata kerja

Nomor	Frasa (kata kerja)
1	konfirmasi penawaran
2	buat penawaran
3	buat tagihan
4	bayar tagihan
5	selesai penjualan
6	selesai pengiriman
7	order penjualan
8	persiapan pengiriman
9	proses penjualan
10	permintaan barang
11	persetujuan permintaan
12	penerimaan barang
13	buat permintaan

3.2 Membuat Data Training Frasa

Dari kumpulan frasa yang ditentukan sebelumnya akan dibentuk 100 frasa ujicoba yang berisi random penulisan frasa yang benar serta penulisan frasa yang salah. Selain itu dari 100 frasa tersebut juga diberikan hasil rekomendasi frasa yang benar dan juga hasil rekomendasi frasa yang salah. Hal tersebut dilakukan agar

dapat mengetahui hasil kinerja algoritma dalam memprediksi rekomendasi frasa yang benar untuk tiap penulisan frasa yang salah. Setelah data masukkan sudah siap kemudian akan diuji tiap algoritma kemiripan teks disertai dengan prediksi hasil pada tiap frasa.

3.3 Menghitung Nilai Spesifitas dan Sensifitas Algoritma

Perhitungan nilai spesifitas dan sensifitas pada tahapan ini digunakan untuk mendeteksi kesalahan penulisan frasa kemudian tiap Algoritma yang ada akan membandingkan dengan data frasa yang mirip. Jika nilai spesifitas tinggi akan menunjukkan keabsahan kinerja algoritma dalam memprediksi rekomendasi frasa yang benar ataupun salah. Sedangkan perhitungan sensifitas digunakan untuk melihat seberapa banyak kinerja Algoritma dapat memberikan hasil prediksi frasa yang benar dan dan sedikit hasil prediksi yang salah. Hasil kriteria nilai spesifitas dan sensifitas dapat diasumsikan sebagai berikut

1) Nilai spesifitas tinggi dan sensifitas tinggi menunjukkan bahwa banyak hasil rekomendasi frasa benar yang lolos, hasil rekomendasi frasa benar yang tidak lolos dan tidak ada rekomendasi frasa salah yang tidak lolos
2) Nilai spesifitas rendah dan sensifitas tinggi menunjukkan bahwa banyak hasil rekomendasi frasa benar yang lolos dan tidak ada rekomendasi frasa salah yang tidak lolos
3) Nilai spesifitas tinggi dan sensifitas rendah menunjukkan bahwa banyak hasil rekomendasi frasa benar yang lolos namun juga ada sedikit rekomendasi frasa salah yang lolos
4) Nilai spesifitas rendah dan sensifitas rendah menunjukkan antara rekomendasi frasa benar dan salah banyak yang lolos

Nilai spesifitas dan sensifitas sangat menentukan nilai akurasi kinerja klasifikasi yang digunakan, atau dalam hal ini kinerja Algoritma dalam memprediksi rekomendasi frasa yang benar untuk memperbaiki kesalahan penulisan frasa dalam bahasa alami.

3.4 Pengujian kinerja algoritma dengan kurva AUC

Setelah mendapatkan nilai spesifitas dan sensifitas tiap algoritma, selanjutnya dilakukan perhitungan nilai akurasi pada Algoritma tersebut. Dengan memasukkan nilai spesifitas, sensifitas dan akurasi pada formula [16] akan didapatkan nilai optimal kurva AUC tiap Algoritma. Untuk menggambarkan grafik kurva AUC pada penelitian ini memanfaatkan program SPSS.

