BAB 2
TINJAUAN PUSTAKA

2.1. Kesalahan Ejaan

Secara garis besar, kesalahan ejaan dapat dibagi menjadi dua :

1. Kesalahan ejaan *non-word*

Kesalahan *non-word* adalah kesalahan kata dimana kata yang salah tidak terdapat dalam kamus [1]. Contoh: Saya makan **masi** goreng. Di sini kata "masi" merupakan kesalahan dari kata "nasi", dan merupakan kesalahan *non-word* karena kata "masi" tidak ada dalam kamus.

Kesalahan non-word sudah banyak diteliti dan dibuatkan algoritmanya. Algoritma-algoritma ini biasanya dinamakan *spell-checker* atau typo-checker, dan banyak terdapat di perangkat lunak pengolah kata seperti Microsoft Word, LibreOffice Writer, Ispell, Aspell, dan lain-lain [2].

2. Kesalahan ejaan *real-word*

Kesalahan *real-word* adalah kesalahan kata dimana kata yang salah ada dalam kamus, tetapi bukan merupakan kata yang dimaksud [1]. Kesalahan-kesalahan real-word biasanya mengganggu sintaks dan semantik dari suatu kalimat, sehingga dibutuhkan bantuan manusia untuk mengkoreksinya. Contoh: Saya makan basi goreng. Di sini kata "basi" merupakan kesalahan real-word, karena walaupun kata ini terdapat dalam KBBI, kata ini bukanlah kata yang dimaksud, yaitu "nasi".

Program *spell-checker* biasa tidak dapat mendeteksi kesalahan real-word, karena program tersebut hanya dapat mendeteksi kesalahan kata jika kata tersebut tidak berada di dalam kamus. Bahkan kadangkala fitur autocorrect yang terdapat pada program malah mengubah kata yang harusnya benar menjadi kata yang salah [3].

Berdasarkan penelitian-penelitian yang disurvei oleh Kukich [6], kesalahan *real-word* mencakup sekitar 25% - 40% dari seluruh kesalahan ejaan yang sudah terdokumentasi.

2.2. Penelitian Terdahulu

Secara umum terdapat dua pendekatan dalam proses koreksi dan deteksi kesalahan ejaan real-word, yaitu pendekatan berbasis Sumber Daya dan pendekatan berbasis *machine-learning* dan statistik [1]. Berikut dijelaskan kedua pendekatan tersebut beserta penelitian-penelitian yang telah dilakukan dengan masing-masing pendekatan.

2.2.1 Pendekatan Berbasis Sumber Daya

Pendekatan berbasis Sumber Daya adalah pendekatan yang menggunakan Sumber Daya leksikal buatan manusia untuk menemukan kesalahan *real-word* [1]. Salah satu penelitian yang menggunakan pendekatan ini adalah penelitian Hirst dan Budanitsky [3], yang mengkoreksi kesalahan ejaan *real-word* dengan mengembalikan kohesi leksikal pada teks. Metode ini mendeteksi kesalahan dengan memeriksa jarak semantik kalimat, yaitu dengan mengidentifikasi token-token yang secara semantik tidak terhubung dengan konteksnya dan menggantinya dengan variasi ejaan kata yang terhubung dengan konteks tersebut.

Sebelum menjalankan algoritma, dilakukan *preprocessing* dengan mengkoreksi semua kesalahan *non-word* dan meniadakan semua kata yang tidak terdapat dalam kamus atau terdapat pada daftar kata *stop-word* dalam daftar kata yang akan diperiksa, misalnya nama orang atau tempat.

Dalam langkah pertama, algoritma akan mengkonfirmasi benar setiap kata yang berhubungan semantik dengan minimal satu kata lain dalam teks. Jika kata yang dikonfirmasi mempunyai lebih dari satu makna, hanya makna yang berhubungan dengan kata-kata lain dalam teks yang akan digunakan. Setiap kata yang tidak bisa dikonfirmasi dalam langkah ini berarti tidak berhubungan dengan konteksnya, dan kemungkinan adalah kesalahan *real-word*. Kata-kata ini dijadikan "tersangka" untuk diperiksa dalam langkah selanjutnya.

