Pengembangan Chatbot Informasi Institut Teknologi Bandung dengan Algoritma Fuzzy Matching untuk Information Retrieval System

Lukas Raja Agripa - 13523158
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: lukasagripa27@gmail.com , 13523158@std.stei.itb.ac.id

Abstract*—Makalah ini membahas pengembangan chatbot berbasis information retrieval untuk Institut Teknologi Bandung (ITB) dengan memanfaatkan algoritma fuzzy matching dan teknik pemrosesan bahasa alami klasik. Sistem mengumpulkan dan memproses data terkait ITB melalui web scraping, dilanjutkan dengan normalisasi teks dan pembuatan database. Pertanyaan pengguna dicocokkan dengan database menggunakan kombinasi algoritma string similarity, seperti Levenshtein distance, n-gram similarity, dan TF-IDF dengan cosine similarity, sehingga mampu memberikan toleransi tinggi terhadap kesalahan penulisan (typo) dan menghasilkan jawaban yang relevan. Backend diimplementasikan menggunakan Python dan library machine learning klasik, sedangkan frontend menggunakan React untuk interaksi real-time. Pengujian menunjukkan bahwa chatbot ini dapat mengambil informasi dengan akurat meskipun terdapat kesalahan input, sehingga efektif untuk layanan informasi akademik. Pendekatan ini menunjukkan penerapan praktis fuzzy matching dan information retrieval dalam membangun chatbot yang tangguh tanpa bergantung pada model bahasa besar.

Keywords—chatbot; information retrieval; fuzzy matching; string similarity; natural language processing; typo tolerance; ITB; web scraping; TF-IDF; Levenshtein distance

I. INTRODUCTION

Era teknologi seperti saat ini yang berkembang seiring waktu, penggunaan chatbot sudah semakian marak,di hamper segala bidang, seperti kesehatan, bisnis, dan pendidikan [1]. Pada konteks ini, chatbot memberikan solusi yang efektif untuk meningkatkan interaksi antara pengguna dan website. Chatbot dapat digunakan untuk membantu kegiatan tanya jawab yang sering kali ditanyakan berulang-ulang yang dapat dilakukan setiap saat tanpa terkendala waktu. Dengan memberikan respons yang instan dan personal, chatbot merampingkan komunikasi, mengurangi waktu respons, dan memberdayakan siswa untuk menemukan informasi dan sumber daya secara efisien [1].

Hampir semua chatbot yang canggih menggunakan AI (*Artificial Intelligence),* di mana menggunakan machine learning/ NLP *(Natural Language Processing)* untuk memahami pertanyaan pengguna dan memberikan respons otomatis terhadap pertanyaan tersebut [2]. Adapula yang berbasis aturan (rule-based), di mana tidak menggunakan AI, dan hanya menjawab berdasarkan pola tertentu. Seperti misalnya adalah chatbot FAQ sederhana yang hanya merespons "1", "2", atau "3". Dan mungkin juga menggunakan sedikit NLP dalam *preprocessing* untuk removal, stemming dan lain sebagainya sehingga mendapatkan data yang lebih baik.

Di zaman sekarang pun, informasi beredar begitu cepat dan luas. Apalagi LLM hampir bisa menjawab untuk seluruh pertanyaan dari pengguna dalam segala situasi. Hal ini memunculkan banyak sekali LLM yang mumpuni untuk menjawab segala kebutuhan pengguna. Bahkan, di Platfrom X pun juga terdapat LLM bernamakan *Grok* yang dapat dipanggil oleh user untuk memberikan bantuan. Tidak selalu dalam platform media sosial terkenal, Saat ini berbagai institusi perpustakaan terus meningkatkan kualitas pelayanannya melalui pemanfaatan teknologi digital yaitu chatbot. Pemanfaatan chatbot tersebut dinilai mampu menciptakan ekosistem *smart library* sehingga penting untuk terus dikembangkan agar pelanggan merasa puas terhadap pelayanan informasi yang diberikan dengan *smart service*, yang tidak hanya berfokus pada pengembangan sistem atau teknologi barusecara teknis saja, namun berupaya untuk memberikan pelayanan sesuai dengan kebutuhan pengguna, khususnya pada akses fasilitas perpustakaan [3].