3.5 Implementasi kinerja Algoritma

Untuk mengetahui hasil dari kinerja Algoritma terbaik dapat digunakan, maka pada penelitian ini akan membuat sebuah racangan aplikasi dengan antar muka bahasa alami. Aplikasi tersebut akan dibangun dengan tekonologi web menggunakan bahasa pemrograman PHP serta basis data MySQL. Penggunaan basis data pada penelitian ini digunakan untuk menyimpan kamus frasa yang mengandung kata kerja yang dijadikan acuan dalam mengukur kinerja Algoritma. Pengujian aplikasi ini akan dilakukan pada komputer dengan spesifikasi:

1) Sistem Operasi Windows
2) Processor Core i7 2.2 Ghz
3) RAM 10 GB
4) Browser Google Chrome versi 67

Desain antarmuka aplikasi yang akan dibuat dapat dilihat pada Gambar 5. Dengan aplikasi tersebut akan diuji coba dengan memasukkan sebanyak 100 frasa aktivitas ataupun proses bisnis yang sudah ditentukan

sebelumnya. Kemudian akan dilihat hasil yang didapatkan dengan prediksi hasil. Kemudian data tersebut dimasukkan kedalam program SPSS agar dapat dilihat bentuk hasil kurva ROC.

Gambar 5. Rancangan aplikasi dengan antarmuka bahasa alami

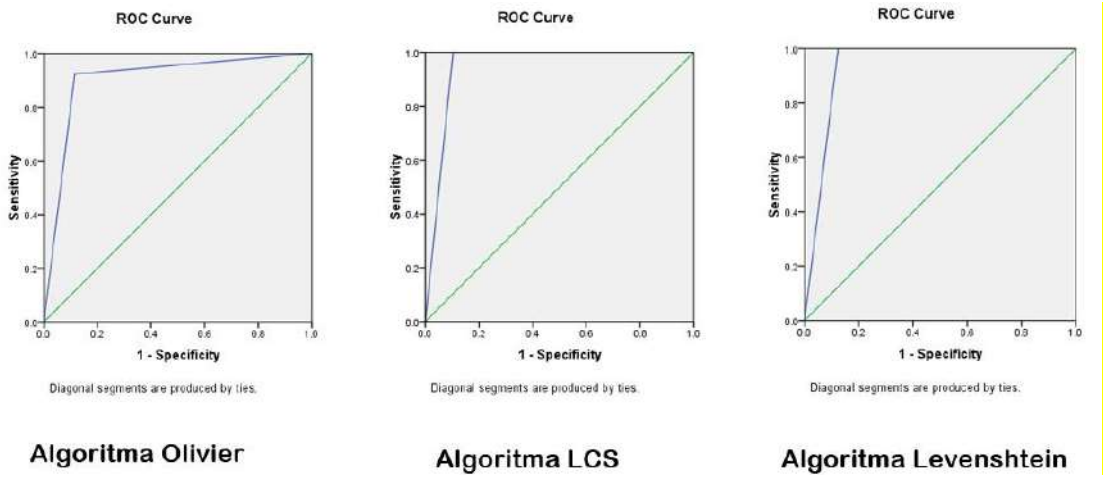
4. Hasil dan Pembahasan

Berdasarkan proses awal didapatkan bahwa nilai minimal yang optimal kurva AUC untuk algoritma pertama yaitu 0.904, dan untuk algoritma kedua yaitu 0.948 dan untuk algoritma terakhir sebesar 0.938. Hasil tersebut didapatkan dengan memasukkan sebanyak 100 frasa yang sudah ditentukan sebelumnya kedalam perangkat lunak SPSS. Data tersebut berisi kolom frasa awal, frasa prediksi dan hasil frasa rekomendasi dari tiap algoritma. Contoh bentuk tabel pengujia pada tiap algoritma dapat dilihat pada Tabel 4.

Tabel 4 Pengujian 100 frasa

#	FRASA DIUJI	FRASA REFERENSI	PREDIKSI	HASIL METODE 1	HASIL METODE 2	HASIL METODE 3
1	konfirm penawaran	konfirmasi penawaran	BENAR	BENAR	BENAR	BENAR
2	konfirm penawaran	buat penawaran	SALAH	SALAH	SALAH	SALAH
3	buat pnawaran	buat tagihan	SALAH	SALAH	SALAH	SALAH
4	buat pnawaran	buat penawaran	BENAR	BENAR	BENAR	BENAR
5	but tgihan	buat tagihan	BENAR	BENAR	BENAR	BENAR
6	but tgihan	bayar tagihan	SALAH	SALAH	SALAH	SALAH
7	selesai ppngiriman	selesai penjualan	SALAH	SALAH	SALAH	SALAH
8	selesai ppngiriman	selesai pengiriman	BENAR	BENAR	BENAR	BENAR
9	byr tagihan	bayar tagihan	BENAR	BENAR	BENAR	BENAR
10	byr tagihan	buat tagihan	SALAH	SALAH	SALAH	SALAH

