# BAB II

# KAJIAN TEORITIK, KERANGKA BERFIKIR, DAN HIPOTESIS PENELITIAN

## 2.1. Kajian Teoretik

### 2.1.1. Plagiarisme

Parsons (2007) mendefinisikan plagiarisme sebagai 'pemakaian tidak sah' atau peniruan bahasa dan pemikiran pengarang lain, pemikiran tersebut diambil tanpa mengutip sumber aslinya dan dianggap sebagai karya asli seseorang. Selain itu plagiarisme juga merupakan perbuatan sengaja dan tidak sengaja dalam memperoleh kredit atau nilai untuk suatu karya ilmiah, dengan mengutip sebagian atau seluruh karya ilmiah orang lain yang diakui sebagai karya ilmiahnya tanpa menyatakan sumber secara cepat dan memadai (Peraturan Menteri Pendidikan RI Nomor 17 tahun 2010).

Dari beberapa definisi diatas maka dapat disimpulkan plagiarisme terbagi menjadi beberapa tipe yaitu (Soelistyo, 2011) :

1. Plagiarisme Kata demi Kata (*Word for word Plagiarism*)

   Penulis menggunakan kata-kata penulis lain dengan mengganti struktur kalimat dalam paragraf atau mengganti kata dalam kalimat tanpa menyebutkan sumbernya. Plagiarisme ini biasanya terjadi dalam pembuatan makalah, tugas, dan karya tulis mahasiswa.

2. Plagiarisme atas Sumber (*Plagiarism Source*)

Penulis menggunakan gagasan orang lain tanpa memberikan pengakuan yang cukup tanpa menyebutkan sumbernya secara jelas.

3. Plagiarisme Kepengarangan (*Plagiarism of authorship*)

   Penulis mengaku sebagai pengarang karya tulis milik orang lain.

4. *Self Plagiarisme*

   Tipe ini termasuk kedalam tipe Penulis mempublikasikan satu artikel pada lebih dari satu redaksi publikasi. Dan mendaur ulang karya tulis atau karya ilmiah. Yang terpenting dalam *Self Plagiarisme*adalah bahwa mengambil karya sendiri, maka ciptaan karya baru yang dihasilkan harus memiliki perubahan yang berarti. Artinya karya lama merupakan bagian kecil dari karya baru yang dihasilkan. Sehingga pembaca akan memperoleh hal baru yang benar-benar penulis sajikan pada karya tulis yang menggunakan karya lama.

Dari jenis-jenis plagiarisme diatas dapat disimpulkan bahwa mengapa plagiarisme terjadi karena beberapa tindakan plagiarisme sering terjadi di sekitar akademis. Tentu saja hal ini cukup menjadi perhatian akademis, sehingga menjadi sangat penting bagi seorang akademis untuk menghindari atau mengantisipasi tindakan tersebut. Plagiarisme dapat merugikan banyak orang terutama sang pengarang atau sumber. Plagiarisme akan mencoreng karir seorang akademis dan tidak berlebihan jika plagiarisme disebut sebagai kejahatan intelektual.

Berikut ini adalah beberapa alasan atau faktor pemicu tindakan plagiarisme bisa terjadi yaitu (Sulianta, 2010):

1. Kurangnya kesadaran beretika sebagai *citizen* atau *netizen*.
2. Fasilitas *internet*.
3. Perangkat teknologi informasi dengan mobilitas tinggi seperti laptop atau *handphone* yang memberikan kemudahan dalam mengakses *internet*.
4. Mendapatkan prestasi dengan berbagai upaya.
5. Tersedianya *search engine* yang semakin mampu untuk menemukan sumber-sumber *internet content* yang dimaksud.

Dari faktor-faktor di atas, sebagai penulis sebenarnya dapat menghindari tindakan plagiarisme tersebut dengan beberapa cara yaitu (Sudigo, 2007):

1. Bila menggunakan ide lain sebutkan sumbernya.
2. Melakukan parafrase yang dimaksud dari parafrase tersebut adalah mengungkapkan ide atau gagasan orang lain di karya ilmiah kita dengan tetap menyebutkan sumbernya.
3. Bila menggunakan kata atau kalimat orang lain sebutkan sumbernya, dengan catatan:
    a. Gunakan tanda kutip bila kata atau kalimat aslinya disalin secara utuh
    b. Tanda kutip tidak diperlukan bila kata atau kalimat telah diubah menjadi kalimat penulis sendiri tanpa mengubah artinya (telah dilakukan parafrase).
    c. Mengubah satu atau beberapa kata dalam satu paragraf bukan merupakan parafrase maka dari itu tanda kutip perlu disertakan.