Sementara, di Institut Teknologi Bandung sendiri, penulis menyadari bahwa dengan informasi terpisah antara yang satu dengan yang lainnya, sehingga penulis sendiri merasakan harus membuka satu per satu page yang mungkin cocok. Bahkan LLM, seperti *ChatGPT 4.1* berpotensi memberikan informasi yang salah, oleh karena *modelling* dan *smart feature deep learning*, yang di mana merupakan pedang bermata dua untuk pengguna. Seperti salah satunya, di bawah ini terdapat respon yang kurang tepat untuk menunjukkan apa saja fakultas/sekolah di ITB melalui ChatGPT, di mana menjawab SITK, yang bukan bagian dari fakultas di ITB.


Institut Teknologi Bandung (ITB) memiliki beberapa fakultas dan sekolah (karena ITB membedakan antara *fakultas* dan *sekolah*). Berikut daftar lengkap **fakultas dan sekolah di ITB** (per 2024):

Fakultas di ITB

1. **Fakultas Ilmu dan Teknologi Kebumian (FITB)**
   Contoh program studi: Teknik Geologi, Meteorologi, Oseanografi
2. **Fakultas Teknik Pertambangan dan Perminyakan (FTTM)**
   Contoh program studi: Teknik Perminyakan, Teknik Pertambangan, Teknik Geofisika
3. **Fakultas Teknologi Industri (FTI)**
   Contoh program studi: Teknik Kimia, Teknik Fisika, Teknik Industri
4. **Fakultas Teknik Sipil dan Lingkungan (FTSL)**
   Contoh program studi: Teknik Sipil, Teknik Lingkungan, Teknik dan Pengelolaan Sumber Daya Air
5. **Fakultas Teknik Mesin dan Dirgantara (FTMD)**
   Contoh program studi: Teknik Mesin, Teknik Dirgantara, Teknik Material
6. **Fakultas Seni Rupa dan Desain (FSRD)**
   Contoh program studi: Desain Produk, Desain Interior, Seni Rupa Murni

Sekolah di ITB

1. **Sekolah Teknik Elektro dan Informatika (STEI)**
   Contoh program studi: Teknik Elektro, Teknik Informatika, Sistem dan Teknologi Informasi
2. **Sekolah Farmasi (SF)**
   Contoh program studi: Sains dan Teknologi Farmasi, Farmasi Klinik dan Komunitas
3. **Sekolah Arsitektur, Perencanaan dan Pengembangan Kebijakan (SAPPK)**
   Contoh program studi: Arsitektur, Perencanaan Wilayah dan Kota
4. **Sekolah Bisnis dan Manajemen (SBM)**
   Contoh program studi: Manajemen, Kewirausahaan
5. **Sekolah Ilmu dan Teknologi Hayati (SITH)**
   Contoh program studi: Biologi, Mikrobiologi, Rekayasa Hayati
6. **Sekolah Ilmu dan Teknologi Hayati – Program Rekayasa (SITH-R)**
   Contoh program studi: Rekayasa Pertanian, Rekayasa Kehutanan
7. **Sekolah Ilmu dan Teknologi Hayati – Program Sains (SITH-S)**
   h program studi: Biologi, Mikrobiologi (lebih fokus ke sains dasar)
8. **Sekolah Ilmu dan Teknologi Kebumian (SITK)**
   Merupakan bagian baru yang mendalami kebumian secara multidisiplin

Gambar 1.1 Beberapa kesalahan dari jawaban LLM

(Sumber : ChatGPT 4.1 oleh penulis pada 23 Juni 2025 pukul 14.41)

Menurut penulis, ada baiknya untuk mengembangkan chatbot khusus Institut Teknologi Bandung sendiri. Mungkin tidak terlalu bergantung dengan NLP, konsep *deep learning*, seperti *Generative Language Modeling, Autoregressive Generation, Attention Mechanism, Prompt Engineering, dan Fine-tuning/Instruction Tuning.* Tetapi dengan *information-retrieval based*, yang diambil dari *database* ITB tersendiri, kemudian dilakukan pencocokan dengan algoritma string matching, yaitu fuzzy matching, dengan sedikit bantuan *machine learning klasik* dan teknik NLP klasik dan sebagainya kemudian sistem mencari jawaban yang paling mirip. Sebagaimana juga sesuai dengan tema yang diberikan oleh mata kuliah yang penulis ambil.

