PERBANDINGAN ALGORITMA *LEVENSHTEIN* DAN *RATCLIFF/OBERSHELP* UNTUK MENDETEKSI PLAGIARISME

TUGAS AKHIR

Muhamad Dicky Pradana Tarigan
41518110148

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS MERCU BUANA
JAKARTA
2022

PERBANDINGAN ALGORITMA *LEVENSHTEIN* DAN *RATCLIFF/OBERSHELP* UNTUK MENDETEKSI PLAGIARISME

Tugas Akhir

Diajukan Untuk Melengkapi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer

Oleh:
Muhamad Dicky Pradana Tarigan
41518110148

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS MERCU BUANA
JAKARTA
2022

LEMBAR PERNYATAAN ORISINALITAS

Yang bertanda tangan dibawah ini:

NIM	: 41518110148
Nama	: Muhamad Dicky Pradana Tarigan
Judul Tugas Akhir	: PERBANDINGAN ALGORITMA *LEVENSHTEIN* DAN *RATCLIFF OBERSHELP* UNTUK MENDETEKSI PLAGIARISME

Menyatakan bahwa Laporan Tugas Akhir saya adalah hasil karya sendiri dan bukan plagiat. Apabila ternyata ditemukan didalam laporan Tugas Akhir saya terdapat unsur plagiat, maka saya siap untuk mendapatkan sanksi akademik yang terkait dengan hal tersebut.

Jakarta, 22 Juni 2022

Muhamad Dicky Pradana Tarigan

SURAT PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR

Sebagai mahasiswa Universitas Mercu Buana, saya yang bertanda tangan di bawah ini :

NIM	:	41518110148
Nama	:	Muhamad Dicky Pradana Tarigan
Judul Tugas Akhir	:	PERBANDINGAN ALGORITMA *LEVENSHTEIN* DAN *RATCLIFF OBERSHELP* UNTUK MENDETEKSI PLAGIARISME

Dengan ini memberikan izin dan menyetujui untuk memberikan kepada Universitas Mercu Buana **Hak Bebas Royalti Noneksklusif** (*None-exclusive Royalty Free Right*) atas karya ilmiah saya yang berjudul diatas beserta perangkat yang ada (jika diperlukan).

Dengan Hak Bebas Royalti/Noneksklusif ini Universitas Mercu Buana berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat dan mempublikasikan tugas akhir saya.

Selain itu, demi pengembangan ilmu pengetahuan di lingkungan Universitas Mercu Buana, saya memberikan izin kepada Peneliti di Lab Riset Fakultas Ilmu Komputer, Universitas Mercu Buana untuk menggunakan dan mengembangkan hasil riset yang ada dalam tugas akhir untuk kepentingan riset dan publikasi selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Jakarta, 22 Juni 2022

Muhamad Dicky Pradana Tarigan

SURAT PERNYATAAN LUARAN TUGAS AKHIR

Sebagai mahasiswa Universitas Mercu Buana, saya yang bertanda tangan di bawah ini :

Nama Mahasiswa	:	Muhamad Dicky Pradana Tarigan
NIM	:	41518110148
Judul Tugas Akhir	:	PERBANDINGAN ALGORITMA *LEVENSHTEIN* DAN *RATCLIFF/OBERSHELP* UNTUK MENDETEKSI PLAGIARISME

Menyatakan bahwa :

1. Luaran Tugas Akhir saya adalah sebagai berikut :

No	Luaran	Jenis			Status	
1	Publikasi Ilmiah	Jurnal Nasional Tidak Terakreditasi			Diajukan	√
		Jurnal Nasional Terakreditasi	√			
		Jurnal International Tidak Bereputasi			Diterima	
		Jurnal International Bereputasi				
	Disubmit/dipublikasikan di :	Nama Jurnal	: JURNAL RESTI (Rekayasa Sistem dan Teknologi Informasi)			
		ISSN	: 2580-0760			
		Link Jurnal	: https://jurnal.iaii.or.id/index.php/RESTI			
		Link File Jurnal Jika Sudah di Publish	:			

2. Bersedia untuk menyelesaikan seluruh proses publikasi artikel mulai dari submit, revisi artikel sampai dengan dinyatakan dapat diterbitkan pada jurnal yang dituju.
3. Diminta untuk melampirkan scan KTP dan Surat Pernyataan (Lihat Lampiran Dokumen HKI), untuk kepentingan pendaftaran HKI apabila diperlukan

Demikian pernyataan ini saya buat dengan sebenarnya.

Mengetahui
Dosen Pembimbing TA

Yaya Sudarya Triana, M.Kom., Ph.D.

Jakarta, 21 Juni 2022

Muhamad Dicky Pradana Tarigan

LEMBAR PERSETUJUAN PENGUJI

NIM	:	41518110148
Nama	:	Muhamad Dicky Pradana Tarigan
Judul Tugas Akhir	:	PERBANDINGAN ALGORITMA *LEVENSHTEIN* DAN *RATCLIFF/OBERSHELP* UNTUK MENDETEKSI PLAGIARISME

Tugas Akhir ini telah diperiksa dan disidangkan sebagai salah satu persyaratan untuk memperoleh gelar Sarjana pada Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Mercu Buana.

Jakarta, 07 – Juli – 2022

(Muhammad Rifqi S.Kom.,M.Kom)

LEMBAR PERSETUJUAN PENGUJI

NIM	:	41518110148
Nama	:	Muhamad Dicky Pradana Tarigan
Judul Tugas Akhir	:	PERBANDINGAN ALGORITMA *LEVENSHTEIN* DAN *RATCLIFF/OBERSHELP* UNTUK MENDETEKSI PLAGIARISME

Tugas Akhir ini telah diperiksa dan disidangkan sebagai salah satu persyaratan untuk memperoleh gelar Sarjana pada Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Mercu Buana.

Jakarta, 07 – Juli - 2022

(Vina Ayumi S.Kom.,M.Kom)

LEMBAR PERSETUJUAN PENGUJI

NIM	:	41518110148
Nama	:	Muhamad Dicky Pradana Tarigan
Judul Tugas Akhir	:	PERBANDINGAN ALGORITMA *LEVENSHTEIN* DAN *RATCLIFF/OBERSHELP* UNTUK MENDETEKSI PLAGIARISME

Tugas Akhir ini telah diperiksa dan disidangkan sebagai salah satu persyaratan untuk memperoleh gelar Sarjana pada Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Mercu Buana.

Jakarta, 07 – Juli - 2022

(Ir. Emil R. Kaburuan, Ph.D, IPM)

LEMBAR PENGESAHAN

NIM	:	41518110148
Nama	:	Muhamad Dicky Pradana Tarigan
Judul Tugas Akhir	:	PERBANDINGAN ALGORITMA *LEVENSHTEIN* DAN *RATCLIFF/OBERSHELP* UNTUK MENDETEKSI PLAGIARISME

Tugas Akhir ini telah diperiksa dan disidangkan sebagai salah satu persyaratan untuk memperoleh gelar Sarjana pada Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Mercu Buana.

