Penerapan *Cosine Similarity* dalam Sistem Deteksi Plagiarisme Dokumen PDF

Carlo Angkisan - 13523091[1,2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13523091@std.stei.itb.ac.id, [2]carloangkisan21@gmail.com

***Abstract*— Dalam era digital, kasus plagiarisme dokumen semakin sering terjadi, khususnya dalam lingkungan akademik. Makalah ini membahas penerapan metode *Cosine Similarity* untuk mendeteksi plagiarisme pada dokumen berformat PDF. Metode ini bekerja dengan merepresentasikan teks sebagai vektor dalam ruang multidimensi dan menghitung tingkat kemiripan berdasarkan sudut antara vektor-vektor tersebut. Hasil pengujian pada tiga skenario menunjukkan bahwa metode ini mampu mendeteksi tingkat kemiripan dengan akurat, meskipun terdapat perbedaan panjang dokumen, perubahan struktur kalimat, atau variasi kapitalisasi. Keunggulan lain adalah hasil perhitungan yang dinormalisasi dalam rentang 0 hingga 1, yang mempermudah interpretasi tingkat kemiripan antar dokumen. Berdasarkan hasil pengujian, metode ini memiliki potensi besar untuk diterapkan secara luas dalam sistem deteksi plagiarisme, dengan pengembangan lebih lanjut untuk menangani kasus plagiarisme yang lebih kompleks, seperti parafrase atau penggantian sinonim.**

***Keywords—Cosine Similarity*, Vektor, Plagiarisme, PDF.**

I. PENDAHULUAN

Di era digital yang semakin berkembang, kebutuhan akan akses informasi mengalami peningkatan yang signifikan terutama dalam dunia pendidikan. Sebagian besar dokumen akademik, seperti tugas, laporan, makalah, dan skripsi, kini disimpan dan dibagikan dalam format digital. Salah satu format yang paling umum digunakan adalah PDF (*Portable Document Format*). Keunggulan format PDF dalam menjaga tata letak dokumen secara konsisten di berbagai perangkat, membuatnya menjadi pilihan utama dalam lingkungan pendidikan.

Namun, di balik dampak positif yang diberikan, kemudahan ini juga memicu meningkatknya kasus plagiarisme. Plagiarisme adalah tindakan mengambil karya, ide, atau tulisan orang lain tanpa menyebutkan sumbernya. Fenomena ini sering terjadi di lingkungan akademik, khususnya di kalangan mahasiswa yang sering kali menghadapi tekanan untuk menyelesaikan tugas-tugas dalam waktu terbatas. Tindakan ini semakin didukung oleh kemajuan teknologi yang memungkinkan aktivitas penyalinan teks dilakukan dengan mudah.

Untuk menangani permasalahan tersebut, diperlukan sistem yang mampu mendeteksi plagiarisme secara akurat dan efisien. Salah satu pendekatan yang dapat digunakan adalah *Cosine Similarity*, yaitu sebuah metode aljabar linear yang membandingkan kesamaan antara dokumen dengan merepresentasikan teks sebagai vektor dalam ruang multidimensi. Metode ini menghitung kesamaan dengan mengukur sudut antara vektor-vektor dokumen, sehingga tingkat kesamaan dapat dinyatakan secara kuantitatif. Makalah ini bertujuan untuk mengeksplorasi penerapan metode *Cosine Similarity* untuk medeteksi plagiarisme pada dokumen berformat PDF, yang meliputi proses ekstraksi teks dari dokumen PDF, transformasi teks menjadi representasi vektor, dan analisis kesamaan antar dokumen.

II. DASAR TEORI

2.1. Plagiarisme

Plagiarisme atau plagiat adalah tindakan mengambil ide, gagasan, atau karya milik orang lain tanpa memberikan pengakuan yang semestinya. Tindakan ini menyebabkan kesan seolah-olah karya tersebut adalah hasil orisinal seseorang, sehingga dapat menimbulkan kesalahpahaman mengenai sumber asli ide atau gagasan tersebut. Menurut Peraturan Menteri Pendidikan Nasional Republik Indonesia Nomor 17 Tahun 2010, Plagiat adalah perbuatan secara sengaja atau tidak sengaja dalam memperoleh atau mencoba memperoleh kredit atau nilai untuk suatu karya ilmiah, dengan mengutip sebagian atau seluruh karya ilmiah pihak lain yang diakui sebagai karya ilmiahnya, tanpa menyatakan sumber secara tepat dan memadai [6]. Orang atau kelompok yang melakukan tindakan plagiat disebut sebagai plagiator.