II. Dasar Teori

A. *Information Retrieval*

Information Retrieval (IR) adalah proses menemukan informasi yang relevan dari kumpulan data besar berdasarkan kebutuhan pengguna [5]. IR berfokus pada pencarian dan pengambilan dokumen atau data yang sesuai dengan permintaan (*query*) pengguna dari koleksi data yang tersimpan di komputer, baik itu dokumen teks, halaman web, katalog online, maupun data multimedia seperti gambar, audio, dan video [6]. Tujuan utama dari sistem IR adalah memberikan akses yang mudah dan cepat bagi pengguna untuk menemukan informasi yang mereka butuhkan dari koleksi dokumen yang sangat besar. Sistem IR tidak serta-merta memberikan pengetahuan baru kepada pengguna, melainkan menginformasikan keberadaan dan lokasi dokumen yang relevan dengan kebutuhan mereka. Biasanya *Information Retrieval* mengandung komponen utama seperti *Query Processing, Indexing, Retrieval Model,* dan juga *Ranking*.

Adapun alur program dan IR secara umum yang meliputi:

1. **Pengguna mengajukan *query*** untuk mendeskripsikan kebutuhan informasinya.
2. **Sistem IR melakukan pencarian** pada indeks dokumen untuk menemukan kandidat dokumen yang relevan
3. **Sistem menampilkan hasil** berupa daftar dokumen yang telah diurutkan berdasarkan tingkat relevansi terhadap *query* pengguna.

[6]

Salah satu aplikasi paling umum dari IR adalah mesin pencari (*search engine*) di internet, yang memungkinkan pengguna mencari halaman web sesuai kebutuhan mereka secara cepat dan efisien.

B. *Similarity-based Matching*

Similarity-based matching adalah pendekatan pencocokan teks yang mengukur tingkat kemiripan antara *query* dan dokumen berdasarkan kata-kata yang terkandung di dalamnya. Berbeda dengan *exact matching* yang hanya mencari kecocokan persis, *similarity-based matching* dapat mengenali kemiripan meskipun terdapat variasi urutan kata atau sebagian kata berbeda. Dalam pengembangan chatbot, akan sangat membantu algoritma *fuzzy matching* dalam menentukan kata yang paling *match*, karena *fuzzy* fokus pada toleransi typo, salah ketik, atau variasi ejaan, sedangkan *similarity-based* yang dimaksud adalah untuk mengukur kemiripan makna atau isi antara dua kalimat/teks, bukan sekadar ejaan, jadi menangkap pertanyaan yang berbeda kata tapi sama maksud.

Adapun yang digunakan dalam pengembangan chatbot ini, yaitu sebagai berikut:

- Word Overlap

Menghitung jumlah kata yang sama antara *query* dan dokumen, lalu membaginya dengan jumlah kata pada *query*. Semakin banyak kata yang sama, semakin tinggi skor kemiripan.

```
Algorithm WordOverlap(Query, Document):
    QueryWords = set(tokenize(Query))
    DocWords = set(tokenize(Document))
    Overlap = QueryWords ∩ DocWords
    Score = |Overlap| / |QueryWords|
    return Score
```

- Jaccard Similarity

Mengukur rasio antara jumlah kata yang sama (irisan) dan jumlah total kata unik (gabungan) dari *query* dan dokumen.

Nilai 1 berarti identik, nilai 0 berarti tidak ada kata yang sama.

```
AlgorithmJaccardSimilarity(Query,
Document):
    QueryWords = set(tokenize(Query))
    DocWords = set(tokenize(Document))
    Intersection = QueryWords ∩ DocWords
    Union = QueryWords ∪ DocWords
    if |Union| == 0:
        return 0
    Score = |Intersection| / |Union|
    return Score
```

C. Algoritma Fuzzy Matching

Algoritma fuzzy matching adalah teknik pencocokan string yang mengidentifikasi kemiripan antara dua teks meskipun terdapat perbedaan ejaan, struktur, atau kesalahan ketik. Teknik ini digunakan untuk mengatasi ketidakpastian dalam data, seperti: variasi penulisan, kesalahan ketik, dan perbedaan format.

Algoritma *fuzzy matching ini* sangat cocok sebagai algoritma utama yang digunakan dalam pengembangan chatbot, karena meningkatkan kemampuan pencocokan dan pemahaman input pengguna yang tidak selalu persis sama dengan data atau pertanyaan yang ada di database chatbot. Metode ini memungkinkan chatbot mengenali dan mencocokkan pertanyaan atau pernyataan pengguna meskipun terdapat variasi penulisan, kesalahan ketik, atau perbedaan kecil dalam kata-kata yang digunakan, sehingga chatbot dapat memberikan jawaban yang relevan dan tepat.

