BAB 2
TINJAUAN PUSTAKA

2.1 Kesalahan Ejaan

Kesalahan ejaan kata dapat dibagi menjadi dua yaitu kesalahan ejaan *real-word* dan kesalahan ejaan *non-word*.[1]

1. Kesalahan ejaan *non-word*

Kesalahan ejaan non-word adalah kesalahan ejaan kata dimana kata yang mengalami kesalahan kata tidak terdapat di dalam kamus[4] atau dengan kata lain tidak memiliki makna. Contoh : "Banyak **paku** bertebaran" menjadi "Banyak **caku** bertebaran". Kata "caku" merupakan kesalahan kata *non-word,* karena kata tersebut tidak terdapat di dalam kamus. Algoritma untuk kesalahan *non-word* sudah banyak diteliti contohnya adalah algoritma *spell-checker* atau *typo-checker.* Algoritma-algoritma ini diaplikasikan pada banyak perangkat lunak contohnya fitur *auto-correct* pada *Microsoft-Word*.

2. Kesalahan ejaan *real-word*

Kesalahan ejaan *real-word* merupakan kesalahan ejaan kata dimana kata yang salah terdapat di dalam kamus, tetapi jika dilihat secara konteks kalimat menjadi tidak sesuai karena bukan merupakan kata yang dimaksud[4]. Contoh : "Banyak **paku** bertebaran" menjadi "banyak **baku** bertebaran". Kata "baku" merupakan kesalahan kata *real-word* karena kata tersebut salah secara konteks kalimat tetapi ada di dalam kamus.

Algoritma yang dibuat untuk mengatasi kesalahan *non-word* seperti *typo-checker* tidak dapat menangani kesalahan *real-word* karena algoritma tersebut hanya dapat menangani kata yang tidak terdapat di dalam kamus. Menurut penelitian yang dilakukan oleh Kukich, kesalahan ejaan *real-word* mencakup sekitar 25%-35% dari seluruh kesalahan ejaan kata yang terdokumentasi[5].

2.2 Penelitian Terdahulu

Kesalahan ejaan real-word secara umum dapat dibagi menjadi dua menurut cara pendeketannya didalam sistem deteksi dan koreksi kesalahan kata yaitu pendekatan yang berbasis sumber daya dan yang kedua berbasis *Machine Learning* dan statistik [4]. Pendekatan berbasis sumber daya adalah pendekatan yang menggunakan sumber daya leksikal untuk mengetahui konteks dari kalimat yang diperiksa apakah sudah tepat. Penelitian yang menggunakan jenis pendeketan sumber daya leksikal adalah Hirst dan Budanitsky[4] dengan memeriksa jarak semantik kalimat yaitu dengan melihat hubungan token-token yang terdapat dalam kalimat apakah berhubungan atau tidak dengan konteks kalimat. Jika tidak maka akan dibangkitkan variasi ejaan dari kata yang sesuai dengan konteks kalimat tersebut.

Pendekatan yang kedua adalah pendekatan berbasis *Machine Learning* dan statistik. Pendeketan ini menggunakan pemodelan bahasa statistik yang merupakan proses menetapkan kemungkinan suatu kata muncul di di dalam konteks kalimat lain yang diletakkan melalui tabel-tabel perkiraan. Pendekatan berbasis *Machine Learning* dan statistik ini bergantung pada *confusion set* yang dibentuk, yaitu himpunan kata-kata yang biasanya tertukar satu sama lain. Contoh *cofusion set* untuk kata “parah” = { barah, carah, darah, karah }.