Menurut Soelistyo (2011), plagiarisme dapat dikelompokkan dalam beberapa jenis, yaitu [4]:

- Berdasarkan aspek yang dicuri
  1. Plagiarisme Ide
  2. Plagiarisme Kata demi Kata
  3. Plagiarisme Sumber
  4. Plagiarisme Kepengarangan
- Berdasarkan kesengajaan
  1. Plagiarisme Sengaja
  2. Plagiarisme Tidak Sengaja

- Berdasarkan proporsi yang dibajak
    1. Plagiarisme Ringan (<30%)
    2. Plagiarisme Sedang (30%-70%)
    3. Plagiarisme Berat (>70%)

Menurut Novanta (2009), dalam mengidentifikasi plagiarisme terdapat beberapa faktor yang dapat digunakan, yaitu [4]:

1. Penggunaan kosakata
2. Perubahan kosakata yang signifikan dalam suatu teks
3. Alur teks yang membingungkan
4. Penggunaan tanda baca
5. Jumlah kemiripan teks
6. Kesalahan ejaan yang sama
7. Frekuensi kata yang sama
8. Struktur sintaksis teks
9. Referensi yang tidak jelas

2.2. PDF

PDF (*Portable Document Format*) adalah format dokumen yang dikembangkan oleh Adobe Systems. Format ini banyak digunakan dalam dokumen akademik, seperti tugas, laporan, makalah, dan skripsi, karena memiliki beberapa kelebihan, yaitu [5]:

1. Keamanan dokumen yang tinggi karena bersifat *read-only* dan dapat dilindungi dengan kata sandi.
2. Tampilan dokumen yang selalu konsisten ketika dibuka di berbagai perangkat dan sistem operasi.
3. Kemampuan kompresi berkas berukuran besar tanpa mengurangi kualitas isi dokumen.
4. Kompatibilitas universal karena dapat dibaca oleh semua sistem operasi komputer dan perangkat *mobile*.
5. Kemudahan dalam berbagi dokumen secara *online* dan mengonversi ke format lainnya.

Namun, di balik kelebihan tersebut, format PDF juga memiliki beberapa kekurangan, yaitu [5]:

1. Sulit untuk diedit karena sifatnya yang *read-only* sehingga membutuhkan aplikasi khusus untuk pengeditan.
2. Dokumen PDF pada dasarnya berupa "gambar" dari dokumen asli, sehingga menyulitkan pengguna untuk menyalin gambar, grafik, atau teks tertentu.

2.3. Vektor dan Ruang Vektor

Vektor adalah objek matematis yang memiliki besar (magnitudo) dan arah, dan sering digunakan untuk merepresentasikan berbagai fenomena dalam fisika, teknik, serta bidang lainnya. Vektor dilambangkan dengan huruf-huruf kecil dicetak tebal seperti **v** atau huruf biasa yang diberi tanda panah di atasnya seperti $\vec{v}$. Secara geometri, vektor digambarkan sebagai garis berarah.

Gambar 2.3 Vektor secara geometri
(Sumber: [1])

Ruang vektor (*vector space*) adalah adalah himpunan objek-objek yang dilengkapi dengan dua operasi di dalam himpunan tersebut, yaitu [3]:

1. Operasi penjumlahan objek-objek
2. Operasi perkalian objek dengan skalar

Salah satu contoh ruang vektor yaitu ruang Euclidean R^n. Vektor biasanya dituliskan dalam bentuk koordinat seperti $\mathbf{v} = [v_1,v_2,\ldots,v_n]$ di ruang R^n, dengan n menyatakan dimensi ruang. Panjang atau *magnitude* sebuah vektor **v** dalam ruang Euclidean R^n didefinisikan sebagai berikut.

$$\|\boldsymbol{v}\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2} = \sqrt{\sum_{i=1}^{n} v_i^2}$$

di mana $v_1,v_2,\ldots,v_n$ adalah komponen-komponen dari vektor **v**. Panjang ini merepresentasikan jarak dari titik asal ke titik yang ditunjukkan oleh vektor dalam ruang n-dimensi.