Adapun yang digunakkan dalam pengembangan chatbot ini, yaitu sebagai berikut:

- Levenshtein Distance

Levenshtein distance mengukur perbedaan antara dua string dengan menghitung jumlah operasi edit minimal yang diperlukan untuk mengubah satu string menjadi string lainnya. Operasi edit tersebut terdiri dari tiga jenis: penyisipan (insertion), penghapusan (deletion), dan penggantian (substitution) karakter [7].

Misalkan dalam notasi algoritmik berikut, `lev(a, b)` adalah jarak Levenshtein antara string a dan b.

```
lev(i, j) =
  if min(i, j) = 0: max(i, j)
  else if a[i] == b[j]: lev(i-1, j-1)
  else : 1 + min(
    lev(i-1, j),     # lakukan hapus
    lev(i, j-1),     # lakukan tambah/sisip
    lev(i-1, j-1)    # lakukan ganti
  )
```

- N-Gram Similarity

N-gram membantu menangkap pola lokal dalam teks, seperti frasa umum atau hubungan kontekstual antar kata, yang tidak dapat diperoleh hanya dari analisis kata tunggal. Artinya ini teknik dasar dalam pengolahan bahasa alami yang memecah teks menjadi rangkaian berurutan dari *n* item untuk menangkap pola lokal dan konteks dalam teks, yang sangat berguna dalam berbagai aplikasi NLP seperti pemodelan bahasa, klasifikasi, dan koreksi ejaan [8].

Sebenarnya N-Gram ini bukan termasuk dalam *fuzzy matching*, tetapi sebuah teknik representasi atau fitur ekstraksi teks. N-gram digunakan sebagai metode representasi untuk meningkatkan fleksibilitas pencocokan, khususnya dalam fuzzy matching berbasis kemiripan karakter atau kata, dalam kasus ini digunakan oleh *Jaccard Similarity over N-Gram.*

- Soundex Matching

Soundex Matching adalah algoritma fonetik yang mengubah kata menjadi kode berdasarkan pengucapan dalam bahasa Inggris, memungkinkan pencocokan kata atau nama yang terdengar sama walaupun ejaannya berbeda [9]. Hal ini sangat cocok untuk informasi Institut Teknologi maupun website internasional, yang sering kali terdapat kata dari bahasa inggris. Misalkan Bandung dan Bandong, ketika pengguna salah menuliskan *query*, dengan kebetulan memiliki kode fonetik yang sama, maka akan dikatakan mirip.

```
SOUNDEX(s):
    1. Convert s ke huruf besar.
    2. Pertahankan huruf pertama dari s
sebagai hasilnya.
    3. For each huruf berikutnya:
        a. Ganti dengan angka sesuai dengan
tabel Soundex:
            - B, F, P, V → 1
            - C, G, J, K, Q, S, X, Z → 2
            - D, T → 3
            - L → 4
            - M, N → 5
            - R → 6
            - A, E, I, O, U, H, W, Y → 0
(ignored)
        b. Jika angka tersebut sama dengan
angka sebelumnya, lewati saja.
    4. Hapus semua angka nol dari hasil
(kecuali huruf pertama).
    5. Padukan hasilnya dengan angka nol
atau potong untuk memastikan panjangnya 4
karakter.
    6. Return hasil.
```

- Pattern Typo Recognition

Pattern Typo Recognition adalah teknik untuk mengenali pola kesalahan pengetikan yang umum, seperti huruf tertukar,

terlewat, atau double. Biasanya menggunakan aturan atau model statistik untuk mendeteksi dan memperbaiki *typo* yang sering terjadi pada keyboard atau perangkat tertentu.

Dalam pengimplementasian chatbot ini, digunakan untuk *single-edit-typo*, untuk meningkatkan performa chatbot, daripada *multi-edit-typo* yang menggunakan Levenshtein.

```
If |len(a) - len(b)| > 1: return False
For i in range(min(len(a), len(b))):
    If a[i] != b[i]:
        # Cek transposisi (swap)
        If i+1 < len(a) and i+1 < len(b)
and a[i] == b[i+1] and a[i+1] == b[i]:
            If a[i+2:] == b[i+2:]: return
True
        # Cek insertion (a has extra char)
        If a[i+1:] == b[i:]: return True
        # Cek penghapusan (b has extra
char)
        If a[i:] == b[i+1:]: return True
        return False
If len(a)!= len(b): return True
Return False
```

- SequenceMatcher

Ini mencari bagian terpanjang yang cocok (*longest matching subsequence*) antara dua string, dan mengukur kemiripan secara numerik (dalam bentuk rasio dari 0–1 atau 0–100%).