Jakarta, 10 – Juli - 2022

Menyetujui,

(Yaya Sudarya Triana,Ph.D.)
Dosen Pembimbing

Mengetahui,

(Wawan Gunawan, S.Kom, MT)	(Ir. Emil R. Kaburuan, Ph.D., IPM.)
Koord. Tugas Akhir Teknik Informatika	Ka. Prodi Teknik Informatika

ABSTRAK

Nama	:	Muhamad Dicky Pradana Tarigan
NIM	:	41518110148
Pembimbing TA	:	Yaya Sudarya Triana, M.Kom., Ph.D.
Judul	:	PERBANDINGAN ALGORITMA LEVENSHTEIN DAN RATCLIFF/OBERSHELP UNTUK MENDETEKSI PLAGIARISME

Dalam beberapa dekade terakhir, insiden plagiarisme meningkat dan telah menimbulkan keprihatinan besar di dunia akademik. Untuk mengatasi praktik plagiarisme, tidak cukup hanya mengingatkan kepada mahasiswa bahwa tindakan tersebut tidak boleh dilakukan, maka harus ada sistem atau metode untuk mendeteksi plagiarisme agar meminimalisir tindakan plagiarisme. Di dalam penelitian ini akan membandingkan metode *Levensthein* dan metode *Ratcliff/Obershelp* untuk mengetahui metode mana yang lebih baik dan efisien untuk mendeteksi plagiarisme. Dari hasil pengujian yang dilakukan didapatkan hasil nilai persentase yang berbeda-beda, dimana algoritma *Levensthein* menghasilkan nilai rata-rata presentase yang tinggi.Hal ini menunjukkan bahwa algoritma *Levensthein* terbukti lebih baik dibandingkan dengan algoritma *Ratcliff/obershelp* dalam mendeteksi plagiarisme.

Kata kunci: Plagiarisme , *Levensthein* , *Ratcliff/Obershelp*

ABSTRACT

Name	:	Muhamad Dicky Pradana Tarigan
Student Number	:	41518110148
Counsellor	:	Yaya Sudarya Triana, M.Kom., Ph.D.
Title	:	PERBANDINGAN ALGORITMA LEVENSHTEIN DAN RATCLIFF/OBERSHELP UNTUK MENDETEKSI PLAGIARISME

In recent decades, the incidence of plagiarism has increased and has caused great concern in the academic world. To resolve the practice of plagiarism, it is not enough just to remind students that such actions should not be carried out, then there must be a system or method to detect plagiarism in order to minimize plagiarism actions. In this study, we will compare the Levensthein method and the Ratcliff/Obershelp method to find out which method is better and more efficient for detecting plagiarism. From the results of the tests carried out, different percentage values obtained where the Levensthein algorithm produces a high average percentage value. This indicates that the Levensthein algorithm is proven to be better than the Ratcliff/obershelp algorithm in detecting plagiarism.

Key words: *Plagiarism , Levensthein , Ratcliff/Obershelp*

KATA PENGANTAR

Puji syukur kita panjatkan kepada kehadirat Allah Subhanahuwata'ala yang telah melimpahkan rahmat dan hidayah-Nya sehingga Tugas Akhir ini dapat penulis selesaikan dengan baik. Tugas Akhir merupakan salah satu persyaratan untuk menyelesaikan Program Studi Strata Satu (S1) pada Jurusan Teknik Informatika Universitas Mercu Buana.

Penulis menyadari bahwa tanpa bantuan, dukungan, bimbingan serta doa dari berbagai pihak laporan Tugas Akhir ini takkan dapat selesai tepat pada waktunya. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Tuhan Yang Maha Esa.
2. Kedua Orang Tua yang telah memberikan dukungan yang tiada henti kepada penulis.
3. Bapak Yaya Sudarya Triana, M.Kom., Ph.D., selaku dosen pembimbing yang telah membimbing selama penyusunan tugas akhir ini.
4. Bapak Leonard Goeirmanto, Dr., ST, M.Sc., selaku dosen pembimbing akademik yang telah membantu dalam hal akademik penulis.
5. Bapak Ir. Emil R. Kaburuan. Ph.D.. IPM., selaku kepala program studi Teknik informatika Universitas Mercubuana.
6. Semua dosen mata kuliah yang tidak dapat disebutkan satu persatu yang telah memberikan ilmu yang bermanfaat bagi penulis.
7. Staf Program Studi Teknik Informatika Universitas Mercu Buana.

Akhir kata, penulis menyadari bahwa penelitian ini jauh dari kesempurnaan. Oleh sebab itu, kritik maupun saran selalu penulis harapkan demi menghasilkan hasil terbaik dari penelitian ini. Besar harapan penulis, semoga penelitian ini dapat memberikan manfaat sekaligus menambah pengetahuan bagi berbagai pihak.

Jakarta, 23 Juni 2022

Penulis

DAFTAR ISI

NASKAH JURNAL

Terakreditasi SINTA Peringkat 2
Surat Keputusan Direktur Jenderal Pendidikan Tinggi, Riset, dan Teknologi, Nomor: 158/E/KPT/2021
masa berlaku mulai Volume 5 Nomor 2 Tahun 2021 sampai Volume 10 Nomor 1 Tahun 2026

Terbit online pada laman web jurnal: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 6 No. x (2022) x - x ISSN Media Elektronik: 2580-0760

PERBANDINGAN ALGORITMA LEVENSHTEIN DAN RATCLIFF/OBERSHELP UNTUK MENDETEKSI PLAGIARISME

Muhamad Dicky Pradana Tarigan[1], Wahyu Wicaksana[2], Yaya Sudarya Triana[3]
[1,2]Teknik Informatika, Fakultas Ilmu Komputer, Universitas Mercu Buana Jakarta
[3]Sistem Informasi, Fakultas Ilmu Komputer, Universitas Mercu Buana Jakarta
[1]yaya.sudarya@mercubuana.ac.id*, [2]41518110148@student.mercubuana.ac.id, [3]41518110138@student.mercubuana.ac.id

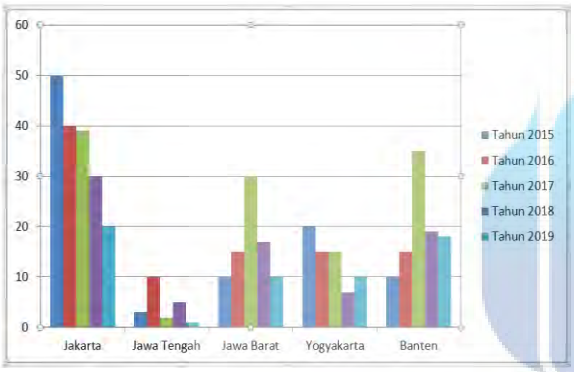
Abstrak
In recent decades, the incidence of plagiarism has increased and has caused great concern in the academic world. To resolve the practice of plagiarism, it is not enough just to remind students that such actions should not be carried out, then there must be a system or method to detect plagiarism in order to minimize plagiarism actions. In this study, we will compare the Levensthein method and the Ratcliff/Obershelp method to find out which method is better and more efficient for detecting plagiarism. From the results of the tests carried out, different percentage values obtained where the Levensthein algorithm produces a high average percentage value. This indicates that the Levensthein algorithm is proven to be better than the Ratcliff/obershelp algorithm in detecting plagiarism.
Keywords: *Plagiarism , Levensthein , Ratcliff/Obershelp*