2.4. Perkalian Titik Vektor

Perkalian titik (*dot product*) adalah operasi antara dua vektor **u** dan **v** dalam ruang R^n yang menghasilkan nilai skalar. Untuk dua buah vektor **u** dan **v** yang memiliki dimensi yang sama, *dot product* didefinisikan sebagai jumlah dari hasil kali komponen-komponen yang bersesuaian dari kedua vektor tersebut.

$$\boldsymbol{u} \cdot \boldsymbol{v} = u_1v_1 + u_2v_2 + \cdots + u_nv_n = \sum_{i=1}^{n} u_iv_i$$

di mana u_i dan v_i adalah komponen-komponen dari vektor **u** dan **v**. Dot product juga dapat digunakan untuk menentukan sudut θ antara dua vektor menggunakan hubungan:

$$\boldsymbol{u} \cdot \boldsymbol{v} = \|\boldsymbol{u}\|\|\boldsymbol{v}\| \cos\theta$$

Jika $\boldsymbol{u} \cdot \boldsymbol{v} = 0$, maka **u** dan **v** saling tegak lurus.

2.5. Cosine Similarity

Cosine Similarity adalah ukuran kesamaan antara dua buah vektor berdasarkan sudut kosinus di antara mereka. Rumus *cosine similarity* antara vektor **u** dan **v** adalah sebagai berikut.

$$sim(\boldsymbol{u},\boldsymbol{v}) = \cos\theta = \frac{\boldsymbol{u} \cdot \boldsymbol{v}}{\|\boldsymbol{u}\|\|\boldsymbol{v}\|}$$

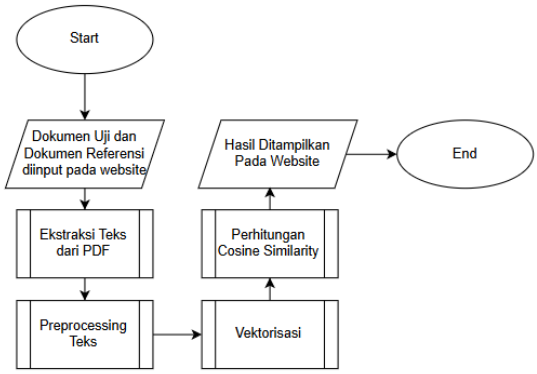
Nilai *cosine similarity* berkisar antara -1 hingga 1:

- 1 menunjukkan vektor identik
- 0 menunjukkan vektor tegak lurus
- -1 menunjukkan vektor berlawanan arah

Dalam kehidupan sehari-hari, *cosine similarity* sering digunakan pada mesin pencari untuk menampilkan hasil yang relevan dengan kueri pengguna, serta pada detektor plagiarisme untuk mengidentifikasi tingkat kesamaan antar dokumen atau teks.

III. IMPLEMENTASI

Implementasi sistem deteksi plagiarism dokumen PDF dikembangkan dengan menggunakan bahasa pemrograman Python. Sistem ini dirancang dengan pendekatan modular yang terdiri dari beberapa komponen fungsional yang saling terintegrasi. Secara garis besar, sistem ini melakukan serangkaian proses pengolahan dokumen PDF untuk menghasilkan nilai kemiripan antara dua dokumen yang dibandingkan, mulai dari ekstraksi teks dari dokumen PDF hingga penerapan C*osine Similarity* untuk mendapatkan nilai kemiripan antara dua dokumen.



Gambar 3 Alur Sistem Deteksi Plagiarisme Dokumen Menggunakan *Cosine Similarity*
(Sumber: Dokumen Penulis)

Dalam implementasinya, sistem ini memanfaatkan beberapa library pendukung, seperti PyPDF2 untuk proses ekstraksi teks dari dokumen PDF, NumPy untuk komputasi numerik dalam perhitungan *Cosine Similarity*, dan Regular Expression (re) untuk proses pembersihan teks. Sistem ini mengimplementasikan metode *Cosine Similarity* sebagai algoritma utama dalam penentuan tingkat kemiripan antar dokumen.

3.1. Implementasi Ekstraksi Teks

Komponen ekstraksi teks diimplementasikan melalui fungsi *extract_text_from_pdf* yang berperan dalam mengekstrak konten tekstual dari dokumen PDF. Fungsi ini mengimplementasikan mekanisme pembacaan dokumen PDF secara sekuensial dengan melakukan iterasi pada setiap halaman dokumen. Hasil ekstraksi teks dari setiap halaman digabungkan menjadi satu kesatuan teks dan dikonversi ke dalam format huruf kecil untuk menjamin konsistensi dalam pemrosesan selanjutnya.

```python
def extract_text_from_pdf(pdf_path):
    try:
        with open(pdf_path, "rb") as file:
            pdf_reader = PyPDF2.PdfReader(file)
            text = ""
            for page in pdf_reader.pages:
                text += page.extract_text()
            return text.lower()
    except Exception as e:
        print(f"Error membaca file {pdf_path}: {str(e)}")
        return ""
```