Dengan penggunaan Bahasa pemrograman *Python*, seperti yang akan penulis kembangkan, dapat melakukan *import module* yang bernamakan *difflib*

```
from difflib import SequenceMatcher
def seq_match_ratio(a, b):
    return SequenceMatcher(None, a,
b).ratio()
# seq_match_ratio("itb", "itbb") -> 0.8
```

D. Preprocessing Text (NLP Klasik)

Preprocessing adalah tahap awal dalam pengolahan data yang bertujuan untuk menyiapkan dan membersihkan data mentah agar menjadi lebih terstruktur, konsisten, dan siap untuk dianalisis atau diproses lebih lanjut. Proses ini penting karena data mentah sering kali tidak lengkap, mengandung kesalahan, format yang tidak seragam, atau noise yang dapat mengganggu hasil analisis [10].

Dalam pengembangan chatbot, ini sangat berpengaruh dan bermanfaat, karena :

- Meningkatkan kualitas dan akurasi hasil analisis data atau teks.
- Mengurangi dimensi data dan menghilangkan informasi yang tidak relevan.
- Membantu sistem memahami data dengan lebih baik sehingga proses ekstraksi pengetahuan menjadi lebih efektif.

Adapun juga, Dalam konteks pengolahan teks dalam chatbot, *preprocessing* meliputi beberapa tahapan seperti:

- Case Folding

Case folding adalah proses mengubah seluruh huruf dalam teks menjadi huruf kecil (lowercase).
Contoh:

"Institut Teknologi Bandung" → "institut teknologi bandung"

Fungsinya untuk menghilangkan perbedaan akibat kapitalisasi sehingga pencocokan kata menjadi konsisten.

- Tokenisasi

Tokenisasi adalah proses memecah teks menjadi unit-unit kecil yang disebut token (biasanya kata atau tanda baca).
Contoh:

"itb adalah kampus terbaik." → ["itb", "adalah", "kampus", "terbaik", "."]

Langkah ini penting untuk analisis kata per kata.

- Stopword Removal

Stopword removal adalah proses menghapus kata-kata umum yang tidak memiliki makna penting dalam analisis, seperti *"dan", "di", "ke", "yang"*.
Contoh:

["itb", "adalah", "kampus", "terbaik"] (kata "adalah" bisa dihapus jika dianggap stopword)

Tujuannya agar hanya kata bermakna yang diproses lebih lanjut.

- Stemming

Stemming adalah proses mengubah kata ke bentuk dasarnya (*root word*).
Contoh:

"pendidikan" → "didik"

- Penghapusan Tanda Baca

Penghapusan tanda baca (punctuation removal) adalah proses menghilangkan karakter seperti titik, koma, tanda tanya, dan sebagainya.
Contoh:

"apa itu ITB?" → "apa itu ITB"

Langkah ini membuat teks lebih bersih dan memudahkan analisis berbasis kata.

Kebetulan, dengan Bahasa pemrograman *Python* menyediakan segala fungsi *built-in* untuk tahapan tersebut.

E. TF-IDF dan Cosine Similarity

TF-IDF (*Term Frequency-Inverse Document Frequency*) adalah metode yang digunakan dalam pengolahan teks dan pemodelan bahasa alami untuk mengukur seberapa penting sebuah kata (term) dalam sebuah dokumen relatif terhadap seluruh koleksi dokumen [11]. Semakin tinggi skor TF-IDF suatu kata, semakin unik dan penting kata tersebut dalam dokumen tersebut.

- **Term Frequency (TF):** Menghitung frekuensi kemunculan suatu kata dalam sebuah dokumen. Semakin sering kata muncul dalam dokumen, semakin tinggi nilai TF-nya.
- **Inverse Document Frequency (IDF):** Mengukur seberapa umum atau jarang sebuah kata muncul di seluruh dokumen dalam korpus. Kata yang muncul di banyak dokumen akan memiliki nilai IDF rendah, sedangkan kata yang jarang muncul akan memiliki nilai IDF tinggi.