Algoritma pertama diuji menggunakan kurva ROC dengan hasil nilai sensivitas sebesar 0.923 atau 92.3%, nilai spesifisitas sebesar 0.115 atau 11.5 % dan nilai kurva ROC rata-rata sebesar 0.904 atau 90.4%. Kemudian untuk algoritma kedua didapatkan hasil nilai sensitivitas sebesar 1 atau 100 % dan spesifitas sebesar 0.103 atau 10.3 % dengan nilai kurva ROC rata-rata sebesar 0.948 atau 94.8%. Sedangkan algoritma yang terakhir memiliki nilai spesifitas sebesar 1 atau 100 % dan sensitifitas sebesar 0.938 atau 93.8% Pada algoritma pertama memiliki nilai sensifitas lebih rendah dan sensifitas lebih tinggi dibandingkan algoritma kedua dan ketiga, hal tersebut menunjukkan bahwa terdapat hasil rekomendasi frasa yang salah yang masih lolos dalam klasifikasi yang menyebabkan ketika diterapkan dalam sebuah masukkan bahasa alami akan memberikan rekomendasi frasa yang tidak sesuai dengan yang dimaksut. Contohnya untuk penunlisan frasa aktivitas "buat tgihan" yang harusnya memberikan rekomendasi "buat tagihan" akan tetapi dengan nilai sensifitas yang rendah akan memberikan rekomendasi frasa aktifitas "bayar tagihan". Hal tersebut terjadi karena antara nilai kemiripan ketiga frasa rekomendasi tersebut memiliki selisih yang sedikit. Sehingga dengan nilai sensifitas yang lebih tinggi akan memberikan hasil yang lebih optimal dalam memberikan rekomendasi frasa. Untuk perbandingan kurva ROC tiap Algoritma dapat dilihat pada Gambar 6.