Gambar 3.1 Implementasi Ekstraksi Teks dari PDF
(Sumber: Dokumen Penulis)

3.2. Implementasi *Preprocessing* Teks

Preprocessing teks diimplementasikan melalui fungsi *clean_text* yang bertujuan untuk menstandarisasi teks hasil ekstraksi. Fungsi ini melakukan serangkaian proses normalisasi teks, meliputi eliminasi karakter-karakter khusus dan tanda baca, penghapusan komponen numerik dalam teks, normalisasi spasi ganda menjadi spasi tunggal, dan penghapusan spasi di awal dan akhir teks.

```python
def clean_text(text):
    text = re.sub(r"[^\w\s]", "", text)
    text = re.sub(r"\d+", "", text)
    text = re.sub(r"\s+", " ", text)
    return text.strip()
```

Gambar 3.2 Implementasi *Preprocessing* Teks
(Sumber: Dokumen Penulis)

3.3. Implementasi Vektorisasi

Proses vektorisasi diimplementasikan dalam fungsi *get_unique_words_and_vectors* yang bertugas mengonversi teks menjadi representasi vektor multidimensi. Fungsi ini mengimplementasikan tahapan vektorisasi yang meliputi pemrosesan awal teks menggunakan fungsi *clean_text*, tokenisasi teks menjadi kumpulan kata-kata individual, pembentukan kamus kata unik dari kedua dokumen, dan konstruksi vektor frekuensi kata untuk masing-masing dokumen.

```python
def get_unique_words_and_vectors(text1, text2):
    clean_text1 = clean_text(text1)
    clean_text2 = clean_text(text2)

    words1 = clean_text1.split()
    words2 = clean_text2.split()

    unique_words = sorted(list(set(words1 + words2)))

    vector1 = []
    vector2 = []

    for word in unique_words:
        vector1.append(words1.count(word))
        vector2.append(words2.count(word))

    return unique_words, vector1, vector2
```

Gambar 3.3 Implementasi Vektorisasi
(Sumber: Dokumen Penulis)

3.4. Implementasi *Cosine Similarity*

Implementasi perhitungan Cosine Similarity direalisasikan melalui fungsi *cosine_similarity*. Fungsi ini mengimplementasikan algoritma *Cosine Similarity* dengan tahapan meliputi konversi vektor ke dalam format NumPy array, kalkulasi dot product antar vektor, perhitungan magnitude masing-masing vektor, dan komputasi nilai kemiripan berdasarkan rumus *Cosine Similarity*.

```python
def cosine_similarity(vector1, vector2):
    v1 = np.array(vector1)
    v2 = np.array(vector2)

    dot_product = np.dot(v1, v2)

    magnitude1 = np.sqrt(np.sum(v1**2))
    magnitude2 = np.sqrt(np.sum(v2**2))

    if magnitude1 == 0 or magnitude2 == 0:
        return 0

    similarity = dot_product / (magnitude1 * magnitude2)
    return float(similarity)
```

Gambar 3.4 Implementasi Perhitungan *Cosine Similarity*
(Sumber: Dokumen Penulis)

3.5. Implementasi Fungsi Utama Deteksi Plagiarisme

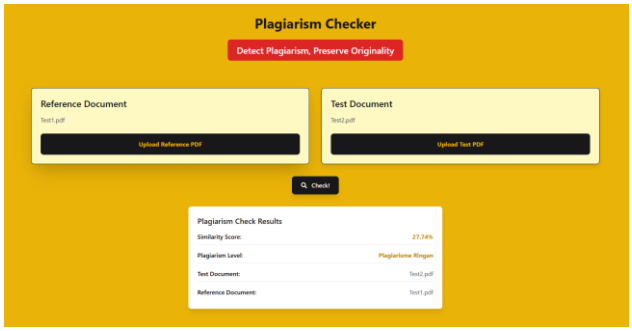
Fungsi utama untuk melakukan pengecekan plagiarisme diimplementasikan dalam dua fungsi yang saling terkait, yaitu *get_plagiarism_level* untuk klasifikasi tingkat plagiarisme dan *check_plagiarism* sebagai fungsi utama yang mengintegrasikan seluruh proses. Fungsi *check_plagiarism* akan mengembalikan sebuah data dictionary yang terdiri dari *similarity* dan *plagiarism_level* yang nantinya ditampilkan di antarmuka pengguna untuk memberikan informasi tentang nilai kemiripan dan klasifikasi plagiarisme dari dua dokumen yang dibandingkan.

```python
def get_plagiarism_level(similarity):
    if similarity > 0.70:
        return "Plagiarisme Berat"
    elif 0.30 <= similarity <= 0.70:
        return "Plagiarisme Sedang"
    elif 0 < similarity < 0.30:
        return "Plagiarisme Ringan"
    else:
        return "Tidak Plagiarisme"


def check_plagiarism(file1_path, file2_path):
    text1 = extract_text_from_pdf(file1_path)
    text2 = extract_text_from_pdf(file2_path)

    if not text1 or not text2:
        return {
            "error": "Extraction error",
            "message": "Error extracting text from PDF files",
        }

    unique_words, vector1, vector2 = get_unique_words_and_vectors(text1, text2)

    similarity = cosine_similarity(vector1, vector2)

    plagiarism_level = get_plagiarism_level(similarity)

    return {"similarity": similarity, "plagiarism_level": plagiarism_level}
```