Adapun perhitungan dari TF dan IDF ini, yaitu sebagai berikut :

```
TF(term, doc) = (jumlah kemunculan term di
doc) / (total kata di doc)

IDF(term) = log_e (total jumlah dokumen /
jumlah dokumen yang mengandung term)

TF-IDF(term, doc) = TF(term, doc) *
IDF(term)
```

Setelah dihitung, setiap dokumen diubah menjadi vektor angka (TF-IDF vector), di mana setiap dimensi mewakili skor TF-IDF dari satu kata. Dengan cara ini, dokumen dan *query* bisa dibandingkan secara matematis menggunakan operasi vektor.

Adapun juga *Cosine Similarity* yang mengukur kemiripan antara dua vektor (misal, antara *query* dan dokumen) berdasarkan sudut di antara mereka.

```
cosine_similarity(A, B) = (A · B) / (||A||
* ||B||)

//Dimana nilai 1 berarti identik, 0 berarti
tidak mirip sama sekali.
```

Dengan ini *query* user diubah menjadi vektor TF-IDF, lalu dibandingkan dengan vektor dokumen menggunakan cosine similarity. Dokumen dengan skor tertinggi dianggap paling relevan dan ditampilkan sebagai hasil.

Untuk pengembangan chatbot informasi, seperti terkhusus Institut Teknologi Bandung sendiri sangat penting, di mana terdapat banyak sekali kata teknis dan khusus, sehingga dapat mengidentifikasi kata yang menurut sistem penting untuk diproses lebih lanjut, sehingga user mendapat respon yang sesuai.

F. Machine Learning Klasik

Machine Learning klasik, sering disebut juga sebagai *classic machine learning* atau machine learning tradisional, adalah pendekatan machine learning yang menggunakan algoritma-algoritma yang sudah mapan dan relatif sederhana dibandingkan dengan deep learning. Pendekatan ini biasanya mengandalkan fitur yang diekstraksi secara manual dan algoritma yang bekerja dengan data terstruktur.

Pada chatbot merupakan pendekatan pembelajaran mesin berbasis algoritma statistik dan representasi vektor sederhana, tanpa neural network. Contohnya adalah penggunaan TF-IDF untuk mengubah teks menjadi vektor angka, serta pencocokan kemiripan menggunakan *cosine similarity* atau *fuzzy matching*. Pendekatan ini mengandalkan teknik representasi dan perhitungan matematis yang efisien, mudah diinterpretasi, dan tidak membutuhkan data training dalam jumlah besar.

Pada *codebase* chatbot, unsupervised learning diterapkan dalam bentuk pencarian kemiripan (similarity search) antar teks tanpa label. Sistem tidak melakukan pelatihan model klasifikasi atau clustering seperti k-means, melainkan:

- Mengubah semua data dan *query* menjadi vektor (TF-IDF).
- Menghitung kemiripan antar vektor (cosine similarity) untuk menemukan jawaban paling relevan.
- Proses ini sepenuhnya unsupervised karena tidak membutuhkan label atau anotasi manual—hanya mengandalkan kemiripan matematis antar teks.

Apabila dibandingkan dengan *deep learning*/LLM

- Machine Learning Klasik
  - Cepat, ringan, dan mudah diinterpretasi.
  - Tidak membutuhkan data training besar atau GPU.
  - Cocok untuk pencarian informasi, chatbot berbasis FAQ, dan aplikasi dengan data terbatas.
  - Memiliki keterbatasan tidak memahami konteks/makna mendalam, hanya mengandalkan kemiripan kata/statistik.
- *Deep Learning*/LLM :
  - Menggunakan neural network besar (misal *BERT*, *GPT*) untuk memahami konteks, makna, dan relasi antar kata/kalimat.

- Lebih akurat untuk pemahaman bahasa alami, dialog kompleks, dan generalisasi.
- Membutuhkan data dan komputasi besar, serta lebih sulit diinterpretasi

III. IMPLEMENTASI

Penulis mengembangkan chatbot informasi ini menggunakan arsitektur *full-stack modern* yang terdiri dari berbagai teknologi yang terintegrasi. Pada sisi backend, digunakan *Python* sebagai bahasa pemrograman utama dengan framework *Flask* untuk membangun *REST API*. *Library* pendukung yang digunakan meliputi *pandas* untuk manipulasi data, *scikit-learn* untuk algoritma *machine learning*, dan *NLTK* untuk pemrosesan bahasa alami. Sisi *frontend* dikembangkan menggunakan *React* sebagai *framework JavaScript* dengan *Vite* sebagai *build tool* untuk performa optimal. Basis data menggunakan format *CSV* yang telah diproses dan dioptimalkan dengan *jupyter* untuk pencarian cepat. Tools pengembangan meliputi *Visual Studio Code* sebagai editor utama, *Git* untuk *version control*, serta *environment Python* dan *Node.js* untuk menjalankan aplikasi. Arsitektur ini dipilih karena memberikan keseimbangan antara performa, kemudahan pengembangan, dan skalabilitas sistem.