Salah satu metode yang menggunakan pendekatan *Machine Learning* dan statistik adalah metode yang diajukan oleh Mays, Demerau dan Mercer(MDM). Metode ini menilat nilai probabilitas dari trigram-trigram kata yang terbentuk. Jika perubahan salah satu kata dengan variasi ejaan menghasilkan nilai yang lebih kecil dibandingkan dengan tidak mengubahnya maka dapat ditentukan hipotesis bahwa kata asli salah dan kata variasi yang benar. Arti kecilnya nilai probabilitas trigram kata menandakan adanya kesalahan *real-word*. [6]

O’Hearn membandingkan metode MDM dengan metode dari Hirst dan Budanitsky[4] yang menggunakan pendeketan sumber daya leksikal dengan data uji yang sama dan menemukan bahwa metode MDM lebih baik. Metode MDM menghasilkan *precision* 54%-79% dan *recall* 25%-64% sedangkan

metode Hirst dan Budanitsky [4] menghasilkan *precision* 18%-25% dan *recall* 23%-50% .

Muhammad Aburizal melakukan penelitian terkait kesalahan ejaan *real-word* Bahasa Indonesia dengan menggunakan metode MDM yang telah dikembangkan lebih lanjut oleh Samanta dan Chaudhuri, yaitu dengan menggunakan perhitungan probabilitas bigram dan trigram. Metode ini dapat mendeteksi kesalahan di lebih dari satu kalimat.[6] Hasil dari penelitian yang dilakukan oleh Aburizal mendapatkan nilai akurasi sebesar 11% jauh lebih kecil dibandingkan penelitian Samantha dan Chauduri sebesar 80%. Oleh karena itu Fernando Sihole melakukan penelitian dengan menggunakan beberapa metode *smoothing* sebagai upaya meningkatkan nilai akurasi deteksi dan koreksi kesalahan *real-word*. Pada penelitian ini akan mengembangkan penelitian Aburizal dan Fernando dengan mengambil metode *smoothing Kneser-Ney* yang mempunyai performa paling baik yang dikembangkan oleh Stanley dan Goodman.

2.3 Deteksi dan Koreksi Kesalahan *Real-Word*

Metode bigram dan trigram ini melakukan pendeteksian dan koreksi kesalahan dengan melihat bigram dan trigram dari kiri dan kanan kata kandidat. Selanjutnya akan dilakukan perhitungan skor *confusion set* sehingga akan diberikan sugesti koreksi kata.

2.3.1 *Confusion Set* Dengan *Levenshtein Distance*

Untuk bisa membuat *confusion set* diperlukan perhitungan *Levenshtein Distance* yaitu untuk menghitung jarak minimum suatu kata (*minimum edit distance*), jumlah operasi *edit* yang dibutuhkan untuk bisa mengubah kata menjadi kata yang lain. Operasi edit disini adalah pemasukan (*insertion*), pergantian (*substitution*), dan penghapusan (*deletion*). Berikut adalah perhitungan *Levenshtein Distance*.

$$(i,j) = min\begin{Bmatrix} d(i-1,j)+1, d(i,j-1)+ \\ 1, d[i-1,j-1]+cost \end{Bmatrix}$$

Dengan i adalah baris dan j adalah kolom. Hasil perhitungan didapatkan pada hasil perhitungan di baris dan kolom terakhir matriks. Kata-kata yang berhak untuk masuk ke dalam daftar *confusion set* adalah kata yang mempunya nilai *levenshtein distance* maksimal 1.

Confusion set dapat di representasikan sebagai berikut.

$$C(W^i) = \{c_1^i, c_2^i, \dots, c_j^i, c_{k_i}^i\}$$

Dimana W^i adalah kata ke-i di dalam kalimat dan k_i adalah jumlah elemen yang terdapat dalam $C(W^i)$.

2.3.2 Membuat Model N-Gram

N-Gram adalah urutan kata yang didapatkan dari teks. Biasanya n-gram yang digunakan adalah unigram, bigram dan trigram . Dengan nilai n dari unigram adalah 1, n dari bigram adalah 2 dan nilai n dari trigram adalah 3. Contoh n-gram untuk kalimat “banyak paku bertebaran”:

1-gram(unigram)= “banyak”, “paku”, “bertebaran”

2-gram(bigram)= “_banyak”, “banyak paku”, “paku bertebaran”, “bertebaran_”

3-gram(trigram)= “_banyak paku”, “banyak paku bertebaran”,
“paku bertebaran_”