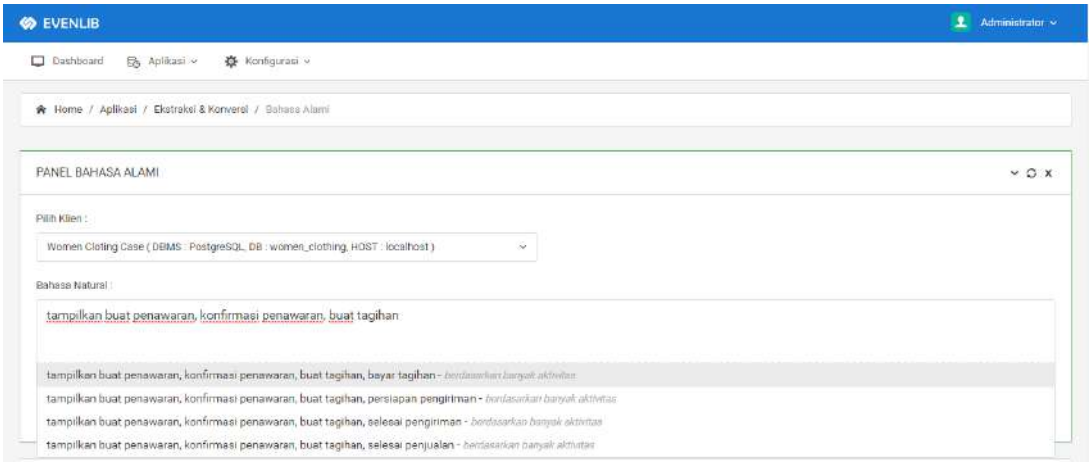


Gambar 6. Perbandingan kurva ROC tiap Algoritma

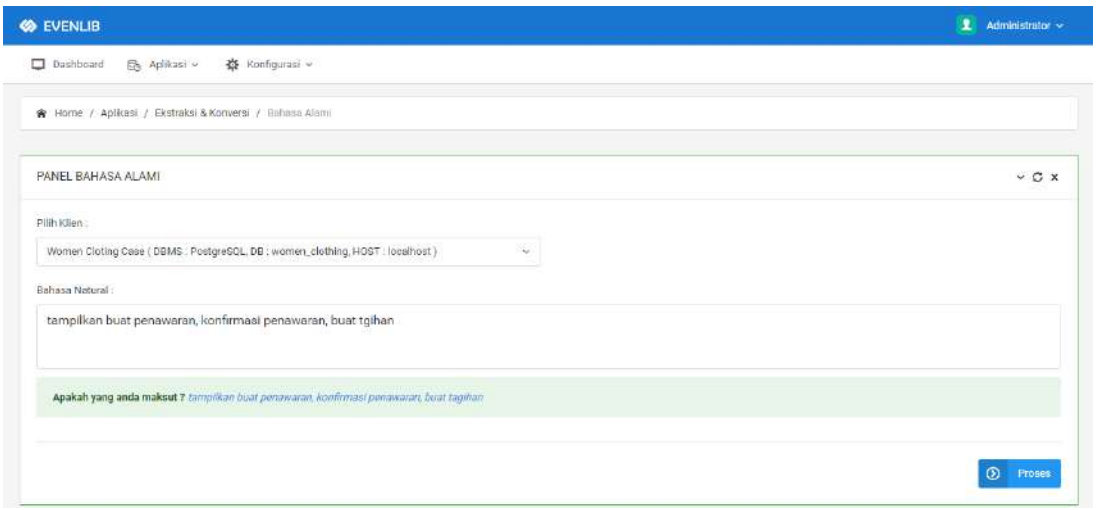
Dari ketiga hasil pengujian didapatkan bahwa Algoritma Oliver memiliki kelompok kategori kinerja "*Good Classification*" sedangkan untuk Algoritma LCS masuk kedalam kelompok kategori kinerja "*Excellent Classification*". Ketiga Algoritma tersebut cocok digunakan untuk perbaikan penulisan frasa maupun pemberian prediksi frasa dalam Bahasa alami khususnya dalam Bahasa Indonesia. Akan tetapi Algoritma LCS menunjukkan kinerja yang lebih baik dengan nilai kurva rata-rata 0.948.

Hasil pengujian tersebut juga bergantung pada nilai *threshold* minimal untuk kemiripan teks yang sudah ditentukan diawal uji coba. Semakin kecil nilai *threshold* maka akan membuat tingkat akurasi prediksi atau perbaikan frasa akan semakin tinggi namun ada beberapa frasa yang tidak dapat diproses padahal prediksi sebenarnya dapat diproses. Sebaliknya jika nilai *threshold* diperbesar maka akan memperbesar nilai error saat perbaikan maupun prediksi frasa, disisi lain juga semakin memperbanyak kemungkinan frasa yang dapat diproses.

Penelitian ini juga membuat sebuah aplikasi sederhana dengan menggunakan antarmuka Bahasa Alami untuk membuktikan kinerja algoritma tersebut dalam penerapan dunia nyata. Dalam aplikasi tersebut telah diterapkan fungsi perbaikan penulisan frasa maupun fungsi pemberian saran. Hasil dari aplikasi tersebut dapat dilihat pada Gambar 7 untuk pemberian rekomendasi frasa aktivitas saat penulisan bahasa alami. Kemudian untuk perbaikan otomatis penulisan frasa aktivitas dapat dilihat di Gambar 8.



Gambar 7 implementasi pemberian saran frasa



Gambar 8 implementasi perbaikan penulisan frasa

5. Kesimpulan

5.1 Simpulan

Penelitian ini berhasil menerapkan tiga algoritma yaitu Algoritma Oliver, Algoritma LCS dan Algoritma Levenshtein Distance untuk perbaikan penulisan maupun pemberian saran penulisan. Hasil penerapan ketiga algoritma tersebut ditunjukkan dengan sebuah aplikasi sederhana dengan antarmuka bahasa alami sebagai masukannya. Ketiga algoritma yang telah diuji menunjukkan bahwa memberikan kinerja yang baik dengan kurva rata-rata ROC diatas 0.80. Kemudian dengan membandingkan kinerja dari ketiga Aagoritma tersebut, didapatkan bahwa Algoritma LCS memiliki kinerja yang lebih baik ditunjukkan dengan nilai optimal AUC 0.948 yang termasuk kedalam "*Exellent Classification*" sesuai pernyataan [13]. Hasil tersebut membantu dalam mengatasi permasalahan perbaikan kesalahan penulisan dan pemberian saran. Hasil yang didapat pada penelitian ini diharapkan dapat digunakan untuk kebutuhan praktis maupun akademisi dalam menyelesaikan permasalahan pada bahasa alami.