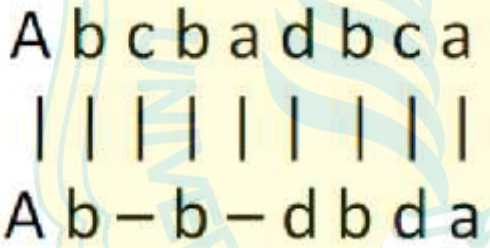
Fenomena plagiarisme didalam dunia perkuliahan sering terjadi. Plagiarisme yang dimaksud adalah yang bersifat sengaja maupun tidak disengaja, dalam suatu kasus perkuliahan sering sekali terjadi plagiarisme pada pengerjaan tugas atau karya tulis mahasiswa hal tersebut sudah menjamur dikalangan mahasiswa. Ketika mahasiswa melakukan plagiarisme pada tugas atau karya tulis mereka pada dasarnya dosen dapat mengidentifikasi hasil dari tugas atau karya tulis mahasiswanya apakah terdapat plagiarisme atau tidak.

Di Teknik Elektro Universitas Negeri Jakarta dosen masih menggunakan cara manual untuk mengidentifikasi plagiarisme dengan cara pemeriksaan secara manual. Dengan membandingkan dokumen atau tugas mahasiswanya satu-persatu, cara tersebut akan menghasilkan tingkat akurasi yang sangat tinggi. Namun cara tersebut tidak efisien terutama jika dokumen yang dibandingkan berjumlah banyak karena akan memerlukan waktu yang cukup banyak tergantung dari jumlah dokumen yang akan dibandingkan.

Didalam penelitian ini penulis akan mencoba memberikan solusi untuk mempersingkat waktu pemeriksaan plagiarisme dari setiap dokumen yang dibandingkan dengan menerapkan Algoritma Smith-Waterman pada aplikasi pendeteksi dugaan awal plagiarisme pada tugas mahasiswa yang bertujuan untuk membantu proses identifikasi plagiarisme, dengan memberikan informasi dugaan persentase kemiripan untuk setiap dokumen yang dibandingkan.

### 2.1.2. Algoritma Smith-Waterman

Algoritma Smith-Waterman merupakan algoritma klasik yang telah dikenal luas dalam bidang bioinformatika sebagai metode yang mengidentifikasi *local similarities* (penyejajaran sekuens) yaitu penyusunan dua *local sequences* (rangkaian atau susunan) *nucleotide* atau *protein squences* sehingga kemiripan antara dua *sequence* tersebut akan terlihat. Berdasarkan fungsi proses penyejajaran sekuens tersebut, maka algoritma ini dapat dikonversikan ke dalam pemrograman komputer untuk digunakan membantu proses pendeteksian dokumen teks yang dianggap cenderung plagiat dengan cara melihat kesamaan isi (*local similarities*) dari beberapa dokumen teks. (Farid Bangkit Djafar, Agus Lahinta, & Lillyan Hadjaratie, 2014)




**Gambar 2.1 Optimal Alignment Dari Dua *Substring***

Algoritma standar Smith-Waterman yang dipakai untuk perhitungan *local alignment*berdasarkan Jurnal Farid Bangkit Djafar, Agus Lahinta, & Lillyan Hadjaratie(2014) adalah:

1. Menambahkan sebuah nilai pada setiap perbandingan
    a) Menggunakan nilai positif, apabila memiliki kemiripan
    b) Menggunakan nilai negatif, apabila memiliki perbedaan
2. Inisialisasi awal Matriks dengan nilai 0 (nol)

3. Semua nilai yang terdapat dalam matriks apabila lebih kecil dari 0 (nol), maka nilai dianggap 0 (nol).
4. Memulai Traceback dari nilai yang tertinggi yang ditemukan dimanapun pada matriks.
5. Penghitungan dilanjutkan hingga skor bernilai 0 (nol).

Berikut adalah cara kerja dari Algoritma Smith-Waterman dalam menemukan kesamaan dari dua DNA :

DNA Sequence 1 = AGCCG *Match* = +2

DNA Sequence 2 = ATGCC *Mismatch* = -1

*Gap* = -1

Langkah awal dari pengerjaan Algoritma Smith-Waterman adalah mendefinisikan nilai 0 pada matriks di titik (0,0), (i,0), dan (0,j) kemudian dilakukan perhitungan pada titik (1,1). Apabila DNA *i* sama dengan DNA *j*, maka akan digunakan nilai *Match*. Sebaliknya, apabila DNA dari kedua *sequence* berbeda, maka akan digunakan nilai *mismatch*.

