BAB II
TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Terdapat penelitian yang terkait dengan pembangunan aplikasi identifikasi *typographical error* yang pernah dilakukan sebelumnya. Pada penulisan proposal tugas akhir ini, penulis menggunakan lima pustaka sebagai bahan acuan dan bahan pembanding. Pustaka tersebut antara lain Identifikasi Kesalahan Penulisan Kata (*Typographical Error*) pada Dokumen Berbahasa Indonesia Menggunakan Metode N-gram dan Levenshtein *Distance* [1], Rancang Bangun Aplikasi Deteksi Kesalahan Penulisan Naskah Dokumen Skripsi [4] dan Studi Perbandingan Algoritma Pencarian *String* dalam Metode *Approximate String Matching* untuk Identifikasi Kesalahan Pengetikan Teks[5], *A Non-Word Error Spell Checker for Patient Complaints in Bahasa Indonesia*[6], dan *Spell Checker for Non Word Error Detection: Survey*[7].

Penelitian pertama yang digunakan sebagai pembanding adalah penelitian karya A. I. Fahma, I. Cholissodin, dan R. S. Perdana [1] dengan judul yang hampir sama yaitu "Identifikasi Kesalahan Penulisan Kata (*Typographical Error*) pada Dokumen Berbahasa Indonesia Menggunakan Metode N-gram dan Levenshtein *Distance*". Bedanya yaitu terdapat pada metode yang digunakan. Penelitian tersebut menggunakan metode *N-gram* dan Levenshtein *Distance* untuk melakukan prediksi kata, koreksi ejaan serta menentukan kandidat kata. Hasil dari penelitian ini yaitu pendekatan *dictionary lookup* dapat diterapkan dengan baik untuk proses pencarian kata yang mengalami *typographical error* dalam dokumen Bahasa Indonesia yang diinputkan dan Metode *Levenshtein Distance* dapat digunakan untuk menentukan kandidat kata yang hasilnya sesuai dengan nilai aktual yang diharapkan pengguna.

Penelitian kedua adalah penelitian karya W. W. A. Umboh, S. R. Sentinuwo dan A. M. Sambul, [4] dengan judul yang hampir sama pula dengan penelitian yang dilakukan penulis yaitu "Rancang Bangun Aplikasi Deteksi

Kesalahan Penulisan Naskah Dokumen Skripsi". Perbedaannya yaitu metode yang digunakan. Penelitian ini menggunakan metode *full text indexing* untuk mendeteksi kesalahan atau *typographical error*. Hasil dari penelitian ini adalah kesalahan yang dijadikan parameter uji berhasil diuji.

Penelitian ketiga adalah penelitian karya Y. Rochmawati dan R. Kusumaningrum yang berjudul "Studi Perbandingan Algoritma Pencarian *String* dalam Metode *Approximate String Matching* untuk Identifikasi Kesalahan Pengetikan Teks" [5]. Penelitian ini membandingkan beberapa algoritma pencarian s*tring* untuk mengidentifikasi kesalahan pengetikan teks. Hasil dari penelitian ini adalah algoritma Jaro-Winkler *Distance* dapat melakukan pengecekan kata dengan nilai MAP sebesar 0,87. Hamming *Distance* memiliki nilai MAP 0,46, Levenshtein *Distance* sebesar 0.74 dan Damerau Levenshtein sebesar 0.85. Algoritma Jaro Winkler *Distance* dinilai paling baik dibanding dengan tiga metode lainnya.

Penelitian keempat adalah penelitian karya Chanifah Indah Ratnasari, Sri Kusumadewi, dan Linda Rosita berjudul *A Non-Word Error Spell Checker for Patient Complaints in Bahasa Indonesia*[6]. Penelitian ini dilakukan untuk mengevaluasi kinerja *spell checker* untuk keluhan pasien berbahasa Indonesia. *Spell checker* ini menggunakan metode *dictionary lookup* dan algoritma Levenshtein Distance. Metode *Dictionary lookup* sebagai metode identifikasi kesalahan kata memiliki akurasi sebesar 97,59 % sedangkan algoritma Levenshtein *Distance* sebagai algoritma untuk mengoreksi kesalahan kata memiliki akurasi sebesar 94,03%. Semakin banyak kata yang disimpan dalam kamus makan akan semakin akurat hasil yang dikeluarkan.

Penelitian kelima adalah penelitian karya Hema P.H dan Sunitha C yang berjudul *Spell Checker for Non-Word Error Detection: Survey*[7]. Dalam penelitian ini penulis melakukan survey mengenai metode deteksi kesalahan kata yang digunakan oleh beberapa *tool spell checker* untuk *non-word error* pada Bahasa yang berbeda. Dalam peneltian ini dikatakan bahwa Teknik yang paling sering digunakan dalam mendeteksi kesalahan kata mencakup *non-word error* adalah *dictionary lookup* dan analisis *N-gram*.