Untuk menentukan apakah kata "tersangka" merupakan kesalahan *real-word*, dicarilah variasi ejaan dari kata tersebut yang lebih cocok secara konteks dibandingkan dengan kata "tersangka" tadi. Bangkitkan semua variasi ejaan dan lakukan konfirmasi untuk tiap-tiap variasi seperti pada langkah pertama. Jika

paling sedikit satu variasi terkonfirmasi, dianggap bahwa variasi tersebut lebih cocok dan kemungkinan merupakan kata yang dimaksud.

Metode ini menggunakan WordNet sebagai Sumber Daya untuk mengukur jarak keterhubungan semantik. Pengujian yang dilakukan pada korpus buatan dari 500 artikel berita *Wall Street Journal* menghasilkan *precision* 18%-25% dan *recall* 23-50%.

2.2.2 Pendekatan Berbasis Machine-Learning dan Statistik

Pendekatan berbasis *machine-learning* dan statistik adalah pendekatan yang menggunakan pemodelan bahasa statistik (Statistical Language Model, SLM). Model bahasa statistik pada intinya adalah tabel-tabel perkiraan kemungkinan kondisional untuk beberapa atau semua kata dalam sebuah bahasa yang menetapkan kemungkinan sebuah kata muncul di dalam konteks kata-kata lain [6]. Pada umumnya, pendekatan berbasis *machine-learning* dan statistik bergantung pada *confusion set* yang telah ditentukan, yaitu himpunan (biasanya pasangan) kata-kata yang biasa tertukar satu sama lain [1]. Contoh untuk *confusion set* kata “nama”: {nama, lama, hama, sama}.

Menurut O’Hearn [1], metode-metode dengan pendekatan ini mempelajari karakteristik konteks untuk setiap anggota himpunan dan mendeteksi situasi dimana suatu anggota muncul dalam konteks yang lebih tipikal untuk anggota lain. Dibanding dengan metode pendekatan Sumber Daya, metode-metode ini terbatas untuk kesalahan-kesalahan umum yang telah ditentukan. Akan tetapi metode berdasarkan Sumber Daya hanya dapat memeriksa kata konten, tidak seperti pendekatan statistik yang dapat memeriksa kata konten maupun kata fungsi.

Salah satu contoh metode dengan pendekatan *machine-learning* dan statistik adalah metode probabilitas trigram kata yang diajukan oleh Mays, Damerau dan Mercer [8] (MDM). Metode ini melihat nilai kemungkinan kalimat-kalimat yang didapat dari trigram-trigram kata. Jika nilai kemungkinan suatu kalimat lebih kecil dari nilai kemungkinan kalimat yang didapat dengan mengubah salah satu kata dengan variasi ejaannya, buat hipotesis bahwa kata asli salah dan kata variasinya yang benar. Dengan kata lain, kecilnya nilai

kemungkinan suatu kalimat mewakili adanya kelainan semantik [1], yang berarti adanya kesalahan real-word.

O'Hearn et al. [1] membandingkan metode MDM dengan metode jarak semantik milik Hirst dan Budanitsky [3], dan menemukan bahwa metode MDM lebih baik. Dengan data uji yang serupa yaitu artikel berita Wall Street Journal, metode MDM menghasilkan precision 54%-79% dan recall 25%-64%, lebih tinggi dibanding metode Hirst dan Budanitsky [1].

Untuk penelitian ini, metode yang digunakan berdasarkan pada pengembangan metode MDM yang dikemukakan oleh Samanta dan Chaudhuri [2], yaitu dengan menggunakan skor bigram dan trigram kata lokal. Walaupun lebih sederhana, metode ini menghasilkan akurasi yang sepadan dengan metode-metode lainnya. Metode ini juga dapat mendeteksi lebih dari satu kesalahan dalam satu kalimat [2]. Penjelasan terperinci untuk metode ini akan dijelaskan pada bagian berikut.