Model N-Gram dibuat dengan membangkitkan himpunan bigram kiri, bigram kanan dan trigram untuk setiap anggota confusion set. Kata ke-i dalam anggota confusion set dilambangkan dengan c_j^i. Maka unigram, bigram dan trigram yang terbentuk adalah:

Unigram : c_j^i

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

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

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

2.3.3 Perhitungan Probabilitas N-Gram

Probabilitas N-Gram dihitung menggunakan *Maximum Likehood Estimation* (MLE) yaitu mengambil asumsi bahwa kemunculan kata bergantung kepada kemunculan kata sebelum dan kemunculan kata sesudahnya dalam suatu kalimat. Maka didapatkan probabilitas unigram, bigram dan trigram sebagai berikut.

Probabilitas dari unigram c_j^i

$$P_0(c_j^i) = \frac{count(c_j^i)}{\sum_r^{k_i} count(c_r^i)}$$

$$P_1(c_j^i | W^{i-1}) = \frac{count(W^{i-1}c_j^i)}{\sum_r^{k_i} count(W^{i-1}c_r^i)}$$

$$P_2(c_j^i | W^{i+1}) = \frac{count(c_j^i W^{i+1})}{\sum_r^{k_i} count(c_r^i W^{i+1})}$$

$$P_3(c_j^i | W^{i-1}, W^{i+1}) = \frac{count(W^{i-1}c_j^i W^{i+1})}{\sum_r^{k_i} count(W^{i-1}c_r^i W^{i+1})}$$

Dimana P_1 merupakan probabilitas untuk bigram kiri, P_2 merupakan probabilitas untuk bigram kanan dan P_3 merupakan probabilitas untuk trigram. Tetapi dengan menggunakan rumus diatas banyak terdapat hasil probabilitas yang bernilai 0. Maka dari itu muncul metode-metode *smoothing* yang menambahkan *pseudocount* pada probabilitas yang bernilai 0. Metode *smoothing Kneser-Ney* menghitung probabilitas menggunakan perhitungan kemunculan kata pada *confusion set* yang lebih dari 0. Berikut merupakan contoh perhitungan probabilitas menggunakan metode *smoothing Kneser-Ney.*Maka rumus-rumus probabilitas n-gram menjadi:

1. Kneser-Ney

Bigram Kiri =

Perhitungan metode *Smoothing Kneser-Ney* untuk bigram kiri menggunakan persamaan

$$P_1\left(c_j^i \middle| W^{i-1}\right) = \frac{max\left\{count\left(W^{i-1}c_j^i\right) + 0.01 - D, 0\right\}}{\sum_{r=1}^{k_i} count\left(W^{i-1}c_r^i\right) + 0.01 \times V_j} + (1-\lambda) \times \frac{N_{1+}\left(c_j^i\right)}{\sum_{r=1}^{k_i} N_{1+}\left(c_r^i\right)}$$

Dimana,

$$(1-\lambda) = \frac{D}{\sum_{r=1}^{k_i} count\left(W^{i-1}c_r^i\right) + 0.01 \times V_j} N_{1+}\left(W^{i-1}c_j^i\right)$$

Nilai D (*Discount*) diasumsikan sebagai $0 \leq D \leq 1$ [3]. Nilai D yang digunakan adalah 0.01 dengan pertimbangan tidak terlalu mempengaruhi perhitungan skor pada tahap selanjutnya

$$N_{1+}\left(c_j^i \middle| W^{i-1}\right) = \left|\left\{W^i : count\left(W^{i-1}c_j^i\right) > 0\right\}\right|$$

$N_{1+}\left(c_j^i \middle| W^{i-1}\right)$ merupakan notasi dari nilai kata W^i dimana jumlah kemunculan kata dari bigram kiri dalam anggota *confusion set* yang lebih dari 0

$$N_{1+}(c_j^i) = \left|\left\{W^i : count\left(c_j^i\right) > 0\right\}\right|$$

$N_{1+}\left(c_j^i\right)$ merupakan notasi dari nilai kata W^i dimana jumlah kemunculan kata dari unigram dalam anggota *confusion set* yang lebih dari 0.