Berikut ini adalah rumus dari Algoritma Smith-Waterman :

$$H(1,1) = \text{maks.} \begin{cases} 0 \\ H(i-1,j-1) + \text{Nilai } \textit{Match/Mismatch} \\ H(i-1,j) + \text{Nilai } \textit{Gap} \\ H(i,j-1) + \text{Nilai } \textit{Gap} \end{cases}$$

$$H(1,1) = \text{maks.} \begin{cases} 0 \\ 0 + 2 = 2 \\ 0 + -1 = -1 \\ 0 + -1 = -1 \end{cases}$$

Rumus (2.1)

Dapat dilihat bahwa DNA dari *i* dan dari *j* memiliki kesamaan, maka akan menggunakan nilai *Match*. Nilai yang paling besar adalah 2, maka nilai pada titik (1,1) adalah 2. Perhitungan selanjutnya ditentukan dengan cara yang sama seperti pada titik (1,1)

| | - | A | G | C | C | G |
|---|---|---|---|---|---|---|
| - | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | 2 | 1 | 0 | 0 | 0 |
| T | 0 | 1 | 1 | 0 | 0 | 0 |
| G | 0 | 0 | 3 | 2 | 1 | 2 |
| C | 0 | 0 | 2 | 5 | 4 | 3 |
| C | 0 | 0 | 1 | 4 | 7 | 6 |

**Gambar 2.2 Nilai Matriks dari dua pasangan DNA**

Setelah dilakukan perhitungan dalam matriks, kemudian dilakukan *traceback* dari nilai tertinggi matriks, yaitu dari titik (5,4). Dari hasil *traceback* yang dilakukan pada matriks adalah sebagai berikut.

| | - | A | G | C | C | G |
|---|---|---|---|---|---|---|
| - | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | 2 | 1 | 0 | 0 | 0 |
| T | 0 | 1 | 1 | 0 | 0 | 0 |
| G | 0 | 0 | 3 | 2 | 1 | 2 |
| C | 0 | 0 | 2 | 5 | 4 | 3 |
| C | 0 | 0 | 1 | 4 | 7 | 6 |

**Gambar 2.3*traceback* pada matriks**

Dari *traceback* tersebut bisa ditentukan *local similarities.Local similarities* dari kedua DNA tersebut adalah :

Sequence 1 = A- GCC

Sequence 2 = ATGCC

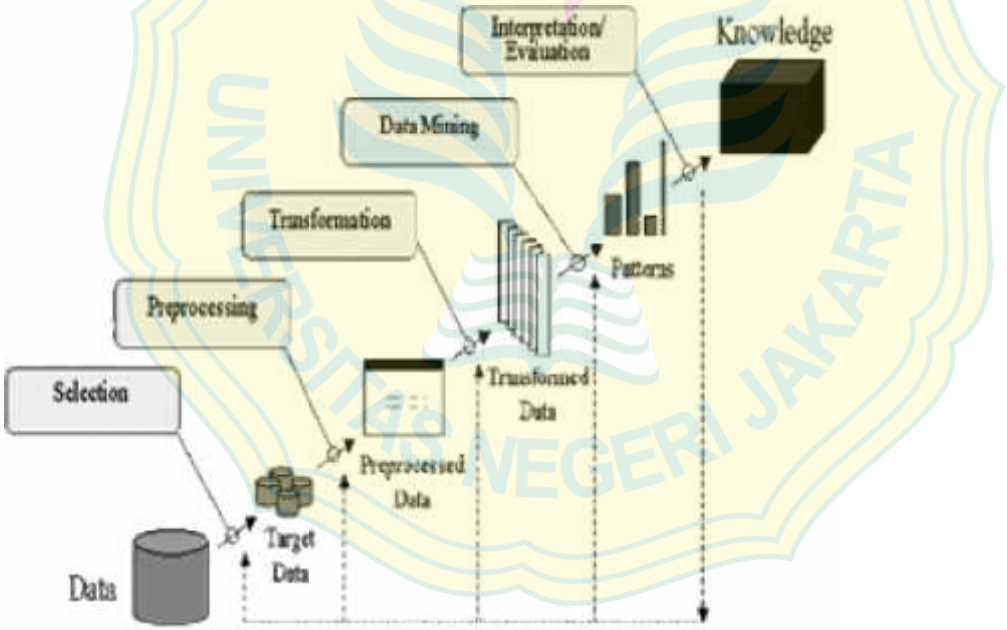
**Gambar 2.4*local similarities* dari dua sequence DNA**

Setelah mendapatkan *local similarities* dari dua pasangan sequence DNA kemudian dilakukan perhitungan bobot, kita bisa menghitung bobot kesamaan. Dari satu *alignment* terdapat 4 karakter yang sama dari total karakter masing-masing

dokumen adalah 5 karakter, maka hasilnya adalah (((4/5) + (4/5)) x 100) / 2 = 80% kemiripan dari kedua DNA tersebut.