2.3 Deteksi dan Koreksi Kesalahan *Real-Word*

Metode bigram dan trigram kata lokal ini mendeteksi kesalahan ejaan real-word dengan melihat bigram dan trigram dari kiri dan kanan kata kandidat, lalu memberikan sugesti koreksi kata berdasarkan perhitungan skor *confusion set* untuk kata tersebut.

Pada awalnya metode ini membuat *confusion set* untuk setiap kata kandidat menggunakan *Levenshtein distance* dari kata-kata kamus, lalu menghitung peringkat setiap elemen dalam *confusion set* tersebut. Berdasarkan itu, kesalahan dideteksi dan diberikan sugesti kata koreksi. [2]

2.3.1 *Confusion Set* Dengan Levenshtein Distance

Untuk membuat *confusion set*, dalam metode ini digunakan *Levenshtein Distance* [25], juga dikenal sebagai *minimum edit distance*, yaitu jumlah minimum operasi edit yang diperlukan untuk mengubah suatu kata menjadi kata yang lain. Operasi edit merupakan pemasukan (*insertion*), penghapusan (*deletion*), atau penggantian (*substitution*) satu karakter dalam suatu kata. *Confusion set* dapat direpresentasikan sebagai

$$C(W^i) = \{W_1^i, W_2^i, \dots, W_j^i, \dots, W_{k_i}^i\}$$

dimana W^i adalah kata ke-i di dalam kalimat tes dan k_i adalah jumlah elemen di dalam $C(W^i)$.

2.3.2. Membuat Model N-gram

N-gram adalah rangkaian n karakter atau kata yang diekstrak dari suatu teks. Biasanya n-gram yang sering digunakan adalah bigram dan trigram, dengan nilai n masing-masing adalah 2 dan 3 [7].Contoh n-gram kata untuk kalimat "saya makan nasi goreng":

2-gram: "_ saya", "saya makan", "makan nasi", "nasi goreng", "goreng _"

3-gram: "_ saya makan", "saya makan nasi", "makan nasi goreng", "nasi goreng _".

Setelah *confusion set* dibuat, bangkitkan himpunan bigram kiri, bigram kanan, dan trigram untuk setiap anggota $C(W^i)$. Ini dibuat dengan mengambil kata di kiri, kanan, dan keduanya (untuk trigram) dari kata tersebut. Untuk W^i, maka bigram dan trigram yang terbentuk adalah:

Bigram kiri: $W^{i-1}W_j^i$

Bigram kanan: $W_j^i W^{i+1}$

Trigram: $W^{i-1}W_j^i W^{i+1}$

dimana $0 \leq j \leq k_i$.

2.3.3. Menghitung Probabilitas N-gram

Salah satu cara menghitung probabilitas n-gram adalah dengan menggunakan aturan *Markov chain*. Menurut asumsi Markov, kemungkinan suatu kejadian masa depan (kata selanjutnya) berdasarkan hanya kepada sejarah terbatas kejadian-kejadian sebelumnya (kata-kata sebelumnya). Contohnya dalam model bigram untuk kalimat sepanjang m kata $W^1, W^2, \dots, W^m$, dapat dihitung sebagai

$$P(W^1, W^2, \dots, W^m) =$$

$$P(W^1|\text{Ь})P(W^2|W^1)P(W^3|W^2) \dots P(W^m|W^{m-1})P(\text{Ь}|W^m)$$

dimana Ь menandakan kosong.