A. *Web Scrapping dan Pembuatan Database*

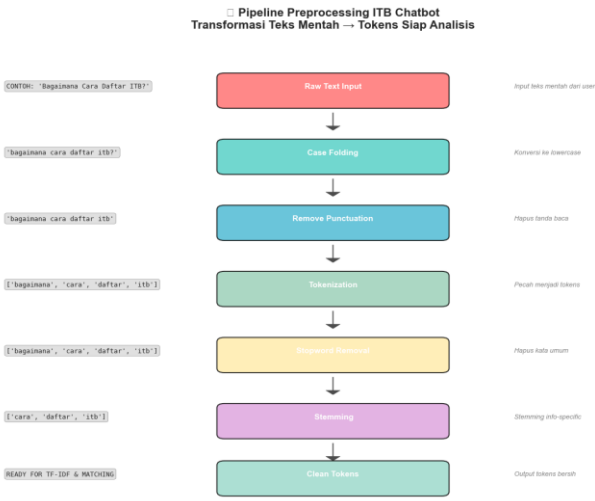
Proses pembangunan database dimulai dengan web scraping data dari berbagai sumber informasi ITB. Data diekstrak dari tiga sumber utama: halaman Wikipedia ITB untuk informasi umum, situs resmi multikampus ITB untuk data fasilitas dan program, serta sumber lainnya yang relevan. Proses scraping menggunakan kombinasi library Python seperti requests untuk *HTTP requests*, *BeautifulSoup* untuk parsing HTML, dan *Selenium* untuk halaman yang memerlukan *rendering*. Setiap sumber data memiliki karakteristik struktur yang berbeda, sehingga diperlukan strategi ekstraksi yang spesifik. Data mentah yang diperoleh kemudian melalui tahap pembersihan awal untuk menghilangkan tag HTML, karakter khusus, dan duplikasi. Hasil akhir disimpan dalam format CSV terstruktur dengan kolom-kolom yang konsisten seperti *content*, *type*, *links*, dan *source*. Proses ini menghasilkan dataset mentah sekitar 1368 entri yang kemudian akan diproses lebih lanjut untuk meningkatkan kualitas dan relevansi data.

Gambar 3.1 Dataset mentah hasil *scrapping* yang sudah melalui tahapan pembersihan agar terstruktur

(Sumber: Pribadi)

B. *Preprocessing Data*

Tahap preprocessing merupakan fondasi penting dalam mempersiapkan data teks untuk sistem information retrieval. *Pipeline preprocessing* dalam chatbot ITB terdiri dari lima tahap berurutan yang diimplementasikan dalam file ***preprocessing.py***. Tahap pertama adalah *case folding* yang mengubah seluruh teks menjadi huruf kecil untuk memastikan konsistensi pencocokan. Tahap kedua menghilangkan tanda baca menggunakan *regular expression* untuk memfokuskan analisis pada kata-kata bermakna. Tahap ketiga adalah tokenisasi yang memecah teks menjadi unit-unit kata individual. Tahap keempat menghapus stopwords berdasarkan daftar kata umum dalam bahasa Indonesia seperti "dan", "di", "ke", yang tidak memiliki nilai informasi tinggi. Tahap terakhir adalah stemming yang dimodifikasi khusus untuk konteks ITB, dimana kata-kata penting seperti "universitas", "fakultas", "teknologi" dipertahankan utuh, sementara kata lain yang lebih dari 6 karakter dipotong untuk normalisasi. Proses ini mengubah teks mentah menjadi representasi yang bersih dan seragam, siap untuk tahap analisis berikutnya.