Abstrak
Dalam beberapa dekade terakhir, insiden plagiarisme meningkat dan telah menimbulkan keprihatinan besar di dunia akademik. Untuk mengatasi praktik plagiarisme, tidak cukup hanya mengingatkan kepada mahasiswa bahwa tindakan tersebut tidak boleh dilakukan, maka harus ada sistem atau metode untuk mendeteksi plagiarisme agar meminimalisir tindakan plagiarisme. Di dalam penelitian ini akan membandingkan metode *Levensthein* dan metode *Ratcliff/Obershelp* untuk mengetahui metode mana yang lebih baik dan efisien untuk mendeteksi plagiarisme. Dari hasil pengujian yang dilakukan didapatkan hasil nilai persentase yang berbeda-beda, dimana algoritma *Levensthein* menghasilkan nilai rata-rata presentase yang tinggi.Hal ini menunjukkan bahwa algoritma *Levensthein* terbukti lebih baik dibandingkan dengan algoritma *Ratcliff/obershelp* dalam mendeteksi plagiarisme.
Kata kunci: : Plagiarisme , Levensthein , Ratcliff/Obershelp.

1. Pendahuluan

Kedatangan Internet telah menyebabkan semakin banyak sumber yang tersedia dan mudah untuk dijiplak. Dalam beberapa tahun terakhir, insiden plagiarisme meningkat dan telah menimbulkan keprihatinan besar di dunia akademik. Ada beberapa kasus yang

terjadi tidak hanya dunia akademik. Salah satu dampak yang terjadi akibat plagiarisme juga menimbulkan pelanggaran hak kekayaan intelektual dalam beberapa tahun terakhir. Pesatnya kemajuan teknologi memudahkan untuk memperoleh literasi di dunia digital. Hal ini memperumit masalah terkait perlindungan hak cipta, dimana masalah pelanggaran buku yang sebelumnya hanya sebatas penyalinan saja, namun sekarang banyak digunakan tanpa sepengetahuan penulis dan penerbit. Berikut ini adalah data dampak dari plagiarisme dalam pelanggaran Hak Kekayaan Intelektual (HKI) yang bersumber dari Laporan Tahunan DJKI 2020.



Gambar 1. Kasus Plagiarisme dalam pelanggaran HKI

Bagi mahasiswa dalam menentukan judul skripsi merupakan suatu hal yang sangat penting. Dalam menentukan judul skripsi, Penulisan Judul Tugas akhir didasarkan pada keadaan pengetahuan saat ini melalui penggabungan ide-ide dari penulis yang berbeda. Dalam beberapa tahun terakhir, Kejadian plagiarisme yang semakin meningkat, menimbulkan keprihatinan besar di kalangan akademisi. [1]. Untuk mengatasi praktik plagiarisme, tidak cukup hanya mengingatkan kepada mahasiswa bahwa tindakan tersebut tidak boleh dilakukan, maka harus ada sistem atau metode untuk mendeteksi plagiarisme agar meminimalkan kecurangan dalam praktik plagiarisme.

Di dalam penelitian ini akan membandingkan metode *Levensthein* dan metode *Ratcliff/Obershelp* untuk mengetahui metode mana yang lebih baik dan efisien untuk mendeteksi plagiarisme. beberapa langkah sebelum masuk ke metode utama. Tahap pertama adalah pengumpulan data. Tahap kedua yaitu *pre-processing* dan tahap yang terakhir adalah proses membandingkan tingkat kesamaan dokumen judul skripsi dalam bentuk persentase menggunakan metode yang akan diteliti.

Algoritma pencocokan pola *Ratcliff/Obershelp* telah digunakan sebagai teknik evaluasi untuk normalisasi mikroteks. Algoritma Ratcliff / Obershelp dapat mengembalikan persentase yang menunjukkan seberapa mirip kedua string tersebut. Keunggulan utama dari algoritma ini adalah dapat mendeteksi kecocokan substring dengan cepat dan mudah. [2].

Levenshtein adalah metrik string yang digunakan untuk mengukur perbedaan antara dua urutan. *Secara informal, jarak Levenshtein antara dua kata adalah jumlah minimum manipulasi karakter (menyisipkan, menghapus, atau mengganti) yang diperlukan untuk mengubah satu kata ke kata lain.*[3].

Deteksi plagiarisme dapat dilakukan secara online atau offline dengan pemeriksa plagiarisme. Namun, memeriksa dokumen dengan pemeriksa plagiarisme seperti *Turnitin, Dupli Checker, Copyleaks, PaperRater, Grammarly* dan lain-lain membutuhkan biaya tambahan. Beberapa penelitian telah dilakukan untuk mendeteksi plagiarisme. *BM25* dan *Rabin Karp* adalah contoh Metode pemeriksa plagiarisme. *BM25* didasarkan pada *tfidf*, sedangkan *RabinKarp* didasarkan pada *hash*. Untuk mendeteksi plagiarisme, perlu mengetahui kinerja setiap metode [4]. kinerja menggunakan *stemmer* pada algoritma *winowwing* dalam mengukur nilai plagiarisme dan kinerja *Nazief-Adriani Stemmer* pada algoritma *Winnowing* lebih unggul daripada *Porter Stemmer* dengan mengurangi kinerja sebesar 0,28% dari nilai kesamaan. *Porter Stemmer*, di sisi lain, unggul dengan meningkatkan waktu pemrosesan sebesar 69%. [5].

State of the art penelitian sebelumnya memberikan gambaran bahwa teknik pendeteksian plagiarisme sudah dilakukan

dengan berbagai macam metode mulai dari metode seperti B*M25,Rabin Karp, Nazief & Adriani* dan *Winnowing*. Kelebihan penelitian ini mendeteksi plagiarisme menggunakan algoritma *Levensthein* dan *Ratcliff /obershelp* yang jarang diteliti dan khusus untuk membandingkan metode *Levensthein* dengan metode *Ratcliff/obershelp* belum ada penelitian secara langsung untuk kasus plagiarisme terutama pada dokumen skripsi. Penelitian ini juga bertujuan untuk mengetahui hasil perbandingan antara metode *Levensthein* dan *Ratcliff /obershelp* dalam mendeteksi plagiarisme. Sehingga dapat ditemukan metode yang efektif dan memiliki tingkat akurasi yang baik. Oleh karena itu pada penelitian ini membandingkan kedua algoritma untuk pendeteksian plagiarisme dokumen skripsi.