### 2.1.3. Text Mining

*Text Mining* bertujuan untuk mendapatkan informasi yang berguna dari sekumpulan dokumen. Jadi, sumber data yang digunakan pada *text mining* adalah kumpulan teks yang memiliki format yang tidak tersetruktur atau minimal semi terstruktur. Adapun tugas khusus dari text mining antara lain yaitu kategorisasi teks (*text catagorization*) dan pengelompokan teks *(text clustering*) (Eldira, 2010).




**Gambar 2.5 Proses *Text mining***

*Text mining* memiliki beberapa tahapan seperti gambar di atas, salah satu tahapan dari*text mining* antara lain adalah *pre-processing.* Tahap *pre-processing*merupakan

tahapan awal dalam mengolah data yang akan diproses pada setiap dokumen. *Pre-processing*dilakukan untuk tujuan penyeragaman dan kemudahan pembacaan. Proses *pre-processing* meliputi tahap *read file, tokenizing*, *case folding*, *filtering*, *stemming*, dan *sorting*.Setelah tahap *Pre-processing* dilanjutkan dengan tahap *processing.* Tahap *processing* bisa disebut juga sebagai *analyzing* dimana semua data yang akan diolah menggunakan algoritma yang diterapkan dan akan mendapatkan hasil berupa bobot atau nilai dari setiap dokumen yang akan diseleksi.

### 2.1.4. Ekstraksi Dokumen

Proses ekstraksi dokumen merupakan proses pemisahan *string* dalam setiap dokumen, ekstraksi dokumen sendiri bertujuan untuk memudahkan dalam mencari informasi dalam sebuah dokumen. Dalam proses ekstraksi dilakukan *pre-processing*pada setiap *string* dokumen. Diantaranya *read file, tokenizing, case folding, sorting*. Berikut adalah susunan pre-processing pada dokumen.

1. *Read file*

   *Read file* adalah tahap awal pada *pre-processing* yang bertujuan untuk membaca teks didalam file atau dokumen.

Andi dan Paul bermain basket di halaman rumah

**Gambar 2.6*Read file* dokumen 1**

Kemarin Andi dan Paul tidak bermain basket di lapangan

**Gambar 2.7*Read file* dokumen 2**

2. *Tokenizing*

*Tokenizing* adalah proses pemisahan kalimat yang terdapat dalam sebuah file menjadi kata dengan cara *parsing* berdasarkan spasi agar terbentuk kumpulan kata seperti gambar dibawah. Contoh hasil dokumen teks setelah dilakukan *tokenizing*

| |
|---|
| Andi |
| dan |
| Paul |
| bermain |
| basket |
| di |
| halaman |
| rumah |

**Gambar 2.8*Tokenizing* dokumen 1**

| |
|---|
| Kemarin |
| Andi |
| dan |
| Paul |
| tidak |
| bermain |
| basket |
| di |
| lapangan |

**Gambar 2.9*Tokenizing* dokumen 2**

3. *Case folding*

Dalam tahap ini dilakukan proses pengubahan karakter atau huruf dalam sebuah dokumen teks menjadi huruf kecil dari a-z. Berikut adalah contoh dari hasil dokumen yang sudah dilakukan *case folding.*

andi dan paul bermain basket di halaman rumah

**Gambar 2.10*Case folding* dokumen 1**

kemarin andi dan paul tidak bermain basket di lapangan

**Gambar 2.11*Case folding* dokumen 2**

4. *Sorting*

Selanjutnya akan menggunakan tahap *sorting* yaitu proses pengurutan data yang sebelumnya disusun secara acak atau tidak teratur menjadi urut dan teratur (Sofyansyah, 2014).Pada dasarnya ada dua macam urutan yang biasa digunakan dalam suatu proses *sorting* namun yang digunakan pada aplikasi ini adalah secara *ascending*. Berikut ini adalah contoh hasil dari tahap *sorting*.