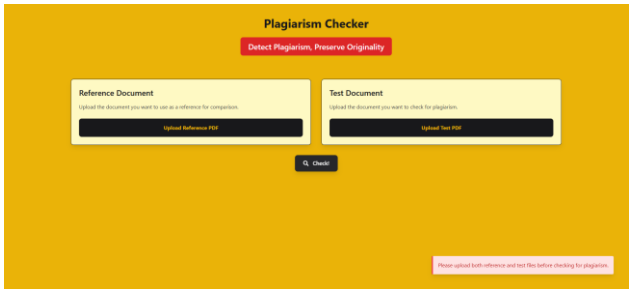
Gambar 3.5 Implementasi Fungsi Utama
(Sumber: Dokumen Penulis)

3.6. Implementasi Antarmuka Pengguna

Untuk memberikan pengalaman yang nyaman dan menyenangkan bagi pengguna, dibuat antarmuka berbasis *website*. Namun, *website* ini masih hanya dapat diakses secara lokal oleh pengguna. Antarmuka ini diimplementasikan menggunakan Vite sebagai *build tool* dengan React TypeScript dan Tailwind CSS untuk keperluan *styling*.



Gambar 3.6.1 Implementasi Antarmuka Pengguna
(Sumber: Dokumen Penulis)



Gambar 3.6.2 Tampilan Antarmuka Saat Tombol 'Check' Ditekan Tanpa File Adanya yang Diunggah
(Sumber: Dokumen Penulis)

Pada *website* ini, terdapat fitur untuk mengunggah dokumen referensi dan dokumen uji secara terpisah, masing-masing melalui tombol 'Upload Reference PDF'

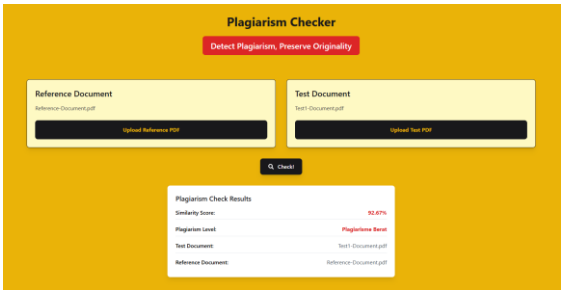
dan 'Upload Test PDF'. Setelah kedua dokumen berhasil diunggah, tombol 'Check' dapat ditekan oleh pengguna untuk memulai proses perhitungan tingkat kemiripan antar dokumen. Apabila tombol 'Check' ditekan tanpa adanya dokumen referensi atau dokumen uji yang terunggah, sistem akan menampilkan peringatan berupa pesan *alert* seperti yang ditunjukkan pada **Gambar 3.6.2**.

IV. HASIL DAN PEMBAHASAN

Pengujian sistem deteksi plagiarisme dilakukan dengan membandingkan tiga dokumen uji terhadap satu dokumen referensi, menghasilkan tiga skenario pengujian. Setiap skenario mengikuti proses yang seragam. Tahapan dimulai dari ekstraksi teks dari dokumen PDF dan *preprocessing* teks, dilanjutkan dengan pemodelan vektor. Pada tahap ini, seluruh kata unik dari kedua dokumen diidentifikasi, dan frekuensi kemunculannya dihitung. Hasilnya, dokumen uji dan dokumen referensi dimodelkan sebagai vektor berdimensi n×1, dengan n merepresentasikan jumlah kata unik dari kedua dokumen. Pemodelan ini memungkinkan perhitungan *Cosine Similarity* untuk mendapatkan nilai tingkat kemiripan antar dokumen.