2. METODE PENELITIAN

2.1. Literatur Review

Berikut ini beberapa tinjauan pustaka yang mencakup dari penulisan inti, pembahasan mengenai beberapa istilah dalam tulisan ini

2.1.1 Plagiarisme

Plagiarisme adalah tindakan menyalin dan menempelkan produk intelektual orang lain dan menyalahgunakannya tanpa mengenal nama penulis, penemu, dan pendiri aslinya[6].plagiarisme dapat didefinisikan sebagai perampasan ide, kata-kata, proses atau hasil orang lain tanpa pengakuan, kredit atau kutipan yang tepat[7] Menurut Sastroasmoro, ambang batas plagiarisme adalah ambang batas untuk menentukan apakah suatu dokumen termasuk dalam perbuatan plagiarisme, terdapat 3 klasifikasi dalam menentukan ambang batas plagiarism, yaitu:

1. Plagiarisme ringan : < 30%
2. Plagiarisme sedang : < 30% - 70%
3. Plagiarisme berat : > 70%

Angka tersebut dapat disesuaikan dan ditentukan oleh institusi atau universitas atau fakultas masing-masing. Untuk mengatasi praktik plagiarisme, tidak cukup dengan mengingatkan mahasiswa bahwa tindakan seperti itu tidak boleh dilakukan, tetapi harus ada sistem atau metode deteksi plagiarisme untuk meminimalisirnya[8].

2.1.2. Pre-processing data

Setelah data dikumpulkan tahap *pre-processing* dilakukan sebelum sistem mengeksekusi proses utama. Tahap *pre-processing* dalam penelitian ini menggunakan data skripsi yang berada dibeberapa *website* perpustakaan universitas. *Pre-processing* adalah mengambil teks mentah yang kemudian diproses dan menghasilkan kata-kata yang telah dibersihkan [9]. Data mentah biasanya datang dengan banyak ketidaksempurnaan seperti inkonsistensi, nilai yang hilang, *noise* dan/atau redundansi. Kinerja algoritma selanjutnya akan terganggu jika disajikan dengan data berkualitas rendah [10]. Tahapan proses *pre-processing* teks termasuk (1) *case folding*, (2) *white space removal*, (3) *tokenization*, dan (4) *punctuation.*

1. *Case folding*
   Pada teks *pre-processing* proses *case folding* bertujuan untuk mengubah semua huruf dalam dokumen teks menjadi huruf kecil, misalnya kata “Wifi” menjadi “wifi”[11].
2. *White space removal*
   Karakter, seperti spasi, tab, dan baris baru sebagian besar waktu dianggap sebagai pembatas dan tidak dihitung sebagai *token*; ini sering disebut *white space*. *White space removal* bertujuan untuk menghilangkan kata kata

pembatas spasi , karakter , *tab* dan baris baru. [12].

3. *Tokenization*
   Tokenisasi adalah proses memecah teks menjadi bagian-bagian yang lebih kecil, dan sering kali melibatkan pemrosesan awal teks untuk menghilangkan tanda baca dan mengubah semua token menjadi huruf kecil[13].
4. *Punctuation*
   Dalam *pre-processing* teks, teknik klasik dalam pencarian informasi dan penambangan data adalah dengan menghilangkan tanda baca. *Punctuation* merupakan teknik untuk menghilangkan tanda baca[14].

2.2. Levensthein

Levenshtein adalah ukuran, dari linguistik komputasi dan ilmu komputer, umumnya digunakan untuk membandingkan dua *string* berdasarkan urutan karakter. Urutan karakter mendefinisikan struktur *string*[15]. Matriks 2 (dua) dimensi digunakan dalam perhitungan nilai jarak *Levenshtein Distance (diff).* Isian nilai pada matriks tersebut adalah jumlah operasi penghapusan, penyisipan dan penukaran yang dibutuhkan dalam mengubah *string* sumber ke *string target*. Rumus operasi penghapusan, penyisipan, dan penukaran karakter yang digunakan untuk mengisi nilai matriks adalah sebagai berikut [16]:

$$D(s,t) = min\, D\,(s\ -\ 1, t)\ +\ 1\ (Penghapusan) \quad (1)$$

$$D(s,t) = min\, D(s, t\ -\ 1)\ +\ 1\ (Penyisipan) \quad (2)$$

$$D(s,t) = min\ D(s\ -\ 1, t\ -\ 1) +\ 1, sj \neq ti\ (Penukaran) \quad (3)$$

$$D(s,t) = min\, D\,(s\ -\ 1, t\ -\ 1)\,, sj\ =\ ti(Tidak\ ada\ perubahan) \quad (4)$$

s= *String* Sumber
(j) = Karakter *String* Sumber ke- j
t = *String Target*
(i) = Karakter *String Target* ke- i
D = Jarak *Levenshtein Distance*

Tabel 1. Perhitungan matiks

		S	**Y**	**S**	**T**	**E**	**M**
	0	1	2	3	4	5	6
S	1						
I	2						
S	3						
T	4						
E	5						
M	6						

Pada Tabel 1. diketahui bahwa *string* sumber "SYSTEM" memiliki 6 (enam) karakter dan *string target* "SISTEM" memiliki 6 (enam) karakter. Selanjutnya karakter ke-1 pada masing-masing *string* dibandingkan dan diketahui bahwa isi karakter ke-1 pada masing-masing *string* sama, maka nilai matriks yang diberikan sesuai dengan Persamaan (4) yaitu (1,1) = $D(1-1,1-1)$, $sj = ti$. Jadi nilai matriks yang diberikan pada (1,1) = $D(0,0)$ yang bernilai 0. Kemudian nilai matriks pada (1,1) diisi seperti pada Tabel 2.

Tabel 2. Perhitungan matiks

		S	**Y**	**S**	**T**	**E**	**M**
	0	1	2	3	4	5	6
S	1	0					
I	2						
S	3						
T	4						

E	5						
M	6						

Selanjutnya perhitungan jarak dilakukan pada karakter ke-1 *string* sumber dengan karakter ke-2 pada *string target* dan diketahui bahwa dibutuhkan operasi penyisipan karakter "Y" pada *string* sumber, maka nilai yang diberikan sesuai dengan rumus operasi penyisipan pada persamaan(2)yaitu $D(1,2) = D(1,2-1) + 1$. Jadi nilai yang diberikan pada $(1,2) = D(1,1) + 1$ yang bernilai $D(1,2) = 0 + 1 = 1$. Kemudian nilai matriks pada (1,2) diisi dengan nilai 1 seperti pada Tabel 3.