Dalam model ini probabilitas kalimat tidak dihitung. Diambil asumsi lemah bahwa kemunculan kejadian (kata) bergantung pada kejadian (kata) sebelum dan sesudahnya terlepas dari kejadian-kejadian (kata-kata) lain dalam kalimat. Dengan menggunakan *Maximum Likelihood Estimation* (MLE), didapatkan probabilitas bigram dan trigram sebagai

$$P_1\left(W_j^i \middle| W^{i-1}\right) = \frac{count(W^{i-1}W_j^i)}{\sum_{r=1}^{k_i} count(W^{i-1}W_r^i)} \quad (1)$$

$$P_2\left(W_j^i \middle| W^{i+1}\right) = \frac{count(W^{i+1}W_j^i)}{\sum_{r=1}^{k_i} count(W^{i+1}W_r^i)} \quad (2)$$

$$P_3\left(W_j^i \middle| W^{i-1}, W^{i+1}\right) = \frac{count(W^{i-1}W_j^iW^{i+1})}{\sum_{r=1}^{k_i} count(W^{i-1}W_r^iW^{i+1})} \quad (3)$$

Dalam persamaan (1), hitung P_1 untuk setiap elemen *confusion set* untuk setiap kata menggunakan jumlah bigram kiri. Penyebut di sini mewakili jumlah semua bigram dari kata sebelumnya dan satu kata dari *confusion set*. Dengan cara yang sama, hitung P_2 dengan persamaan 2. Hitung persamaannya untuk setiap elemen di setiap *confusion set* sehingga memuaskan kondisi berikut:

$$\sum_{r=1}^{k_i} P_1\left(W_j^i \middle| W^{i-1}\right) = 1$$

$$\sum_{r=1}^{k_i} P_2\left(W_j^i \middle| W^{i+1}\right) = 1$$

Untuk persamaan (3) lakukan langkah yang sama seperti persamaan (1) dan (2), bedanya kita menggunakan trigram dari kata sebelum, satu kata dari *confusion set*, dan kata sesudah. Lakukan untuk setiap elemen di setiap *confusion set* sehingga memuaskan kondisi berikut:

$$\sum_{r=0}^{k_i} P_3\left(W_j^i \middle| W^{i-1}, W^{i+1}\right) = 1$$

Gabungkan ketiga hasil persamaan (1), (2), (3) untuk menghasilkan skor bukti bahwa kata W_j^i adalah alternatif yang benar untuk W^i. Skor dihitung dengan menggunakan *weighted combination score*.

2.3.4. Weighted Combination Score

Model-model n-gram berorde tinggi maupun rendah memiliki kelebihan dan kekurangan masing-masing. N-gram berorde lebih tinggi lebih sensitif terhadap konteks, namun memiliki jumlah hitungan sedikit. N-gram berorde lebih rendah lebih terbatas mengenal konteks, tetapi jumlah hitungannya lebih banyak. Oleh karena itu model ini menggabungkan bigram dan trigram agar tidak terlalu bergantung pada suatu n-gram. [2]

Dengan menggunakan *weighted combination score*, jumlahkan ketiga hasil persamaan (1), (2), (3) yang masing-masing telah diberi bobot, menjadi:

$$Score(W_j^i) = \lambda_1 P_1(W_j^i | W^{i-1}) + \lambda_2 P_2(W_j^i | W^{i+1}) + \lambda_3 P_3(W_j^i | W^{i-1}, W^{i+1})$$

Nilai $\lambda_1, \lambda_2, \lambda_3$ dapat dihitung dengan mengoptimasi ketepatan training set. Dari penelitian [2], nilai terbaik adalah $\lambda_1 = \lambda_2 = 0.25$ dan $\lambda_3 = 0.5$. Juga, W_j^i terbatas pada $0 \leq Score(W_j^i) \leq 1$.

2.3.5. Deteksi Kesalahan dan Pemilihan Kata Sugesti

Untuk mengkonfirmasi suatu kata sebagai kesalahan *real-word*, diberikan beberapa aturan. Pertama, urutkan elemen-elemen *confusion set* berdasarkan skornya dari yang terbesar sampai yang terkecil. Gunakan kepercayaan apriori bahwa kata uji yang diamati bukanlah suatu kesalahan *real-word*. Dalam penelitiannya Mays et al. mendapatkan nilai optimum kepercayaan ini sebagai 0.99 [8] yang akan digunakan dalam metode ini juga. Dengan kata lain, dipercaya bahwa kata yang diuji dapat menjadi kesalahan kata *real-word* dalam 1% kasus. Nilai ini digunakan untuk normalisasi skor dalam algoritma deteksi kesalahan *real-word* yang akan dijelaskan di bawah ini.