Kelemahan pada penelitian ini yaitu masih ada beberapa frasa yang tidak bisa diperbaiki dengan manggunakan Algoritma Oliver maupun Algoritma LCS. Hal tersebut dikarenakan perbedaan penulisan dan frasa yang dimaksud terlalu jauh, sehingga penggunaan konsep kemiripan teks masih belum maksimal.

5.2 Saran

Saran dari penulis untuk penelititian selanjutnya yaitu penggunaan beberapa algoritma untuk kemiripan dokumen atau teks serperti *Bayesian, Cosine Similiarity* dan *Smith-waterman* ataupun algoritma lain yang diharapkan dapat memberikan hasil lebih baik dalam menyelesaikan permasalahan kesalahan penulisan frasa dalam bahasa alami.

6. Daftar Rujukan

[1] M. Tanana, et al., *A comparison of natural language processing methods for automated coding of motivational interviewing*. J. Subst. Abuse Treat., vol. 65, hal. 43–50, 2016.

[2] M. Topaz., dkk., *Automated identification of wound information in clinical notes of patients with heart diseases: Developing and validating a natural language processing application*. Int. J. Nurs. Stud., vol. 64, hal. 25–31, 2016.

[3] A. J.-P. Tixier, et al., *Automated content analysis for construction safety: A natural language processing system to extract precursors and outcomes from unstructured injury reports*. Autom. Constr., vol. 62, hal. 45–56, 2016.

[4] R. Leaman, et al., *Challenges in clinical natural language processing for automated disorder normalization*. J. Biomed. Inform., vol. 57, hal. 28–37, 2015.

[5] K. Liu, et al., *Natural language processing methods and systems for biomedical ontology learning*. J. Biomed. Inform., vol. 44, no. 1, hal. 163–179, 2011.

[6] D. D. Lewis and K. S. Jones, *Natural language processing for information retrieval*. Commun. ACM, vol. 39, no. 1, hal. 92–101, 1996.

[7] F. Reinaldha and T. E. Widagdo, *Natural Language Interfaces to Database (NLIDB): Question handling and unit conversion*. Data and Software Engineering (ICODSE), 2014 International Conference on, hal. 1–6, 2014.

[8] D. Jurafsky and J. H. Martin, *Speech and language processing*, vol. 3. Pearson London, 2014.

[9] P. Ruch and A. Gaudinat, *Comparing corpora and lexical ambiguity*. Proceedings of the workshop on Comparing corpora-Volume 9, hal. 14–19, 2000.

[10] I. Oliver, *Programming classics: implementing the world's best algorithms*. New York: Prentice Hall, 1994

[11] M. Paterson & V. Dančík, *Longest common subsequences.* International Symposium on Mathematical Foundations of Computer Science, hal. 127–142, 1994.

[12] M. Gilleland and others, *Levenshtein distance, in three flavors*. Merriam Park Softw. Httpwww Merriampark Comld Htm, 2009.

[13] F. Gorunescu, *Data Mining: Concepts, models and techniques*. vol. 12. Springer Science & Business Media, 2011.

[14] M. H. Zweig and G. Campbell, *Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine*. Clin. Chem., vol. 39, no. 4, hal. 561–577, 1993.

[15] C. E. Metz, "Basic principles of ROC analysis," Seminars in nuclear medicine, vol. 8, hal. 283–298, 1978.

[16] U. Brefeld and T. Scheffer, *AUC maximizing support vector learning*. Proceedings of the ICML 2005 workshop on ROC Analysis in Machine Learning, 2005.

Halaman ini sengaja dikosongkan