Tabel 3. Perhitungan matiks

		S	**Y**	**S**	**T**	**E**	**M**
	0	1	2	3	4	5	6
S	1	0	1				
I	2						
S	3						
T	4						
E	5						
M	6						

Perhitungan jarak *Levenshtein Distance (diff)* selanjutnya berjalan sampai semua nilai pada matriks terisi. Jarak *Levenshtein Distance* merupakan nilai yang terdapat di bawah-kanan matriks, dan pada kasus *string* sumber "SISTEM" dan *string target* "SYSTEM" berada di (6,6). Setelah dilakukan seluruh perhitungan matriks diketahui hasil dari perhitungan jarak antara *string* sumber "SISTEM" dan *string target* "SYSTEM" adalah 1 (Satu) seperti pada Tabel 4.

Tabel 4. Perhitungan matiks

		S	**Y**	**S**	**T**	**E**	**M**
	0	1	2	3	4	5	6
S	1	0	1	1	2	3	4
I	2	1	1	2	2	3	4
S	3	1	2	1	2	3	4
T	4	2	2	2	1	2	3
E	5	3	3	3	2	1	2
M	6	4	4	4	3	2	**1**

Setelah nilai jarak *Levenshtein Distance (diff)* sudah diketahui, langkah selanjutnya menghitung nilai kemiripan dengan menggunakan formulasi berikut [17].

S = *String* sumber (SISTEM)
T = *String Target* (SYSTEM)
S = 6
T = 6
diff = (1) Nilai dari jarak *string* asli ke *string* pembanding*d*

$$diff = \left(\left(\frac{diff}{\max(S,T)}\right)\right) \quad (5)$$

$$diff = \frac{1}{6} = 0.1666$$

Plagiarized value =

$$\left(1 - \left(\frac{diff}{\max(S,T)}\right)\right) x\ 100 \quad (6)$$

$$\left(1 - \left(\frac{1}{6}\right)\right)\ x\ 100 = 0.833 = 83.3\%$$

Maka didapatlah hasil perhitungan persentase plagiasi dari kata "SYSTEM" dengan kata "SISTEM" adalah 83,3%. Hasil 83,3% menunjukan bahwa data yang diinputkan terdeteksi plagiarisme.

2.3. Ratcliff/obershelp

Algoritma *Ratcliff / Obershelp* adalah metode perbandingan string yang menghitung kesamaan antara dua string[18]. Algoritma *Ratcliff / Obershelp* menggunakan proses yang sama untuk menentukan seberapa mirip kedua pola satu dimensi. Karena string teks adalah satu dimensi, algoritma ini mengembalikan nilai yang dapat digunakan sebagai faktor kepercayaan atau persentase yang menunjukkan kesamaan antara dua string[19].Algortima *Ratcliff/Obershelp* dinyatakan dengan rumus[20]:

$$D_{ro} = \frac{(2*K_m)}{(|S_1|+|S_2|)} \quad (7)$$

Mencari kesamaan kedua *string* SYSTEM dan SISTEM

Tabel 5. Perhitungan string

	[1]	[2]	[3]	[4]	[5]	[6]
S1	S	Y	S	T	E	M
S2	S	I	S	T	E	M

1. Panjang dari *string* S1 :
   |S1 |=6

   Panjang dari *string* S2 :

   |S2 |=6

2. *Substring* yang terpanjang dari kedua *string* adalah STEM. Maka STEM disebut *anchor*, dan Km= |STEM|=4 seperti pada Tabel 6.

Tabel 6. Perhitungan string

	[1]	[2]	[3]	[4]	[5]	[6]
S1	S	Y	S	T	E	M
S2	S	I	S	T	E	M

3. Di sebelah kiri dari *anchor* tersisa kumpulan huruf SY dan SI. *Substring* yang terpanjang dan memiliki kesamaan dari kumpulan huruf tersebut adalah S. Maka, Km= 4+|S|=5 seperti pada Tabel 7.

Tabel 7. Perhitungan string

	[1]	[2]	[3]	[4]	[5]	[6]
S1	S	Y	S	T	E	M
S2	S	I	S	T	E	M

4. S adalah *substring* yang berada di awal dari kedua *string* S1 dan S2 , disebelah kiri *substring* tersebut sudah tidak terdapat huruf lagi. Pada sebelah kanan dari S, terdapat huruf Y di dalam *string* S1 dan pada *string* S2 huruf I. Karena kedua huruf tersebut berbeda maka mereka tidak cocok. Maka Km tetap sama dan dilanjutkan pada huruf sebelah kanan dari *anchor*.
5. Disebelah kanan dari *anchor* sudah tidak terdapat huruf . Maka nilai dari Km tetap 5.

Kemudian kita memiliki data untuk menghitung nilai dari *Ratcliff/Obershelp*.

Persamaan *Ratcliff/Obershelp* untuk *string* SYSTEM dan SISTEM adalah:

$$D_{ro} = \frac{(2*5)}{(|6|+|6|)} = 0.833 = 83.3\%$$

Hasil dari kedua *string* SYSTEM dan SISTEM memiliki nilai yang bisa dikatakan sama yaitu 83,3%.

3. HASIL DAN PEMBAHASAN

Pada penelitian ini akan menjelaskan mengenai perbandingan algoritma *Levensthein* dan *Ratcliff/Obershelp*. Data yang akan dianalisa adalah data dokumen skripsi atau tugas akhir yang didapatkan berdasarkan dari beberapa *website* perpustakaan universitas. Kemudian data tersebut pada tahap awal akan dilakukan *pre-processing* data. Hasil pengujian akan ditampilkan dalam bentuk tabel dan grafik dengan data nilai persentase. Data nilai persentase diperoleh dari hasil perhitungan algoritma *Levenshtein* dan algoritma *Ratcliff/Obershelp*

3.1 Pre-processing Data

Sebelum digunakan untuk analisa, dataset akan diproses pada tahap *pre-processing*. Tahap *pre-processing* data diimplementasikan menggunakan pemrograman *python*, yang memanfaatkan library dasar *pre-processing*. Pada tahap ini data dilakukan *Case folding* , *White space removal*, *Tokenization* dan *Punctuation* agar data tersebut lebih bersih.

Pre-processing data yang pertama adalah *case folding*, yaitu mengubah huruf kapital menjadi huruf kecil. Berikut ini adalah contoh proses *case folding* yang ditunjukkan pada Tabel 8.

Tabel 8. Hasil *Case folding*

Sebelum	Sesudah
PERANCANGAN APLIKASI PENGAJARAN BERBANTUAN KOMPUTER DENGAN MODEL PROBLEM SOLVING PELAJARAN MATEMATIKA SEKOLAH MENENGAH ATAS	perancangan aplikasi pengajaran berbantuan komputer dengan model problem solving pelajaran matematika sekolah menengah atas

Pre-processing data selanjutnya adalah *White space removal* bertujuan untuk menghilangkan kata kata pembatas spasi , karakter , *tab* dan baris[21]. Berikut ini adalah contoh proses *White space removal* yang ditunjukkan pada Tabel 9.