Dokumen Referensi (r): Reference-Document.pdf
"Mahasiswa Informatika ITB harus memiliki kemampuan coding yang baik karena informatika adalah jurusan yang menuntut mahasiswa untuk selalu belajar coding"

4.1. Hasil Uji Skenario 1



Gambar 4.1 Hasil Uji Skenario 1
(Sumber: Dokumen Penulis)

Dokumen Uji 1 (u1): Test1-Document.pdf
"MAHASISWA INFORMATIKA ITB HARUS MEMILIKI KEMAMPUAN CODING YANG BAIK KARENA INFORMATIKA ADALAH JURUSAN YANG MENUNTUT MAHASISWA."

Kata	Frekuensi	
	u1	r
adalah	1	1
baik	1	1
belajar	0	1
coding	1	2
harus	1	1
informatika	2	2
itb	1	1
jurusan	1	1
karena	1	1
kemampuan	1	1
mahasiswa	2	2
memiliki	1	1
menuntut	1	1
selalu	0	1
untuk	0	1
yang	2	2

Secara matematis, dapat dinyatakan sebagai vektor:

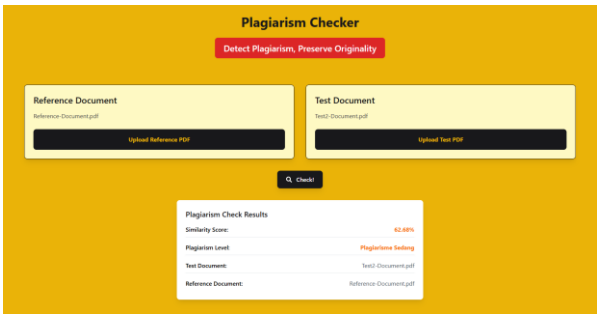
$$\boldsymbol{u1} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 2 \\ 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 1 \\ 1 \\ 0 \\ 0 \\ 2 \end{pmatrix} \qquad \boldsymbol{r} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 1 \\ 2 \\ 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 1 \\ 1 \\ 1 \\ 1 \\ 2 \end{pmatrix}$$

$$\begin{aligned} \boldsymbol{u1} \cdot \boldsymbol{r} &= \sum_{i=1}^{n} u1_i r_i \\ &= 1\times1 + 1\times1 + 0\times1 + 1\times2 + 1\times1 + \\ &\quad 2\times2 + 1\times1 + 1\times1 + 1\times1 + 1\times1 + \\ &\quad 2\times2 + 1\times1 + 1\times1 + 0\times1 + 0\times1 + \\ &\quad 2\times2 \\ &= 23 \end{aligned}$$

$$\|\boldsymbol{u1}\| = \sqrt{\sum_{i=1}^{n} u1_i^2} = \sqrt{22}$$
$$\|\boldsymbol{r}\| = \sqrt{\sum_{i=1}^{n} r_i^2} = \sqrt{28}$$

$$sim(\boldsymbol{u1}, \boldsymbol{r}) = \cos\theta = \frac{\boldsymbol{u1} \cdot \boldsymbol{r}}{\|\boldsymbol{u1}\|\|\boldsymbol{r}\|} = \frac{23}{\sqrt{22}\sqrt{28}} = 0{,}9267$$

4.2. Hasil Uji Skenario 2



Gambar 4.2 Hasil Uji Skenario 2
(Sumber: Dokumen Penulis)

Dokumen Uji 2 (u2): Test2-Document.pdf
"Mahasiswa Informatika perlu memiliki kemampuan coding yang baik dalam mengembangkan sistem"

Kata	Frekuensi	
	u2	r
adalah	0	1
baik	1	1
belajar	0	1
coding	1	2
dalam	1	0
harus	0	1
informatika	1	2
itb	0	1
jurusan	0	1
karena	0	1
kemampuan	1	1
mahasiswa	1	2
memiliki	1	1
mengembangkan	1	0
menuntut	0	1
perlu	1	0
selalu	0	1
sistem	1	0
untuk	0	1
yang	1	2

Secara matematis, dapat dinyatakan sebagai vektor:

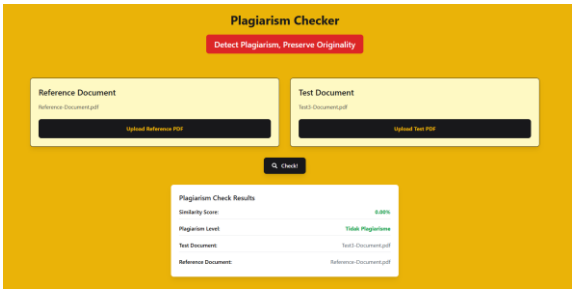
$$\boldsymbol{u2} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \qquad \boldsymbol{r} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 0 \\ 1 \\ 2 \\ 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 2 \end{pmatrix}$$

$$\boldsymbol{u2} \cdot \boldsymbol{r} = \sum_{i=1}^{n} u2_i r_i = 11$$

$$\|\boldsymbol{u2}\| = \sqrt{\sum_{i=1}^{n} u2_i^2} = \sqrt{11}$$
$$\|\boldsymbol{r}\| = \sqrt{\sum_{i=1}^{n} r_i^2} = \sqrt{28}$$

$$sim(\boldsymbol{u2}, \boldsymbol{r}) = \cos\theta = \frac{\boldsymbol{u2} \cdot \boldsymbol{r}}{\|\boldsymbol{u2}\|\|\boldsymbol{r}\|} = \frac{11}{\sqrt{11}\sqrt{28}} = 0{,}6268$$

4.3. Hasil Uji Skenario 3



Gambar 4.3 Hasil Uji Skenario 3
(Sumber: Dokumen Penulis)

Dokumen Uji 3 (u3): Test3-Document.pdf
"Pembelajaran pemrograman membutuhkan latihan dan praktik konsisten"

Kata	Frekuensi	
	u3	r
adalah	0	1
baik	0	1
belajar	0	1
coding	0	2
dan	1	0
harus	0	1
informatika	0	2
itb	0	1
jurusan	0	1
karena	0	1
kemampuan	0	1
konsisten	1	0
latihan	1	0
mahasiswa	0	2
membutuhkan	1	0
memiliki	0	1
menuntut	0	1
pembelajaran	1	0
pemrograman	1	0
praktik	1	0
selalu	0	1
untuk	0	1
yang	0	2

Secara matematis, dapat dinyatakan sebagai vektor:

$$\boldsymbol{u3} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad \boldsymbol{r} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 0 \\ 1 \\ 2 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 2 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 2 \end{pmatrix}$$

$$\boldsymbol{u3} \cdot \boldsymbol{r} = \sum_{i=1}^{n} u3_i r_i = 0$$

$$\|\boldsymbol{u3}\| = \sqrt{\sum_{i=1}^{n} u3_i^2} = \sqrt{7}$$
$$\|\boldsymbol{r}\| = \sqrt{\sum_{i=1}^{n} r_i^2} = \sqrt{28}$$

$$sim(\boldsymbol{u3}, \boldsymbol{r}) = \cos\theta = \frac{\boldsymbol{u3} \cdot \boldsymbol{r}}{\|\boldsymbol{u3}\|\|\boldsymbol{r}\|} = \frac{0}{\sqrt{7}\sqrt{28}} = 0$$

4.4. Pembahasan

Hasil pengujian pada tiga skenario menunjukkan bahwa metode *Cosine Similarity* mampu mengukur tingkat kesamaan antara dokumen uji dan dokumen referensi dengan baik. Pada skenario pertama, nilai kemiripan sebesar 0,9267 (92,67%) mengindikasikan tingkat kesamaan yang sangat tinggi. Hal ini sesuai dengan konten dokumen uji 1 yang hampir identik dengan dokumen referensi, dengan perbedaan hanya pada beberapa kata, seperti "selalu", "untuk", dan "belajar". Pada skenario kedua, nilai kemiripan mencapai 0,6268 (62,68%), yang menunjukkan adanya kesamaan sebagian antara dokumen. Beberapa kata kunci, seperti "mahasiswa", "informatika", "coding", dan "kemampuan", muncul di kedua dokumen, meskipun dalam konteks dan struktur kalimat yang berbeda. Sebaliknya, pada skenario ketiga, nilai kemiripan sebesar 0 mengindikasikan tidak adanya kesamaan sama sekali antara dokumen uji 3 dan dokumen referensi, mencerminkan perbedaan total dalam kontennya.

Dari hasil tersebut, dapat disimpulkan bahwa metode *Cosine Similarity* memiliki sejumlah keunggulan dalam mendeteksi plagiarisme. Metode ini tidak terpengaruh oleh panjang dokumen, sehingga memungkinkan perbandingan yang konsisten meskipun dokumen memiliki jumlah kata yang berbeda. Selain itu, pendekatan berbasis vektor yang digunakan membuat metode ini tidak sensitif terhadap urutan kata, sehingga tetap efektif dalam mendeteksi plagiarisme meskipun terjadi perubahan pada struktur kalimat. Sensitivitas metode ini terhadap proporsi kata yang sama antar dokumen memberikan hasil yang akurat dalam mengukur tingkat kesamaan teks. *Preprocessing* teks yang mengubah seluruh huruf menjadi huruf kecil, juga membantu mengabaikan perbedaan kapitalisasi. Dengan hasil perhitungan yang dinormalisasi pada rentang 0 hingga 1, metode ini memberikan interpretasi yang jelas dan sederhana tentang tingkat kemiripan dokumen, memudahkan pengguna dalam menentukan ambang batas yang sesuai untuk klasifikasi plagiarisme.