Tabel 2.1. Tabel pembanding dengan penelitian sebelumnya

No	Peneliti	Judul Penelitian	Metode / Algortima	Batasan	Hasil
1.	Fahma (2018)	Identifikasi Kesalahan Penulisan Kata (*Typographical Error*) pada Dokumen Berbahasa Indonesia Menggunakan Metode N-gram dan Levenshtein *Distance*	*N-gram*, Levenshtein *Distance*	Dokumen berbahasa Indonesia	Pendekatan dictionary lookup dapat diterapkan dengan baik untuk proses pencarian kata yang mengalami *typographical error* dalam dokumen Bahasa Indonesia yang diinputkan. Metode Levenshtein *Distance* dapat digunakan untuk menentukan kandidat kata yang hasilnya sesuai dengan nilai aktual yang diharapkan pengguna.
2.	Umboh (2017)	Rancang Bangun Aplikasi	*Full text indexing*	Naskah dokumen	Dengan menggunakan

		Deteksi Kesalahan Penulisan Naskah Dokumen Skripsi		skripsi	*metode full text indexing* kesalahan-kesalahan seperti pengetikan judul bab, kata yang tidak sesuai KTIS, kesalahan ejaan, spasi pada karakter, kata saling sambung dengan karakter spesial maupun tanpa karakter spesial serta kesalahan penyimpanan di bagian daftar gambar berhasil diuji.
3.	Rochmawati (2015)	Studi Perbandingan Algoritma Pencarian *String* dalam Metode *Approximate String Matching* untuk Identifikasi Kesalahan	Levenshtein *Distance,* Hamming *Distance,* Damerau Levenshtein *Distance,* Jaro Winkler *Distance*	Teks berbahasa Indonesia	Algoritma atau metode Jaro Winkler *Distance* dapat melakukan pengecekan kata dengan nilai MAP sebesar 0,87. Hamming *Distance*

		Pengetikan Teks			memiliki nilai MAP 0,46, Levenshtein *Distance* sebesar 0.74 dan Damerau Levenshtein sebesar 0.85. Algoritma Jaro Winkler *Distance* dinilai paling baik dibanding dengan tiga metode lainnya.
4.	Ratnasari (2017)	*A Non-Word Error Spell Checker for Patient Complaints in Bahasa Indonesia*	*Dictionary Lookup,* Levenshtein *Distance*	*Non-word error* pada teks berbahasa Indonesia	Metode Dictionary lookup sebagai metode identifikasi kesalahan kata memiliki akurasi sebesar 97,59 % sedangkan algoritma Levenshtein *Distance* sebagai algoritma untuk mengoreksi kesalahan kata

					memiliki akurasi sebesar 94,03%. Semakin banyak kata yang disimpan dalam kamus maka akan semakin akurat hasil yang dikeluarkan.
5.	Hema (2015)	*Spell Checker for Non-Word Error Detection: Survey*	*Dictionary lookup, N-gram*	Hanya mencakup *non-word error* pada beberapa Bahasa	Dalam penelitian ini penulis melakukan survey mengenai metode deteksi kesalahan kata yang digunakan oleh beberapa *tool spell checker* untuk *non-word error* pada Bahasa yang berbeda.
6.	Londo. Grelly, 2019	Pembangunan Aplikasi Identifikasi *typographical error* pada	Jaro Winkler *Distance*	Dokumen teks berbahasa Indonesia, tidak	Penggunaan algoritma Jaro-Winkler *distance* hanya dapat

		Dokumen Berbahasa Indonesia Menggunakan Algoritma Jaro Winkler *Distance*.		mencakup pengecekan kata entitas, kesalahan secara semantic dan kontekstual	melakukan pengecekan kata secara manual berdasarkan *dataset* yang ada. Untuk melakukan pengecekan kata-kata entitas dibutuhkan penambahan model untuk *Name Entity Recognation* (NER). Penggunaan algoritma Jaro-Winkler *distance* pada penelitian ini belum dapat menentukan nilai akurasi model secara langsung.

2.2. Landasan Teori

2.2.1. *Natural Language Processing* (NLP)

Natural language processing (NLP) merupakan area penelitian serta aplikasi yang mengeksplorasi bagaimana mesin (komputer) dapat digunakan untuk mengerti dan memanipulasi bahasa alami teks ataupun pidato untuk melakukan hal berguna [8]. Bahasa alami merupakan bahasa yang digunakan oleh manusia untuk berkomunikasi setiap hari. Pada dasarnya komputer tidak mengerti bahasa manusia. NLP memungkinkan komputer mengerti dan dapat memanipulasi bahasa manusia.

Natural Language Processing (NLP) adalah cabang ilmu komputer dan teknik yang telah dikembangkan dari studi bahasa dan linguistik komputasional dalam bidang ini kecerdasan buatan [9]. NLP memiliki tujuan untuk merancang dan membangun aplikasi yang dapat memfasilitasi interaksi manusia dengan mesin melalui penggunaan bahasa alami. Menurut Pustejovky dan Stubbs dalam bukunya yang berjudul *Natural Language Annotation for Machine Learnin*g terdapat beberapa bidang utama NLP antara lain *Question Answering System* (QAS), *Summarization, Machine Translation, Speech Recognition*, dan *Document Classification*.