Misalkan W^i adalah kata uji dalam suatu kalimat. Berdasarkan hasil skor yang didapat, buat keputusan berdasarkan *pseudocode* berikut:

Begin
if $Score(W^i) = 0$
    nyatakan W^i sebagai kesalahan *real-word*
else
    if $Score(W^i) < 0.01 *$Skor tertinggi elemen *confusion set*
        nyatakan W^i sebagai kesalahan *real-word*
    else
        W^i benar
    end if
end if
End

2.4 Pemodelan

Skema pemodelan data adalah metode yang memungkinkan untuk memodelkan atau menggambarkan *database* pada aplikasi pengkoreksi kesalahan ejaan *real-word* yang akan dibangun.

2.4.1 Entity Relationship Diagram

ERD merupakan suatu model jaringan yang menggunakan susunan data yang disimpan pada sistem secara abstrak. Tujuan dari *entity relationship* adalah untuk menunjukkan objek data dan relationship yang ada pada objek tersebut. Di samping itu model ER ini merupakan salah satu alat untuk perancangan dalam basis data. [10]

Komponen-komponen ERD adalah sebagai berikut:

1. Entitas

   Entitas (entity) menunjukkan objek-objek dasar yang terkait di dalam sistem. Objek dasar dapat berupa orang, benda atau hal lain yang keterangannya perlu disimpan dalam basis data.

2. Atribut

   Atribut sering juga disebut sebagai properti merupakan keterangan-keterangan yang terkait pada sebuah entitas yang perlu disimpan sebagai basis data. Atribut berfungsi sebagai penjelas sebuah entitas.

   Jenis-jenis Kardinalitas :

   a. Satu ke Satu (1 : 1)

   Yaitu perbandingan antara entity pertama dengan entity kedua berbanding satu berbanding satu.

   b. Satu ke Banyak (1 : N)

   Yaitu perbandingan antara entity pertama dengan entity kedua berbanding satu berbanding banyak.

   c. Banyak ke Satu (N : 1)

   Yaitu perbandingan antara entity pertama dengan entity kedua berbanding banyak berbanding satu.

   d. Banyak ke Banyak (N : N)

   Yaitu perbandingan antara entity pertama dengan entity kedua berbanding banyak berbanding banyak.

3. Relasi

   Adalah karakteristik dari *entity* atau *relationship* yang menyediakan penjelasan detail tentang entity atau relationship tersebut. Relasi atau hubungan adalah kejadian atau transaksi yang terjadi di antara dua entitas yang keterangannya perlu disimpan dalam basis data.

4. Derajat Relasi (Kardinalitas)

   Kardinalitas relasi menunjukkan maksimum entitas yang dapat berelasi dengan entitas pada himpunan entitas yang lain. Kardinalitas relasi yang terjadi di antara dua himpunan entitas (misalkan A dan B) dapat berupa satu ke satu (*one to one*), satu ke banyak (*one to many*), banyak ke satu (*many to one*), dan banyak ke banyak (*many to many*).

2.4.2 Flowchart

Flowchart adalah bagan-bagan yang mempunyai arus yang menggambarkan langkah-langkah penyelesaian suatu masalah. *Flowchart* merupakan cara penyajian suatu algoritma. *Flowchart* adalah penggambaran secara grafik dari langkah-langkah dan urutan-urutan prosedur dari suatu program. *Flowchart* biasanya mempermudah penyelesaian suatu masalah khususnya masalah yang perlu dipelajari dan dievaluasi lebih lanjut [10].

Tujuan membuat *Flowchart* adalah menggambarkan suatu tahapan penyelesaian masalah secara sederhana, terurai, rapi dan jelas menggunakan simbol-simbol standar.