andi basket bermain dan di halaman paul rumah

**Gambar 2.12*Sorting* dokumen 1**

Andi basket bermain dan di kemarin lapangan tidak paul

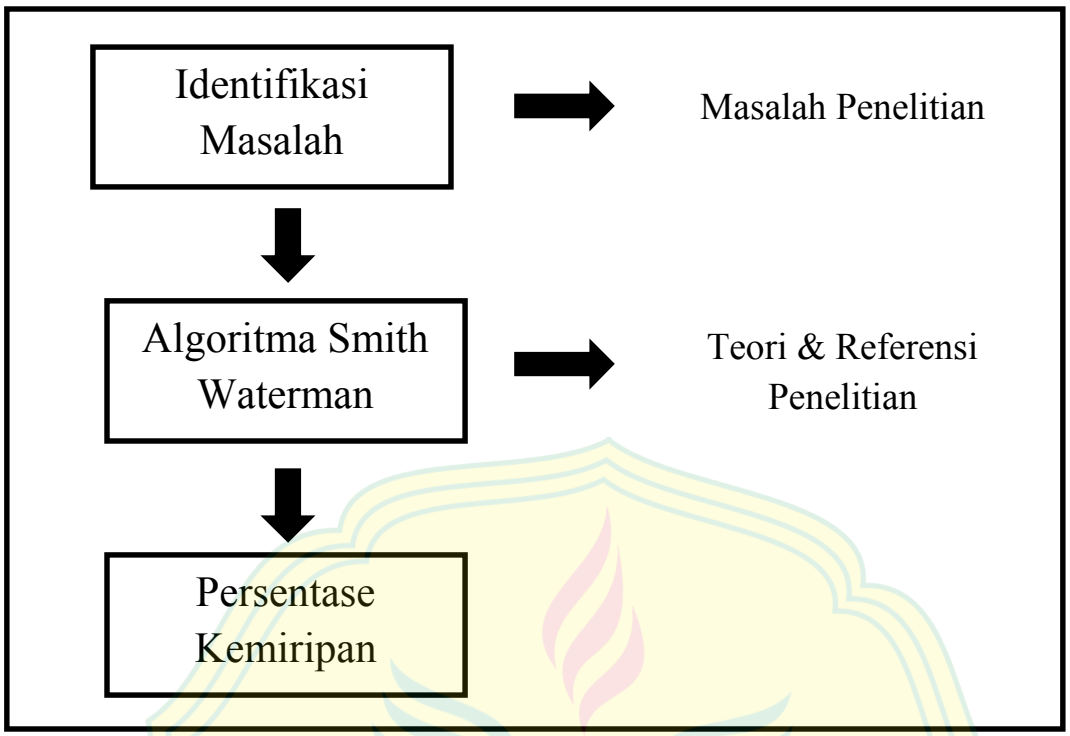
**Gambar 2.13*Sorting* dokumen 2**

### 2.1.5. Threshold

*Threshold* adalah nilai batas dari suatu parameter berupa *output* nilai minimum. Dalam penerapan aplikasi ini *threshold* berfungsi untuk menentukan batas nilai minimum perbandingan *token* kalimat antar dokumen melalui proses *analyzing* yang kemudian akan dilanjutkan dengan tahap berikutnya, yaitu perhitungan nilai kemiripan dokumen dengan menggunakan algoritma Smith-Waterman. dalam penelitian ini menetapkan *threshold* pada nilai 60. Dan selanjutnya nilai *threshold* itu akan ditentukan oleh *user*.

## 2.2. Kerangka Berpikir

Pendeteksian plagiarisme secara manual di jurusan Teknik Elektro Universitas Negeri Jakartatidak efisien karena membutuhkan waktu yang sangat lama apabila dokumen yang dibandingkan banyak maka dari itu dibutuhkan aplikasi untuk membantu dosen dalam mengidentifikasi dugaan awal plagiarisme. Aplikasi ini sendiri dikembangkan dengan menggunakan algoritma Smith-Waterman untuk membandingkan isi teks dokumen yang kemudian akan menghasilkan bobot atau nilai persentase kemiripan dalam isi teks dokumen namun sebelum dokumen-dokumen itu dibandingan akan dilakukan *pre-proscessing* terlebih dahulu kemudian dilakukan proses perhitungan isi teks dokumen dengan menggunakan algoritma Smith-Waterman untuk mendapatkan bobot atau nilai persentase kemiripan dokumen-dokumen tersebut. Bagan kerangka berfikir ini diilustrasikan pada gambar 2.14




**Gambar 2.14 Bagan Kerangka Berpikir**

### 2.3. Hipotesis Penelitian

Berdasarkan teori-teori yang telah dijabarkan diatas dapat diambil sebuah hipotesis bahwa Algoritma Smith-Waterman dapat diimplementasi pada aplikasi yang dapat menghitung kemiripan isi teks dokumen tugas mahasiswa sehingga dapat membantu dalam mendeteksi dugaan awal plagiarisme, terutama pada plagiarisme kata demi kata.