V. KESIMPULAN DAN SARAN

Berdasarkan hasil pengujian, metode *Cosine Similarity* mampu mendeteksi plagiarisme dengan baik, ditunjukkan oleh nilai kemiripan yang akurat pada berbagai skenario pengujian. Metode ini memiliki keunggulan utama, yaitu tidak terpengaruh oleh panjang dokumen, sensitif terhadap kemiripan kata, dan tetap efektif meskipun terdapat perubahan struktur kalimat atau perbedaan kapitalisasi. Selain itu, hasil perhitungan yang dinormalisasi mempermudah interpretasi tingkat kemiripan dokumen. Untuk pengembangan lebih lanjut, metode ini dapat dioptimalkan dengan mendukung analisis dokumen dalam format yang lebih beragam, serta meningkatkan kemampuan dalam mendeteksi bentuk plagiarisme yang lebih kompleks, seperti parafrase atau penggantian sinonim.

VI. LAMPIRAN

Program lengkap dari makalah ini dapat diakses melalui link repository github berikut:
https://github.com/carllix/Plagiarism-Checker

Video penjelasan dari makalah ini dapat diakses melalui link berikut:
https://www.youtube.com/watch?v=-5PaLiRTCS0

VII. UCAPAN TERIMA KASIH

Penulis mengucapkan puji syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmat-Nya yang melimpah dalam proses penulisan makalah ini, sehingga dapat diselesaikan tepat waktu. Penulis juga mengucapkan terima kasih kepada Ir. Rila Mandala, M.Eng., Ph.D., selaku dosen mata kuliah IF2123 Aljabar Linear dan Geometri Kelas K01, yang telah memberikan bimbingan dan ilmu pengetahuan yang sangat berharga dalam penyelesaian makalah ini. Ucapan terima kasih juga disampaikan kepada Dr. Ir. Rinaldi Munir, M.T., yang telah menyediakan *website* sebagai sumber belajar untuk mata kuliah Aljabar Linear dan Geometri, yang sangat

membantu dalam memahami materi dengan lebih baik. Selain itu, penulis juga menyampaikan terima kasih yang sebesar-besarnya kepada kedua orang tua yang selalu memberikan dukungan, semangat, dan doa yang tak terhingga, yang menjadi sumber motivasi dan kekuatan dalam setiap langkah penulis.

REFERENSI

[1] Munir, Rinaldi. 2023. "Vektor di Ruang Euclidean (Bagian 1)". https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2024-2025/Algeo-11-Vektor-di-Ruang-Euclidean-Bag1-2024.pdf (Diakses pada 26 Desember 2024).

[2] Munir, Rinaldi. 2023. "Aplikasi Dot Product pada Sistem Temu-balik Informasi (Information Retrieval System)". https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-14-Aplikasi-dot-product-pada-IR-2023.pdf (Diakses pada 26 Desember 2024).

[3] Munir, Rinaldi. 2023. "Ruang Vektor Umum (Bagian 1)". https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-15-Ruang-vektor-umum-Bagian1-2023.pdf (Diakses pada 26 Desember 2024).

[4] KajianPustaka.com. 2021. "Pengertian, Jenis dan Identifikasi Plagiarisme". https://www.kajianpustaka.com/2019/02/Plagiarisme.html (Diakses pada 26 Desember 2024).

[5] Kumparan.com. 2022. "PDF Singkatan Apa? Ini Pengertian, Fungsi, hingga Kelebihan dan Kekurangannya". https://kumparan.com/berita-hari-ini/pdf-singkatan-apa-ini-pengertian-fungsi-hingga-kelebihan-dan-kekurangannya-1yx2h1v7aWN (Diakses pada 26 Desember 2024).

[6] Kementerian Pendidikan dan Kebudayaan. 2010. "Peraturan Menteri Pendidikan dan Kebudayaan Nomor 17 Tahun 2010 tentang Pencegahan dan Penanggulangan Plagiat di Perguruan Tinggi". https://peraturan.bpk.go.id/Download/156668/Permendiknas%20Nomor%2017%20Tahun%202010.pdf (Diakses pada 26 Desember 2024).

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 28 Desember 2024

Carlo Angkisan
13523091