2.2.2. *Typographical Error*

Menurut *American Herigate Dictionary* dalam *website* https://www.thefreedictionary.com/Typographic+error, *typographical error* adalah kesalahan dalam pencetakan, penyusunan huruf atau pengetikan terutama yang disebabkan oleh pengetikan kunci yang salah pada *keyboard*. *Typographical error* merupakan kesalahan dalam pengetikan teks yang dapat menyebabkan berubahnya sebuah arti kata atau bahkan kalimat. *Typographical error* mencakup mekanis atau kesalahan tangan atau jari dan juga terjadi karena kurang-pahamnya penulis dengan ejaan yang benar.

Typographical error tidak hanya mencakup kesalahan dalam pengetikan atau pengejaan teks tetapi juga mencakup kesalahan tata bahasa yang digunakan. *Typographical error* yang berkaitan dengan kesalahan pengejaan terbagi menjadi dua jenis yaitu *real-word error* dan *non-word error*. *Real-word error* merupakan kesalahan dimana kata yang salah valid dalam bahasa yang digunakan namun tidak sesuai untuk maksud kalimat tersebut. *Non-word error* merupakan kesalahan dimana kata yang salah tersebut tidak valid dalam bahasa yang digunakan [7].

2.2.3. Jaro-Winkler *Distance*

Algoritma Jaro-Winkler *distance* merupakan salah satu jenis algoritma yang menggunakan pendekatan *string matching*. Pendekatan *string matching* merupakan suatu Teknik untuk menemukan nilai kesamaan diantara dua *string* yaitu antara *string input* dan *string* yang ada di *database* [10]. Nilai kesamaan antar *string* umumnya dalam bentuk angka yaitu 0 sampai 1.

Algoritma Jaro-Winkler *distance* merupakan pengembangan dari Jaro distance metric yang dirancang oleh Matthew A. Jaro pada tahun 1989. Awalnya Jaro *distance* digunakan sebagai pembanding dalam sensus dan *file* data kesehatan. Selanjutnya algoritma ini dimodifikasi dan disempurnakan oleh William E. Winkler. Gagasan utama dari penyempurnaan algoritma yang dilakukan Winkler adalah untuk memberikan nilai yang lebih tinggi pada dua *string* pembanding jika kedua *string* tersebut memiliki kesamaan karakter pada awal susunan *string* tersebut. Ia memiliki teori jika kesalahan pada pengetikan biasanya tidak terjadi pada awal kata-kata [11].

Jaro-Winkler *distance* memiliki nilai jarak yaitu 0 sampai dengan 1. 0 menyatakan tidak ada kesamaan diantara dua string tersebut dan 1 menyatakan kedua *string* tersebut sama persis. Jaro *distance* memiliki rumus sebagai berikut:

$$Dj = \frac{1}{3} \times \left(\frac{m}{|S_1|} + \frac{m}{|S_2|} + \frac{m-t}{m}\right)$$

| S_1 | dan | S_2 | adalah panjang masing-masing string dari string pertama dan string kedua. *m* adalah jumlah karakter yang sama di dua string tersebut, dan *t* adalah jumlah transposisi. Dua karakter dianggap cocok jika karakter pada S_1 serupa dengan karakter pada S_2 dan berada di posisi yang sama atau letaknya tidak jauh dari $\left[\frac{\max(|S_1|,|S_2|)}{2}\right] - 1$. Untuk setiap karakter yang sama dengan urutan yang berbeda maka transposisi *t* ditambah 1.

Rumus di atas merupakan perhitungan untuk Jaro *distance*. Dari rumus tersebut, Winkler melakukan improvisasi berdasarkan teorinya yaitu kesalahan pengetikan biasanya tidak terjadi pada awal sebuah kata sehingga ia menambahkan rumus Jaro *distance* yang sudah ada menjadi rumus sebagai berikut:

$$D_{jw} = D_j + l \times p \times (1 - D_j)$$

D_j adalah Jaro *distance* dari dua *string* yang akan dibandingkan, *l* adalah jumlah karakter pada awal *string* yang serupa dan *p* adalah koefisien yang bernilai antara 0 sampai 1. Setelah beberapa kali melakukan percobaan, Winkler data pada kesimpulan bahwa nilai koefisien p = 0,1 adalah nilai koefisien yang paling mendekati untuk kebanyakan kasus [12].

2.2.4. *Spelling Checker*

Spelling checker adalah aplikasi yang dapat melakukan pengecekan pada sebuah dokumen dan mencari apakah terdapat kesalahan penulisan teks di dokumen tersebut. Selanjutnya jika perlu, aplikasi tersebut dapat memberikan tanda kepada penulis jika terdapat kesalahan pada teks dokumen tersebut dengan memberi tanda pada kata yang salah dan atau memberikan saran untuk memperbaikinya [13].