$\sum_{r=1}^{k_i} N_{1+}\left(c_r^i\right)$ merupakan notasi dari jumlah semua anggota *confusion set* data uji dari unigram yang lebih dari 0.

Nilai V_j merupakan jumlah anggota *confusion set* yang terbentuk.

Bigram Kanan =

Perhitungan metode *Smoothing Kneser-Ney* untuk bigram kanan menggunakan persamaan

$$P_2(c_j^i|W^{i+1}) = \frac{max\{count(W^{i+1}c_j^i) + 0.01 - D, 0\}}{\sum_{r=1}^{k_i} count(W^{i+1}c_r^i) + 0.01 \times V_j} + (1-\lambda) \times \frac{N_{1+}(c_j^i)}{\sum_{r=1}^{k_i} N_{1+}(c_r^i)}$$

Dimana,

$$(1-\lambda) = \frac{D}{\sum_{r=1}^{k_i} count(c_r^i W^{i+1}) + 0.01 \times V_j} N_{1+}(c_j^i|W^{i+1})$$

Nilai D (*Discount*) sama seperti perhitungan pada bigram kiri yaitu diasumsikan sebagai $0 \leq D \leq 1$ [3]. Nilai D yang digunakan adalah 0.01 dengan pertimbangan tidak terlalu mempengaruhi perhitungan skor pada tahap selanjutnya.

Nilai V_j merupakan jumlah anggota *confusion set* yang terbentuk.

$$N_{1+}(c_j^i|W^{i+1}) = |\{W^i : count(c_j^i|W^{i+1}) > 0\}|$$

$N_{1+}(c_j^i|W^{i+1})$ merupakan notasi dari nilai kata W^i dimana jumlah kemunculan kata dari bigram kanan dalam anggota *confusion set* yang lebih dari 0

$$N_{1+}(c_j^i) = |\{W^i : count(c_j^i) > 0\}|$$

$N_{1+}(c_j^i)$ merupakan notasi dari nilai kata W^i dimana jumlah kemunculan kata dari unigram dalam anggota *confusion set* yang lebih dari 0.

$\sum_{r=1}^{k_i} N_{1+}(c_r^i)$ merupakan notasi dari jumlah semua anggota *confusion set* data uji dari unigram yang lebih dari 0.

Trigram=

Persamaan yang digunakan untuk menghitung probabilitas dari trigram dengan menggunakan metode *kneser-ney* yaitu

$$P_3\left(c_j^i \middle| W^{i-1}, W^{i+1}\right) = \frac{max\left\{count\left(W^{i-1}c_j^i W^{i+1}\right) + 0.01 - D, 0\right\}}{\sum_{r=1}^{k_i} count\left(W^{i-1}c_r^i W^{i+1}\right) + 0.01 \times V_j} + (1-\lambda) \times \frac{\left(\frac{N_{1+}\left(c_j^i \middle| W^{i-1}\right)}{\sum_{r=1}^{k_i} N_{1+}\left(c_j^i \middle| W^{i-1}\right)}\right) + \left(\frac{N_{1+}\left(c_j^i \middle| W^{i+1}\right)}{\sum_{r=1}^{k_i} N_{1+}\left(c_j^i \middle| W^{i+1}\right)}\right)}{2}$$

Dimana,

$$(1-\lambda) = \frac{D}{\sum_{r=1}^{k_i} count\left(W^{i-1}c_r^i\, W^{i+1}\right) + 0.01 \times V_j} N_{1+}\left(c_j^i \middle| W^{i-1}, W^{i+1}\right)$$

$$N_{1+}\left(c_j^i \middle| W^{i-1}, W^{i+1}\right) = \left|\left\{W^i : count\left(W^{i-1}c_j^i\, W^{i+1}\right) > 0\right\}\right|$$

$N_{1+}\left(c_j^i \middle| W^{i-1}, W^{i+1}\right)$ merupakan nilai kata W^i dimana jumlah kemunculan kata dari trigram yang lebih dari 0