2.4.3 Diagram Konteks

Diagram Konteks adalah sebuah diagram sederhana yang menggambarkan hubungan antara *entity* luar, masukan dan keluaran dari sistem. Diagram konteks direpresentasikan dengan lingkaran tunggal yang mewakili keseluruhan sistem [10]. Diagram konteks dimulai dengan penggambaran terminator, aliran data, aliran kontrol penyimpanan, dan proses tunggal yang menunjukan keseluruhan sistem. Ada beberapa aturan dalam menggambarkan diagram konteks, berikut di bawah ini menunjukan aturan tersebut.

1. Menggunakan hanya satu simbol proses
2. Memberi label simbol proses tersebut untuk menggambar seluruh sistem, biasanya berupa kata kerja di tambah objek
3. Tidak memberi nomor pada simbol proses
4. Menyertakan semua terminator dari sistem
5. Menunjukan semua arus data antara terminator dan sistem

2.4.4 DFD (Data Flow Diagram)

DFD adalah suatu model logika data atau proses yang dibuat untuk menggambarkan darimana asal data dan kemana tujuan data yang keluar dari sistem, dimana data disimpan, proses apa yang menghasilkan data tersebut dan interaksi antara data yang tersimpan dan proses yang dikenakan pada data tersebut [10].

1. Arus Data (*Data Flow*)

   Menunjukan arus dari data yang dapat berupa masukan untuk sistem atau dari proses sistem.

2. Proses

   Proses adalah kegiatan yang dilakukan oleh orang, mesin atau komputer dari hasil arus data yang masuk ke dalam proses untuk dihasilkan arus data yang akan keluar dari proses.

3. Kesatuan Luar (*External Entity*)

   Kesatuan luar merupakan kesatuan di lingkungan luar sistem yang dapat berupa orang, organisasi atau sistem lain yang akan memberikan masukan atau menerima keluaran dari sistem.

4. File

   Kumpulan data yang disimpan dengan cara tertentu. Data yang mengalir disimpan dalam file. Aliran data di-update atau ditambahkan ke dalam file.

2.4.5 Kamus Data

Kamus data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukan (*input*) dan keluaran (*output*) dapat dipahami secara umum [10]. Kamus data mendefinisikan elemen data dengan menjelaskan arti aliran data dan penyimpanan data dalam DFD, mendeskripsikan komposisi paket data yang bergerak melalui aliran, mendeskripsikan komposisi penyimpanan data, menspesifikasikan nilai dan satuan yang relevan bagi penyimpanan dan aliran, mendeskripsikan hubungan detil antar penyimpanan.

Kamus data berfungsi untuk membantu pelaku sistem untuk mengartikan alokasi secara detail dan mengorganisasikan semua elemen data yang

digunakan dalam sistem secara persis sehingga baik pemakai atau penganalisis sistem mempunyai dasar pengertian yang sama tentang masukan, keluaran, penyimpanan dan proses.

Kamus data biasanya berisi:

- Nama-nama dari data
- Digunakan pada – merupakan proses-proses yang terkait data
- Deskripsi – merupakan deskripsi data
- Informasi tambahan – seperti tipe data, nilai data, batas nilai data, dan komponen yang membentuk data.

Kamus data memiliki simbol untuk menjelaskan informasi tambahan seperti pada tabel 2.3 berikut.

Tabel 2.1 Kamus Data

Simbol	Keterangan
=	disusun atau terdiri dari
+	Dan
[|]	baik ... atau....
{}n	n kali diulang/ bernilai banyak
()	data opsional
...	batas komentar

2.5 Bahasa Pemrograman

Bahasa pemrograman merupakan kumpulan aturan yang disusun sedemikian rupa sehingga memungkinkan pengguna komputer membuat program yang dapat dijalankan dengan aturan tersebut [11]. Beberapa contoh bahasa pemrograman yang banyak dipakai, yaitu C, C++, C#, Pascal, Java, JavaScript, PHP, SQL, dan Python. Bahasa pemrograman yang digunakan pada pembangunan aplikasi pengkoreksi kesalahan ejaan *real-word* ini adalah PHP, dan JavaScript.

