# BAB 2
# LANDASAN TEORI

## 2.1 Tinjauan Teori

Pada bagian ini dijabarkan teori-teori yang mendasari penelitian ini secara lengkap.

### 2.1.1 Portal Berita Daring

Portal berita daring adalah suatu produk dari perkembangan teknologi internet yang menyajikan berita dalam bentuk *website*, aplikasi, dan media sosial [2]. Portal berita daring seringkali dijadikan sebagai referensi terhadap peristiwa-peristiwa terbaru yang terjadi di masyarakat [16]. Informasi-informasi yang telah dikumpulkan tersebut kemudian diberitakan pada platform media sosialnya masing-masing secara luas. Hal ini menunjukkan bahwa adanya kerjasama yang terjadi antara pihak media dan masyarakat dalam memberikan informasi yang faktual dan aktual melalui portal berita daring.

### 2.1.2 Tribunnews

Tribunnews adalah salah satu portal berita daring yang dimiliki Indonesia dan dikelola oleh PT. Tribun Digital *Online* yang termasuk ke dalam grup Kompas Gramedia. Tribunnews memiliki kantor yang berpusat di Jakarta dan menyajikan berita-berita nasional, regional, internasional, olahraga, ekonomi dan bisnis, serta selebritas dan *lifestyle* [17]. Menurut Helen dan Gafar [18], Tribunnews menarik perhatian pembacanya dengan menggunakan teknik memancing (*clickbait*) dan termasuk sebagai salah satu portal berita yang memasuki peringkat lima teratas berdasarkan Alexa.com pada tahun 2019.

### 2.1.3 Natural Language Processing

*Natural Language Processing* (NLP) adalah cabang dari ilmu kecerdasan buatan (*Artificial Intelligence*) dalam melatih komputer untuk dapat memahami, memproses, dan menghasilkan bahasa. Teknologi-teknologi terkait NLP seringkali dijumpai dalam kehidupan sehari-hari tanpa disadari seperti, *search engines*

contohnya Google, *machine translation services* contohnya Google Translate, dan *voice assistants* contohnya Siri, Alexa, dan Google Home. Penerapan NLP dalam kehidupan sehari-hari dapat membantu masyarakat dalam mengembangkan proses bahasa yang efektif [19]. Selain untuk mengembangkan proses bahasa yang efektif, penerapan NLP ini juga menjadi salah satu upaya dalam melestarikan bahasa nasional yaitu, bahasa Indonesia.

### 2.1.4 Text Preprocessing

*Text preprocessing* adalah tahapan awal yang dilakukan untuk menyeleksi data-data yang nantinya diproses supaya lebih terstruktur. *Text Preprocessing* adalah langkah yang penting dalam melakukan *Text Mining*, *Natural Language Processing* (NLP), dan *Information Retrieval* (IR). Menurut Latius dan Maria [20], ada 4 tahap dalam proses *text preprocessing* sebagai berikut:

1. *Case folding*

   *Case folding* adalah tahap dimana semua huruf yang ada dalam sebuah dokumen teks diubah menjadi huruf kecil, misalnya kata ”Pada” diubah menjadi ”pada” [21].

2. *Tokenizing*

   *Tokenizing* adalah tahap untuk memecah dokumen teks yang masih berbentuk kalimat menjadi kata-kata yang disebut token [21].

3. *Filtering*

   *Filtering* atau yang lebih dikenal dengan *Stop-Word Removal* adalah tahap dimana kata-kata yang tidak penting dihapus seperti pakaian, kata depan, dan kata benda [21].

4. *Stemming*

   *Stemming* adalah proses penguraian suatu kata menjadi bentuk kata dasarnya dengan menghapus imbuhan-imbuhan yang ada pada suatu kata [21].

Selain itu, menurut Syifa et al. [22], penggunaan teknik *text preprocessing* dapat meningkatkan persentase akurasi sistem yang dikembangkan.

### 2.1.5 Sastrawi

Sastrawi adalah salah satu *library* yang digunakan untuk melakukan proses *stemming* yang bergantung dengan kamus kata dasar pada kateglo.com. Berikut adalah aturan-aturan dari *stemmer* sastrawi [23].

1. Pertama adalah mengecek apakah kata yang di-*stem* ada dalam kamus kata dasar atau tidak. Jika kata yang di-*stem* ada dalam kamus kata dasar, maka proses berhenti pada langkah ini.
2. Jika kata tersebut tidak ada dalam kamus, artinya kata tersebut merupakan kata berafiks. Jika kata tersebut adalah kata berafiks, hilangkan akhirannya yang terdiri dari -lah, -kah, -ku, -mu, -nya, -lah, -kah, -tah atau -pun.
3. Menghilangkan imbuhan turunan seperti -i, -kan, dan -an, kemudian menghapus be-, di-, ke-, me, pe-, se- dan te-.
4. Jika kata dasar yang dihasilkan dari langkah-langkah sebelumnya tidak ditemukan dalam kamus, maka dilakukan pengecekan apakah kata tersebut termasuk dalam tabel ambigu pada kolom terakhir atau tidak.
5. Terakhir, ketika semua langkah di atas gagal, algoritma mengembalikan kata ke kata aslinya.

Dyah, et. al [23] juga melakukan penelitian untuk membandingkan efektivitas algoritma *stemming* pada dokumen berbahasa Indonesia. Dalam penelitian tersebut, kesimpulan yang didapatkan adalah dari tiga algoritma yang dibandingkan, sastrawi memiliki hasil tertinggi dengan akurasi *stemming* sebesar 95,2%.