Tabel 9. Hasil *White space removal*

Sebelum	Sesudah
perancangan aplikasi pengajaran berbantuan komputer dengan model problem solving pelajaran matematika sekolah menengah atas	perancangan aplikasi pengajaran berbantuan komputer dengan model problem solving pelajaran matematika sekolah menengah atas

Pre-processing data selanjutnya adalah *Tokenization*, yaitu memisahkan kata berdasarkan spasi[22]. Pemisahan dilakukan agar setiap kata dapat dianalisis dengan mudah. Berikut ini adalah contoh proses *Tokenization* yang ditunjukkan pada Tabel 10.

Tabel 10. Hasil *Tokenization*

Sebelum	Sesudah
perancangan aplikasi pengajaran berbantuan komputer dengan model problem solving pelajaran matematika sekolah menengah atas	‘perancangan’ ‘aplikasi’ ’pengajaran’ ’berbantuan’ ‘komputer’ ‘dengan’ ’model’ ’problem’ ‘solving’ ‘pelajaran’ ‘matematika’ ‘sekolah’ ‘menengah’ ‘atas’

Pre-processing data terakhir adalah *Punctuation*, yaitu menghilangkan tanda baca, angka numerik dan symbol unik lainnya. Berikut ini adalah contoh proses *Punctuation* yang ditunjukkan pada Tabel 11.

Tabel 11. Hasil *Punctuation*

Sebelum	Sesudah
perancangan aplikasi pengajaran berbantuan komputer dengan model problem solving pelajaran matematika sekolah menengah atas	perancanganaplikasipengajaranberbantuankomputerdenganmodelproblemsolvingpelajaranmatematikasekolahmenengahatas

3.2 Penerapan Algoritma Levenshtein

```
print('levenshtein', Levenshtein.ratio
( df_text.iloc[0]['Dokumen A_punc'],
   df_text.iloc[0]['Dokumen B_punc']) )
```

```
levenshtein 0.8809523809523809
```

Gambar 2. Penerapan algoritma levenshtein

Terlihat pada Gambar 2. Penerapan algoritma *Levenshtein* dengan menggunakan bahasa pemrograman *python* menampilkan hasil nilai persentase plagiarisme pada judul skripsi yang ditampilkan pada Tabel 12.

Tabel 12. Hasil penerapan algoritma levenshtein

Dokumen	Dok1	Dok2	Dok3	Dok4	Dok5
Dok1	100%	88%	69%	67%	65%
Dok2	88%	100%	70%	68%	67%
Dok3	69%	70%	100%	78%	59%
Dok4	67%	68%	78%	100%	61%
Dok5	65%	67%	59%	61%	100%

Tabel 12. menunjukkan nilai persentase yang diperoleh pada penerapan algoritma *Levenshtein* memiliki nilai yang besar, nilai rata – rata persentase algoritma *Levenshtein* sebesar 75.36% .

3.3 Penerapan Algoritma Ratcliff/obershelp

```
print('Ratcliff/obershelp', difflib.SequenceMatcher
      (None, df_text.iloc[0]['Dokumen A_punc'],
       df_text.iloc[0]['Dokumen B_punc']).ratio() )
```

```
Ratcliff/obershelp 0.8809523809523809
```

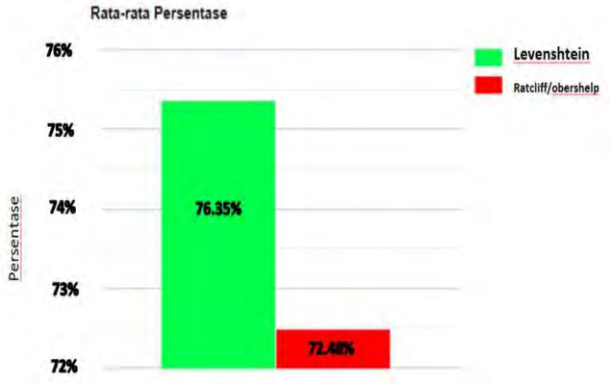
Gambar 3. Penerapan algoritma ratcliff/obershelp

Terlihat pada Gambar 3. Penerapan algoritma *Ratcliff/obershelp* dengan menggunakan bahasa pemrograman *python* menampilkan hasil nilai persentase plagiarisme pada judul skripsi yang ditampilkan pada Tabel 13.

Tabel 13. Hasil penerapan algoritma ratcliff/obershelp

Dokumen	Dok1	Dok2	Dok3	Dok4	Dok5
Dok1	100%	88%	63%	63%	60%
Dok2	85%	100%	67%	66%	67%
Dok3	61%	67%	100%	78%	56%
Dok4	64%	66%	78%	100%	51%
Dok5	60%	65%	54%	53%	100%

Tabel 13. menunjukkan nilai persentase yang diperoleh pada penerapan algoritma *Ratcliff/obershelp* memiliki nilai yang lebih kecil, nilai rata – rata persentase algoritma *Ratcliff/obershelp* sebesar 72.48% .



Gambar 4. Hasil rata-rata persentase plagiarisme

Tabel 14. Hasil rata-rata persentase

Algoritma	Total
Levenshtein	**75.36%**
Ratcliff/Obershelp	**72.48%**

Terlihat pada Gambar 4 dan Tabel 14. didapatkan hasil nilai persentase yang berbeda-beda. Pada penerapan algoritma *Levenshtein* menunjukkan nilai hasil rata-rata persentase yang bernilai 75.36% . Sedangkan pada penerapan algoritma *Ratcliff/obershelp* menunjukkan nilai rata – rata persentase yang bernilai 72.48%. Algoritma *Levenshtein* menunjukkan nilai yang lebih besar, namun tidak terlalu jauh berbeda daripada nilai rata – rata persentase algoritma *Ratcliff/obershelp.* Perbedaan hasil nilai persentase dikarenakan dalam penerapan terhadap kedua algoritma tersebut dipengaruhi oleh struktur kalimat,struktur kata dan posisi kalimat yang mempengaruhi nilai hasil persentase.

4. KESIMPULAN

Penggunaan algoritma *Levenshtein* terbukti menunjukkan nilai rata – rata persentase yang diperoleh lebih besar dibandingkan algoritma *Ratcliff/Obershelp* untuk mendeteksi plagiarisme pada judul teks skripsi atau tugas akhir. Hal ini dibuktikan dalam beberapa penerapan algoritma diatas dalam mendeteksi plagiarisme beberapa dokumen skripsi.

DAFTAR RUJUKAN

[1] J. C. Torres-Diaz, J. M. Duart, and M. Hinojosa-Becerra, "Plagiarism, internet and academic success at the university," *J. New Approaches Educ. Res.*, vol. 7, no. 2, pp. 98–104, 2018, doi: 10.7821/naer.2018.7.324.