Nilai D (*Discount*) diasumsikan sebagai $0 \leq D \leq 1$ [3]. Nilai D yang digunakan adalah 0.01 dengan pertimbangan tidak terlalu mempengaruhi perhitungan skor pada tahap selanjutnya

$\frac{\left(\frac{N_{1+}\left(c_j^i \middle| W^{i-1}\right)}{\sum_{r=1}^{k_i} N_{1+}\left(c_j^i \middle| W^{i-1}\right)}\right) + \left(\frac{N_{1+}\left(c_j^i \middle| W^{i+1}\right)}{\sum_{r=1}^{k_i} N_{1+}\left(c_j^i \middle| W^{i+1}\right)}\right)}{2}$ merupakan nilai rata-rata dari probabilitas bigram kiri dan bigram kanan yang dihitung menggunakan jumlah kemunculan bigram kiri dan kanan anggota *confusion set* dari kata yang lebih dari 0 dibagi dengan total jumlah kemunculan bigram kiri dan kanan anggota *confusion set* semua token data uji yang lebih dari 0.

2.3.4 *Weighted Combination Score*

Perhitungan *score* untuk probabilitas setiap metode menggunakan persamaan

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

Dengan memberikan bobot kepada setiap perhitungan probabilitas bigram kiri, bigram kanan dan trigram. Nilai $\lambda_1\lambda_2\lambda_3$ diberikan dengan tujuan agar

$Score(c_j^i)$ terbatas pada $0 \leq Score(c_j^i) \leq 1$. Berdasarkan penelitian yang dilakukan oleh Samantha nilai terbaik untuk setiap bobot yang diberikan adalah $\lambda_1 = \lambda_2 = 0.25$ dan $\lambda_3 = 0.5$[6]

2.3.5 Deteksi Kesalahan dan Pemilihan Kata Sugesti

Untuk mengetahui suatu kata merupakan kesalahan *real-word*, diberikan beberapa syarat. Pertama, elemen-elemen *confusion set* harus terurut mulai dari skor terbesar sampai terkecil. Dalam penelitian Mays didapatkan nilai optimum yaitu 0.99 dengan kata lain dipercaya bahwa kata yang diuji dapat menjadi kesalahan kata *real-word* dalam 1% kasus. Maka untuk suatu kata dapat dikatakan sebuah kesalahan kata yaitu jika nilai skor kata yang diuji kurang dari 1% nilai skor terbesar elemen-elemen *confusion set* yang telah diurutkan.

2.3.6 Rencana Pengujian

Kinerja dari metode-metode smoothing yang digunakan dalam sistem ini diukur dengan melakukan pengukuran akurasi deteksi dan akurasi koreksi[1]. Pengukuran akurasi deteksi dilakukan dengan persamaan

$$\text{Akurasi deteksi} = \frac{ds}{dm} \times 100\%$$

Dimana,

Ds :jumlah deteksi yang dihasilkan oleh sistem

Dm :jumlah deteksi yang dihasilkan secara manual

Sedangkan pengukuran akurasi koreksi dilakukan dengan persamaan

$$\text{Akurasi koreksi} = \frac{kb}{kb+ks} \times 100\%$$

Dimana,

Kb : jumlah koreksi yang dihasilkan oleh sistem

Ks : jumlah koreksi yang dihasilkan oleh sistem

2.4 Pemodelan

Pemodelan data merupakan metode yang digunakan untuk menggambarkan data, hubungan data, batasan data yang ada di dalam sistem deteksi dan koreksi kesalahan kata yang akan dibangun[7]. Pada penelitian ini

diagram yang digunakan untuk memodelkan data adalah Flowchart, Diagram Konteks, dan DFD (*Data Flow Diagram*). Berikut penjelasannya

2.4.1 Flowchart

Flowchart merupakan representasi secara grafik dari suatu langkah-langkah atau prosedur untuk menyelesaikan suatu masalah . Dengan menggunakan *flowchart* akan memudahkan pengecekan bagian-bagian yang terlupakan dalam analisis masalah [8].

Flowchart biasanya memepermudah penyelesaian suatu masalah khususnya masalah yang perlu dipelajari dan dievaluasi lebih lanjut.