2.5.1 PHP

Hypertext Prepocessor (PHP) adalah skrip yang menyatu dengan kode HTML dan berjalan pada sisi server (server-side). Artinya semua sintaks yang kita berikan akan sepenuhnya dijalankan pada server secangkan yang dikirimkan ke browser hanya hasilnya [12]. Keuntungan menggunakan PHP adalah tidak

diedarkannya kode sumber yang membangun PHP ke sisi klien sehingga kerahasiaan kode dapat dilindungi. PHP berjalan pada sisi server, sehingga untuk dapat menggunakannya kita harus mengaktifkan web server terlebih dahulu baik offline maupun online.

Deklarasi variabel dengan tipe data tidak diperlukan dalam PHP seperti bahasa pemrograman pascal. Setiap variabel yang terbentuk dalam program dianggap bertipe variant, dengan kata lain dapat menampung tipe data dengan jenis apapun (*dynamic variable*). Sistem yang dibangun menggunakan PHP versi 5.6.19.

2.5.2 JavaScript (AJAX & jQuery)

HTTP (*Hypertext Transfer protocol*) merupakan protokol utama dibalik WWW (*World Wide Web*). Protokol ini yang menangani semua permintaan halaman website, gambar, link, dan form. Dengan adanya HTTP memungkinkan sebuah cara yang terstandarisasi untuk komputer berkomunikasi dengan komputer lainnya. HTTP akan menspesifikasikan bagaimana klien meminta data, dan bagaimana server memproses permintaan tersebut. HTTP memiliki tiga tipe permintaan dasar : POST, GET, dan HEAD [13]. Permintaan HEAD digunakan untuk memverifikasi bahwa sebuah halaman web yang diminta masih tersedia. Sebuah mesin pencari biasanya menggunakan permintaan HEAD untuk memastikan bahwa hasil pencariannya masih aktif. Jika permintaan HEAD gagal, maka mesin pencari tersebut bisa langsung menghapus hasil pencarian dari *database* yang dimiliki oleh mesin pencari tersebut.

Javascript merupakan salah satu bahasa pemrograman yang pertama kali dikembangkan oleh Netscape yang digunakan dalam sebuah browser atau perambah *web* [14]. *JavaScript* dibuat agar mudah diintegrasikan ke dalam program dan aplikasi berbasiskan web. AJAX merupakan singkatan dari *Asynchronous JavaScript* and XML merupakan bagian dari JavaScript yang bertugas atau berfungsi untuk memodifikasi sebagian atau semua dari sebuah halaman web secara dinamis pada saat dibutuhkan saja [15].

2.6 Pengelola *Database*

Database merupakan kumpulan file-file yang berhubungan secara logis dan digunakan secara rutin pada operasi-operasi sistem informasi manajemen. Semua *database* umumnya berisi elemen-elemen data yang disusun ke dalam file-file yang diorganisasikan berdasarkan sebuah skema atau struktur tertentu, tersimpan di hardware komputer dan dengan software untuk melakukan manipulasi data untuk kegunaan tertentu. Jadi, suatu *database* adalah menunjukkan suatu kumpulan tabel yang dipakai dalam suatu lingkup perusahaan atau instansi untuk tujuan tertentu [16].

Software yang digunakan untuk mengelola dan permintaan panggilan (*query*) basis data yang disebut sistem manajemen *database* (*database management system*, DBMS).

Dalam pengelolaan *database* peneliti menggunakan MySQL dengan menggunakan bahasa SQL. Penjabaran MySQL dan SQL akan dijelaskan seperti berikut.

2.6.1 SQL (Structured Query Language)

Structured Query Language adalah bahasa yang digunakan untuk mengelola data pada RDBMS. SQL digunakan untuk mengambil, menambahkan, dan memanipulasi data dari sebuah basis data relasional [17].