[2] W. Hidayat, E. Utami, and A. D. Hartanto, "Effect of Stemming Nazief Adriani on the Ratcliff/Obershelp algorithm in identifying level of similarity between slang and formal words," *2020 3rd Int. Conf. Inf. Commun. Technol. ICOIACT 2020*, pp. 22–27, 2020, doi: 10.1109/ICOIACT50329.2020.9331973.

[3] S. Zhang, Y. Hu, and G. Bian, "Research on string similarity algorithm based on Levenshtein Distance," *Proc. 2017 IEEE 2nd Adv. Inf. Technol. Electron. Autom. Control Conf. IAEAC 2017*, no. 1, pp. 2247–2251, 2017, doi: 10.1109/IAEAC.2017.8054419.

[4] I. N. S. W. Wijaya, K. A. Seputra, and W. G. S. Parwita, "Comparison of the BM25 and rabinkarp algorithm for plagiarism detection," *J. Phys. Conf. Ser.*, vol. 1810, no. 1, 2021, doi: 10.1088/1742-6596/1810/1/012032.

[5] A. Rahmatulloh, N. I. Kurniati, I. Darmawan, A. Z. Asyikin, and J. D. Witarsyah, "Comparison between the stemmer porter effect and nazief-adriani on the performance of winnowing algorithms for measuring plagiarism," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 9, no. 4, pp. 1124–1128, 2019, doi: 10.18517/ijaseit.9.4.8844.

[6] I. M. S. Putra, Putu Jhonarendra, and Ni Kadek Dwi Rusjayanthi, "Deteksi Kesamaan Teks Jawaban pada Sistem Test Essay Online dengan Pendekatan Neural Network," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 5, no. 6, pp. 1070–1082, 2021, doi: 10.29207/resti.v5i6.3544.

[7] H. A. Chowdhury and D. K. Bhattacharyya, "Plagiarism: Taxonomy, Tools and Detection Techniques," no. 1, 2018, [Online]. Available: http://arxiv.org/abs/1801.06323

[8] N. ALAMSYAH, "Perbandingan Algoritma Winnowing Dengan Algoritma Rabin Karp Untuk Mendeteksi Plagiarisme Pada Kemiripan Teks Judul Skripsi," *Technol. J. Ilm.*, vol. 8, no. 3, p. 124, 2017, doi: 10.31602/tji.v8i3.1116.

[9] M. Anandarajan, C. Hill, and T. Nolan, "Text Preprocessing BT - Practical Text Analytics: Maximizing the Value of Text Data," M. Anandarajan, C. Hill, and T. Nolan, Eds. Cham: Springer International Publishing, 2019, pp. 45–59. doi: 10.1007/978-3-319-95663-3_4.

[10] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Woźniak, and F. Herrera, "A survey on data preprocessing for data stream mining: Current status and future directions," *Neurocomputing*, vol. 239, pp. 39–57, 2017, doi: 10.1016/j.neucom.2017.01.078.

[11] M. A. Rosid, A. S. Fitrani, I. R. I. Astutik, N. I. Mulloh, and H. A. Gozali, "Improving Text Preprocessing for Student Complaint Document Classification Using Sastrawi," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 874, no. 1, 2020, doi: 10.1088/1757-899X/874/1/012017.

[12] A. Alakrot, L. Murray, and N. S. Nikolov, "Towards Accurate Detection of Offensive Language in Online Communication in Arabic," *Procedia Comput. Sci.*, vol. 142, pp. 315–320, 2018, doi: 10.1016/j.procs.2018.10.491.

[13] L. A. Mullen, K. Benoit, O. Keyes, D. Selivanov, and J. Arnold, "Fast, Consistent Tokenization of Natural

Language Text," *J. Open Source Softw.*, vol. 3, no. 23, p. 655, 2018, doi: 10.21105/joss.00655.

[14] S. Symeonidis, D. Effrosynidis, and A. Arampatzis, "A comparative evaluation of pre-processing techniques and their interactions for twitter sentiment analysis," *Expert Syst. Appl.*, vol. 110, pp. 298–310, 2018, doi: 10.1016/j.eswa.2018.06.022.

[15] K. N. S. Behara, A. Bhaskar, and E. Chung, "A novel approach for the structural comparison of origin-destination matrices: Levenshtein distance," *Transp. Res. Part C Emerg. Technol.*, vol. 111, no. May 2019, pp. 513–530, 2020, doi: 10.1016/j.trc.2020.01.005.

[16] A. Ene and A. Ene, "An application of Levenshtein algorithm in vocabulary learning," *Proc. 9th Int. Conf. Electron. Comput. Artif. Intell. ECAI 2017*, vol. 2017-Janua, pp. 1–4, 2017, doi: 10.1109/ECAI.2017.8166449.

[17] D. Soyusiawaty and F. Rahmawanto, "Similarity detector on the student assignment document using levenshtein distance method," *2018 Int. Semin. Res. Inf. Technol. Intell. Syst. ISRITI 2018*, pp. 656–661, 2018, doi: 10.1109/ISRITI.2018.8864339.

[18] Y. L. Joane, A. Sinsuw, and A. Jacobus, "Rancang Bangun Aplikasi Deteksi Kemiripan Dokumen Teks Menggunakan Algoritma Ratcliff/Obershelp," *J. Tek. Inform.*, vol. 11, no. 1, 2017, doi: 10.35793/jti.11.1.2017.17654.

[19] N. Izzah, N. Yusliani, and D. Roodiah, "Sistem Deteksi Kemiripan Teks Pada Berita Berbahasa Indonesia Menggunakan Algoritma Ratcliff / Obershelp," vol. 5, no. 1, pp. 1–6, 2022.

[20] E. Kalbaliyev and S. Rustamov, "Text Similarity Detection Using Machine Learning Algorithms with Character-Based Similarity Measures," 2021, pp. 11–19. doi: 10.1007/978-3-030-74728-2_2.

[21] W. A. W. A. Bakar, N. L. N. B. Josdi, M. B. Man, and Y. S. Triana, "An Evaluation of Artificial Neural Networks and Random Forests for Heart Disease Prediction," *J. Hunan Univ. Nat. Sci.*, vol. 49, no. 2, pp. 41–49, 2022, doi: 10.55463/issn.1674-2974.49.2.4.

[22] W. A. W. A. Bakar, M. A. Zuhairi, M. Man, J. A. Jusoh, and Y. S. Triana, "a Critical Review of Deep Learning Algorithm in Association Rule Mining," *J. Theor. Appl. Inf. Technol.*, vol. 100, no. 5, pp. 1487–1494, 2022.