2.4.2 Diagram Konteks

Diagram Konteks merupakan diagram yang terdiri dari suatu proses dan menggambarkan ruang lingkup suatu sistem[9]. Diagram konteks berfungsi menggambarkan transformasi dari suatu proses yang melakukan transformasi data input dan menjadi transformasi data output[10]. Diagram konteks menggarisbawahi sejumlah karakteristik penting dari suatu sistem yaitu,

1. Kelompok pemakai, sistem atau pihak lain dimana sistem melakukan komunikasi yang disebut sebagai terminator.
2. Data dimana sistem menerima dari lingkungan dan diproses dengan cara tertentu dan data yang dihasilkan sistem diberikan kepada dunia luar.
3. Penyimpanan data yang digunakan secara bersamaan antara sistem dan terminator.
4. Batasan antara sistem dan lingkungan

2.4.3 DFD (*Data Flow Diagram*)

DFD (*Data Flow Diagram*) merupakan pemodelan yang menggambarkan sistem sebagai suatu jaringan proses fungsional yang dihubungkan satu sama lain dengan alur data, baik secara manual atau komputerisasi. DFD juga disebut sebagai bubble diagram, diagram alur kerja atau model fungsi[11]. Berikut merupakan komponen dari DFD

1. Komponen Terminator/ Entitas Luar

Terminator mewakili entitas luar yang berkomunikasi langsung dengan sistem. Terdapat dua jenis terminator yaitu terminator sumber dan terminator tujuan. Terminator dapat berupa orang, sekelompok orang, dapartemen di dalam organisasi atau sistem lain yang berkomunikasi dengan sistem yang sedang dikembangkan.

2. Proses

Merupakan bagian dari sistem yang menggambarkan transformasi *input* menjadi *output*. Proses diberi nama untuk menjelaskan kegiatan apa yang sedang atau akan dilaksanakan.

3. Data *Store*

Data *Store* berkaitan dengan penyimpanan, seperti *file* dan *database* yang berkaitan dengan penyimpanan secara komputerisasi. Suatu data *store* dihubungkan dengan alur data hanya pada komponen proses, tidak dengan komponen DFD lainnya.

4. Data *Flow*

Data *flow* digambarkan dengan anak panah yang berfungsi untuk menunjukan arah menuju dan keluar dari suatu proses. Data *flow* berguna untuk menjelaskan perpindahan informasi dari satu bagian sistem ke bagian lainnya.

2.5 Bahasa Pemrograman

Bahasa pemrograman atau yang biasa disebut bahasa komputer merupakan himpunan dari aturan sintaks dan semantik yang berfungsi untuk mendefinisikan program komputer. Bahasa pemrograman memerintah komputer untuk mengolah data sesuai dengan alur berpikir yang manusia perintahkan[12]. Beberapa bahasa pemrograman yang banyak digunakan antara lain *Java, JavaScript*, PHP, C, C++, *Cobol* dan *Python*. Pada pembangunan sistem deteksi dan koreksi kesalahan kata ini, bahasa pemrograman yang digunakan adalah PHP, dan *JavaScript*.

2.5.1 PHP

PHP atau (*Hypertext Preprocessor*) merupakan bahasa pemrograman yang berada pada sisi server (*script server-side)* yang didesain untuk pengembangan *website*. Disebut *script server-side* karena diproses pada komputer server[13].

Kelebihan menggunakan bahasa pemrograman PHP adalah sangat kompatibel dengan berbagai sistem operasi seperti Linux, Windows atau MacOs. PHP juga sangat kompatibel dengan HTML dan sangat efisien jika digabungkan dengan *javascript* dan bahasa pemrograman lainnya.

2.5.2 Javascript

Javascript merupakan bahasa pemrograman yang berbentuk *script* yang berfungsi untuk memberikan tampilan yang kelihatan lebih interaktif pada *website* yang dibangun. Bahasa pemrograman ini memberikan kemampuan tambahan kepada HTML(*Hypertext Markup Language*) dengan melakukan eksekusi perintah pada sisi *client (script client-side*)[14]. Meskipun secara umum digunakan pada *script client-side* namun dapat pula digunakan pada *script server-side* dengan memprogramnya sebagai bahasa *python* atau *perl*[15].