### 2.1.6 Algoritma Rabin-Karp

Algoritma Rabin-Karp adalah algoritma yang ditemukan oleh Michael O. Rabin dan Richard M. Karp. Algoritma ini menggunakan metode *hashing* dalam mencari suatu kata. Algoritma ini lebih efektif jika digunakan untuk pencarian kata jamak (komparasi) dibandingkan dengan pencarian kata tunggal [24].

### A Hashing

*Hashing* adalah suatu cara untuk mengubah *string* menjadi suatu nilai unik dengan panjang tertentu yang berfungsi sebagai penanda string tersebut. Perubahan jenis data tersebut bisa dari data dengan jenis alfabet menjadi jenis ASCII (*American Standard Code for Information Interchange*). Salah satu metode *hasing* yang ada adalah metode *rolling hash*. Metode *rolling hash* digunakan untuk mencari pola pada teks supaya pencariannya lebih akurat dan nilai *hash*-nya dapat diperbarui dengan rumus tertentu berdasarkan karakter yang dibuang dan ditambahkan. Salah satu varian dari metode *rolling hash* adalah *polynomial rolling hash*. Metode *Rolling hash* melakukan perhitungan dengan mengambil *substring* dalam sebuah *string* sepanjang k karakter yang kemudian nilai *hash*-nya dihitung berdasarkan nilai ASCII masing-masing karakter [25] dengan rumus perhitungan sebagai berikut [26].

$$H(c_1...c_k) = (c_1 * p^{(k-1)} + c_2 * p^{(k-2)} + ... + c_{k-1} * p^k + c_k) \; mod \; m \qquad (2.1)$$

dimana :
H = *substring*
c = nilai ASCII per-karakter
p = konstan bilangan prima
k = banyak karakter
m = modulo bilangan prima

### 2.1.7 Fuzzy-Wuzzy

Pencocokan *string* menggunakan fuzzy sering digunakan untuk menemukan pola antara dua buah *string*. Fuzzy *string matching* ini dapat digunakan untuk melakukan pemeriksaan ejaan kata, penggunaan kata yang berulang, *spam*, dan pencocokan urutan DNA dalam ranah bioinformatika. Fuzzy-Wuzzy adalah salah satu *library* yang dapat digunakan pada bahasa pemrograman python untuk melakukan pencocokan bahasa satu dengan yang lain. Fuzzy-Wuzzy melakukan pencocokan berbasis jarak menggunakan algoritma Levenshtein Distance [27]. Berikut adalah *pseudocode* dari algoritma Levenshtein Distance yang ditunjukkan pada kode 2.1 [28].

```
int LevenshteinDistance (char s[1..m], char t[1..n])
```

```
{
declare int d[0..m, 0..n]
declare int cost
    for i from 0 to m d[i,0] :=i
    for j from 0 to n d[0,j] :=j
        for j from 1 to n{
            for i from 1 to m {
                if s[i]!=t[j] then cost := 1
                else cost := 0
                d[i,j] := minimum(
                    d[i-1,j]+1,
                    d[i, j-1]+1,
                    d[i-1,j-1]+cost
                )
            }
        }
return d[m,n]
}
```

Kode 2.1: Pseudocode Algoritma Levenshtein Distance

### 2.1.8 Confusion Matrix

*Confusion matrix* adalah tabel yang dapat menunjukkan performa dari sebuah algoritma klasifikasi. Tabel 2.1 adalah tabel *confusion matrix* yang digunakan pada penelitian ini.

| | **Predicted Value** | |
|---|---|---|
| **Actual Value** | 1 | 0 |
| 1 | TP | FN |
| 0 | FP | TN |

Tabel 2.1. Tabel *Confusion Matrix*

Tabel *confusion matrix* memiliki empat karakteristik dasar yang dapat menentukan metrik pengukuran klasifikasinya. Berikut adalah empat karakteristik *confusion matrix* berdasarkan kasus diagnosis penyakit menggunakan *machine learning* [29].

- TP (*True Positive*).

  TP berarti kata diprediksi salah dan memang benar kata tersebut dideteksi salah.

- TN (*True Negative*).

  TN berarti kata diprediksi benar dan memang benar tidak ada kesalahan ejaan kata luluh.

- FP (*False Positive*).

  FP berarti kata diprediksi salah tapi ternyata kata tidak mengalami kesalahan ejaan kata luluh. FP juga dikenal dengan nama *Type 1 error*.

- FN (*False Negative*).

  FN berarti kata diprediksi benar tapi ternyata kata tersebut mengalami kesalahan ejaan kata luluh. FN juga dikenal dengan nama *Type II error*.

Performa dari algoritma yang digunakan dapat dilihat berdasarkan hasil perhitungan *accuracy*, *precission*, *recall*, dan *F1 score* [29].

- *Accuracy*.

  Akurasi dari sebuah algoritma direpresentasikan sebagai rasio prediksi kata yang benar (TP+TN) dengan total katanya (TP+TN+FP+FN).

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \tag{2.2}$$

- *Precision*.

  Presisi dari sebuah algoritma direpresentasikan sebagai rasio kata yang diprediksi mengalami kesalahan ejaan dan memang mengalami kesalahan ejaan (TP) dengan total prediksi kata yang mengalami kesalahan ejaan (TP+FP).

$$Precision = \frac{TP}{TP+FP} \tag{2.3}$$

- *Recall*.

  *Recall* didefinisikan sebagai rasio kata yang diprediksi mengalami kesalahan dan memang mengalami kesalahan (TP) dibagi dengan jumlah kata yang benar mengalami kesalahan tersebut (TP+FN).

$$Recall = \frac{TP}{TP+FN} \tag{2.4}$$

- *F1 score*.

  *F1 score* atau yang lebih dikenal dengan *F Measure* adalah keadaan dimana *precision* dan *recall* berada pada kondisi seimbang.

$$F1Score = \frac{2 * precision * recall}{precision + recall} \tag{2.5}$$