KERTAS KERJA

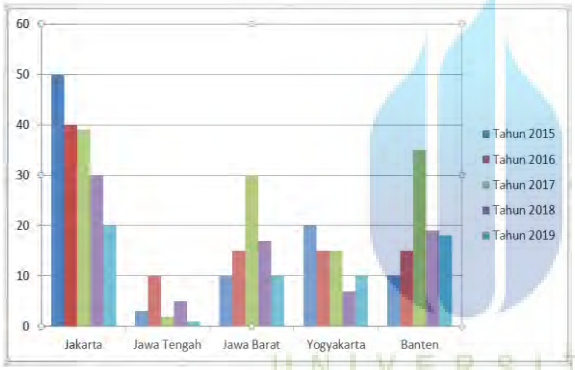
Ringkasan

Kertas kerja ini merupakan material kelengkapan artikel jurnal dengan judul "PERBANDINGAN ALGORITMA LEVENSHTEIN DAN RATCLIFF/OBERSHELP UNTUK MENDETEKSI PLAGIARISME". Kertas kerja ini berisi semua material hasil penelitian Tugas Akhir. Di dalam kertas kerja ini disajikan beberapa bagian yang terdiri dari literature review , Analisa perancangan , dataset yang digunakan , tahapan eksperimen , dan hasil eksperimen secara keseluruhan.

Bagian I membahas mengenai literature review yang berisi artikel jurnal yang menjadi dasar atau landasan dalam penelitian ini. Bagian II menjelaskan Analisa perancangan dan alur sistem rekomendasi. Bagian III menjelaskan mengenai source code yang digunakan pada penelitian ini. Bagian VI menjelaskan mengenai dataset yang digunakan. Bagian V memuat tahapan eksperimen yang disajikan dalam gambar dan table dengan penjelasan dari setiap tahapan. Bagian VI merupakan bagian terakhir dari kertas kerja ini yang menjelaskan hasil keseluruhan dari eksperimen yang telah dilakukan, meliputi penjelasannnya

PENDAHULUAN

Kedatangan Internet telah menyebabkan semakin banyak sumber yang tersedia dan mudah untuk dijiplak. Dalam beberapa tahun terakhir, insiden plagiarisme meningkat dan telah menimbulkan keprihatinan besar di dunia akademik. Ada beberapa kasus yang terjadi tidak hanya dunia akademik. Salah satu dampak yang terjadi akibat plagiarisme juga menimbulkan pelanggaran hak kekayaan intelektual dalam beberapa tahun terakhir. Pesatnya kemajuan teknologi memudahkan untuk memperoleh literasi di dunia digital. Hal ini memperumit masalah terkait perlindungan hak cipta, dimana masalah pelanggaran buku yang sebelumnya hanya sebatas penyalinan saja, namun sekarang banyak digunakan tanpa sepengetahuan penulis dan penerbit. Berikut ini adalah data dampak dari plagiarisme dalam pelanggaran Hak Kekayaan Intelektual (HKI) yang bersumber dari Laporan Tahunan DJKI 2020.



Gambar 1 . Kasus Plagiarisme dalam pelanggaran HKI

Bagi mahasiswa dalam menentukan judul skripsi merupakan suatu hal yang sangat penting. Dalam menentukan judul skripsi, Penulisan Judul Tugas akhir didasarkan pada keadaan pengetahuan saat ini melalui penggabungan ide-ide dari penulis yang berbeda. Dalam beberapa tahun terakhir, Kejadian plagiarisme yang semakin meningkat, menimbulkan keprihatinan besar di kalangan akademisi. [1]. Untuk mengatasi praktik plagiarisme, tidak cukup hanya mengingatkan kepada mahasiswa bahwa tindakan tersebut tidak boleh dilakukan, maka harus ada sistem atau metode untuk mendeteksi plagiarisme agar meminimalkan kecurangan dalam praktik plagiarisme.

Di dalam penelitian ini akan membandingkan metode *Levensthein* dan metode *Ratcliff/Obershelp* untuk mengetahui metode mana yang lebih baik dan efisien untuk mendeteksi plagiarisme. beberapa langkah sebelum masuk ke metode utama. Tahap pertama adalah pengumpulan data. Tahap kedua yaitu *pre-processing* dan tahap yang terakhir adalah proses membandingkan tingkat kesamaan dokumen judul skripsi dalam bentuk persentase menggunakan metode yang akan diteliti.

Algoritma pencocokan pola *Ratcliff/Obershelp* telah digunakan sebagai teknik evaluasi untuk normalisasi mikroteks. Algoritma Ratcliff / Obershelp dapat mengembalikan persentase yang menunjukkan seberapa mirip kedua string tersebut. Keunggulan utama dari algoritma ini adalah dapat mendeteksi kecocokan substring dengan cepat dan mudah. [2]. *Levenshtein* adalah metrik string yang digunakan untuk mengukur perbedaan antara dua urutan. *Secara informal, jarak Levenshtein antara dua kata adalah jumlah minimum manipulasi karakter (menyisipkan, menghapus, atau mengganti) yang diperlukan untuk mengubah satu kata ke kata lain.*[3].

Deteksi plagiarisme dapat dilakukan secara online atau offline dengan pemeriksa plagiarisme. Namun, memeriksa dokumen dengan pemeriksa plagiarisme seperti *Turnitin, Dupli Checker, Copyleaks, PaperRater, Grammarly* dan lain-lain membutuhkan biaya tambahan. Beberapa penelitian telah dilakukan untuk mendeteksi plagiarisme. *BM25* dan *Rabin Karp* adalah contoh Metode pemeriksa plagiarisme. *BM25* didasarkan pada *tfidf,* sedangkan *RabinKarp* didasarkan pada *hash*. Untuk mendeteksi plagiarisme, perlu mengetahui kinerja setiap metode [4]. kinerja menggunakan *stemmer* pada algoritma *winowwing* dalam mengukur nilai plagiarisme dan kinerja *Nazief-Adriani Stemmer* pada algoritma *Winnowing* lebih unggul daripada *Porter Stemmer* dengan mengurangi kinerja sebesar 0,28% dari nilai kesamaan. *Porter Stemmer*, di sisi lain, unggul dengan meningkatkan waktu pemrosesan sebesar 69%. [5].

State of the art penelitian sebelumnya memberikan gambaran bahwa teknik pendeteksian plagiarisme sudah dilakukan dengan berbagai macam metode mulai dari metode seperti B*M25,Rabin Karp, Nazief & Adriani* dan *Winnowing*. Untuk mendeteksi plagiarisme algoritma *Levensthein* dan *Ratcliff /obershelp* jarang diteliti dan khusus untuk membandingkan metode *Levensthein* dengan metode *Ratcliff/obershelp* belum ada penelitian secara langsung untuk kasus plagiarisme terutama pada judul skripsi. Penelitian ini bertujuan untuk mengetahui hasil perbandingan antara metode *Levensthein* dan *Ratcliff /obershelp* dalam mendeteksi plagiarisme. Sehingga dapat ditemukan metode yang efektif dan memiliki tingkat akurasi yang baik. Oleh karena itu pada penelitian ini membandingkan kedua algoritma untuk pendeteksian plagiarisme judul tugas akhir atau skripsi.