Javascript berfungsi untuk mendeteksi dan merespon *event-event* yang diberikan oleh pengguna. *Javascript* dapat digunakan untuk perhitungan aritmatika, manipulasi tanggal dan waktu, modifikasi *array* dan menangani *event* yang diinisiasi oleh pengguna serta menetapkan waktu.

2.6 Database

Database merupakan suatu komponen yang penting dalam sebuah sistem karena di dalam *database* tersimpan semua informasi yang akan diolah dan dihasilkan akan tersimpan di dalam *database*[16]. *Database* merupakan kumpulan *file-file* yang berhubungan secara logis dan digunakan secara rutin pada operasi-operasi sistem. Suatu *database* umumnya berisi elemen-elemen data yang disusun ke dalam file-file yang diorganisasikan berdasarkan sebuah skema atau struktur tertentu sehingga *database* menunjukkan suatu kumpulan

tabel yang dipakai dalam suatu perusahaan atau instansi untuk tujuan tertentu[17].

Dalam pengelolaan *database* menggunakan sebuah sistem manajemen *database* yaitu *database management system* (DBMS). *Software* yang digunakan untuk mengelola database pada pembangunan sistem deteksi dan koreksi kesalahan kata menggunakan MySQL dengan bahasa SQL. Berikut penjelasannya.

2.6.1 SQL (Structured Query Language)

Structured Query Language(SQL) adalah bahasa yang digunakan untuk mendesign manajemen data pada *relational database management systems* (RDMS). SQL adalah bahasa standar komputer yang pada awalnya dikembangkan oleh IBM untuk query mengubah dan mendefinisikan basis data relasional, menggunakan pernyataan deklaratif[18].

Terdapat 3 jenis perintah SQL[19], yaitu

1. *Data Definition Language* (DDL)

   DDL merupakan perintah SQL yang berhubungan dengan definisi dari suatu struktur database, yaitu database dan table. Perintah dasar yang termasuk DDL adalah CREATE, ALTER, RENAME, DROP.

2. *Data Manipulation Language* (DML)

   DML merupakan perintah SQL yang berhubungan dengan manipulasi atau pengolahan data atau *record* dalam table. Perintah yang termasuk di dalam DML adalah SELECT, INSERT, UPDATE, DELETE.

3. *Data Control Language* (DCL)

   DCL merupakan perintah SQL yang berhubungan dengan pengelolaan user dah hak akses terhadap setiap objek di MySQL. Perintah SQL yang termasuk di dalam DCL adalah GRANT, REVOKE.

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[20]. Pada MySQL dikenal dengan istilah *database* dan tabel. Tabel sendiri merupakan sebuah struktur data

dua dimensi yang terdiri dari baris-baris record dan kolom. 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 Sistem

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

2.7.1 Sublime Text

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.

2.7.2 Web *Browser*

Web *Browser* adalah layanan internet yang berfungsi menjelajahi dunia maya dengan menggunakan jaringan internet [21]. Fungsi dari Web *Browser* sendiri adalah untuk menampilkan halaman web atau melakukan interaksi dengan dokumen yang disediakan server.

Dalam penelitian ini penulis Google Chrome versi 85.0.4183.83 sebagai Web *Browser* untuk pembangunan dan pengujian sistem deteksi dan koreksi kesalahan ejaan *real-word* yang dibangun.

2.7.3 XAMPP

XAMPP adalah aplikasi web server yang berdiri sendiri terdiri dari Apache HTTP Server, MySQL *database* dan PHP. XAMPP dilengkapi dengan manajemen *database* PHPMyAdmin [20]. 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 berupa PHP, MySQL dan Perl. Dalam penelitian ini XAMPP digunakan sebagai server localhost untuk membuka sistem deteksi dan koreksi kesalahan ejaan *real-word* secara *offline*.