Bahasa SQL terbagi menjadi dua kategori yaitu kategori pertama *schema statements* kategori pernyataan *(statements)* untuk membuat objek basis data berupa tabel, index, constraints, dan sebagainya. Kategori kedua (SQL *data statements*) yaitu pernyataan yang digunakan untuk memanipulasi dan mengambil data yang tersimpan dalam basis data.

2.6.2 MySQL

MySQL merupakan *database server* yang bersifat *multiuser* dan *multi-threaded*. *SQL* adalah bahasa *database* standar yang memudahkan penyimpanan, pengubahan dan akses informasi. Pada MySQL dikenal dengan istilah *database* dan tabel. Tabel sendiri merupakan sebuah struktur data dua dimensi yang terdiri dari baris-baris record dan kolom [17]. MySQL termasuk salah satu *database* open source yang paling banyak digunakan karena selain gratis MySQL pun

mempunyai banyak dukungan bahasa pemrograman dan aplikasi sebagai solusi *database*. Dalam penelitian ini MySQL digunakan untuk membuat dan mengola *database* korpus kamus dan korpus n-gram beserta isinya.

2.7 Perangkat Lunak Pembangunan

Perangkat lunak adalah istilah khusus untuk data yang diformat, dan disimpan secara digital, termasuk program komputer, dokumentasinya, dan berbagai informasi yang bisa dibaca, dan ditulis oleh komputer. Dengan kata lain, bagian sistem komputer yang tidak berwujud.

Dalam pembangunan dan pengujian aplikasi pengkoreksi kesalahan ejaan *real-word*, peneliti menggunakan XAMPP sebagai *web server* untuk pembangunan dan pengujian aplikasi pengkoreksi kesalahan ejaan *real-word* yang dibangun, *Web Browser*, dengan Sublime Text 3 sebagai penyunting kode bahasa pemrograman.

2.7.1 XAMPP

XAMPP adalah aplikasi *web server* yang berdiri sendiri terdiri dari Apache HTTP Server, MySQL *database* dan PHP. XAMPP juga dilengkapi dengan fitur manajemen *database* PHPMyAdmin seperti pada server hosting yang terhubung internet [18]. Untuk pembangunan sumber daya jenis kata, penulis menggunakan XAMPP sebagai *webserver*. XAMPP memiliki kelebihan untuk bisa berperan sebagai server web Apache dalam melakukan simulasi pengembangan web. Tool pengembangan web ini mendukung teknologi web populer seperti PHP, MySQL dan Perl.

Dalam penelitian ini XAMPP digunakan sebagai server *localhost* untuk membuka aplikasi koreksi kesalahan ejaan *real-word* secara *offline.*

2.7.2 Web Browser

Web Browser adalah layanan internet untuk menjelajahi atau berselancar di dunia maya dengan menggunakan jaringan internet [19]. Fungsi dari *Web Browser* sendiri adalah untuk menampilkan halaman web atau melakukan interaksi dengan dokumen yang disediakan server.

Dalam penelitian ini penulis menggunakan Mozilla Firefox versi 46.0.1 dan Google Chrome versi 57.0.2987.133 sebagai *Web Browser* untuk pembangunan dan pengujian aplikasi pengkoreksi kesalahan ejaan *real-word* yang dibangun.

2.7.3 Sublime Text 3

Sublime Text 3 adalah program aplikasi yang berguna untuk mengedit teks dan skrip kode pemrograman seperti HTML, CSS, PHP, XML, Java, dan lain-lain yang bekerja pada sistem operasi windows. Kelebihan Sublime jika dibandingkan dengan Notepad bawaan Windows adalah memiliki kelengkapan fitur untuk mempermudah pengguna saat mengedit kode termasuk saat mengedit kode HTML dan kode CSS. Dalam penelitian ini Sublime Text 3 digunakan untuk mengedit kode program.