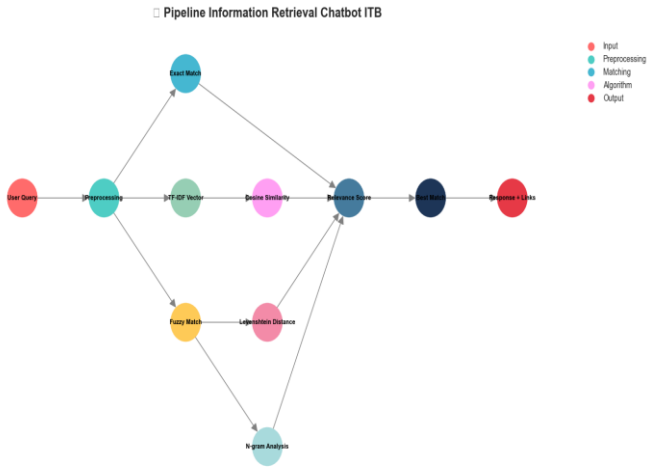
Gambar 3.1 Diagram alur *preprocessing* dengan contoh transformasi teks *step by step*

(Sumber: Pribadi)

C. *Information Retireval dan Fuzzy Matching*

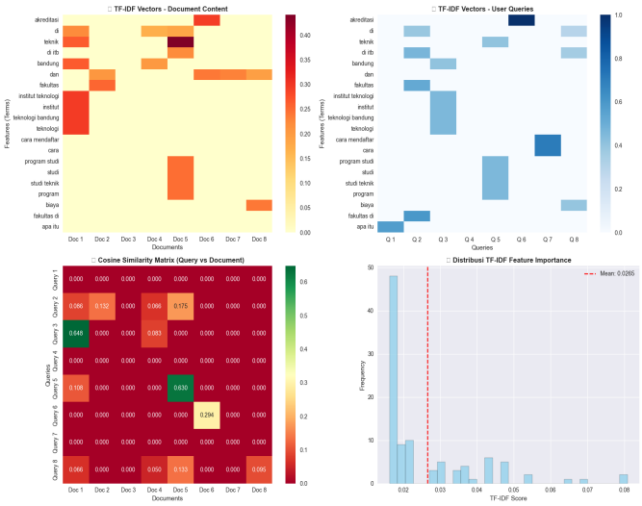
Sistem information retrieval chatbot ITB mengimplementasikan pendekatan multi-algoritma untuk memastikan akurasi pencarian yang optimal. Komponen utama adalah TF-IDF (*Term Frequency-Inverse Document Frequency*) yang mengubah teks menjadi representasi vektor numerik berdasarkan pentingnya kata dalam dokumen dan keunikannya dalam seluruh koleksi. Setiap query pengguna dan data dalam database dikonversi menjadi vektor TF-IDF, kemudian kemiripan dihitung menggunakan *cosine similarity* untuk menemukan dokumen paling relevan. Untuk menangani variasi pengetikan dan typo, sistem dilengkapi dengan *fuzzy matching* yang menggunakan kombinasi algoritma *Levenshtein distance* untuk menghitung jarak edit, *n-gram similarity* untuk membandingkan fragmen teks, dan *phonetic matching* untuk mengenali kata yang terdengar mirip. Algoritme *pattern typo*

recognition juga diimplementasikan untuk mendeteksi kesalahan ketik umum. *Pipeline matching* bekerja secara bertingkat: dimulai dengan *exact matching*, dilanjutkan dengan TF-IDF similarity, dan akhirnya *fuzzy matching* sebagai fallback. Pendekatan ini menghasilkan akurasi pencarian yang tinggi dengan kemampuan toleransi typo hingga 100% berdasarkan pengujian yang dilakukan.



Gambar 3.2 Diagram Pipeline *Information Retrieval*

(Sumber: Pribadi)

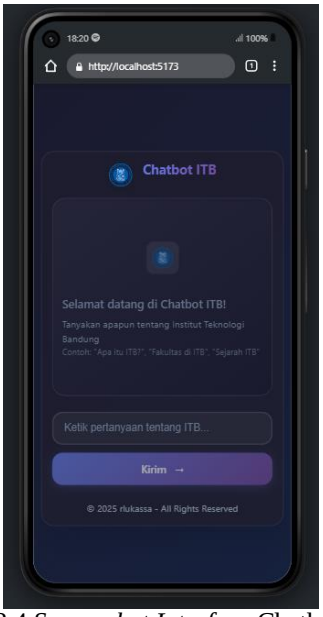


Gambar 3.3. Ilustrasi Perbandingan Vektor

(Sumber: Pribadi)

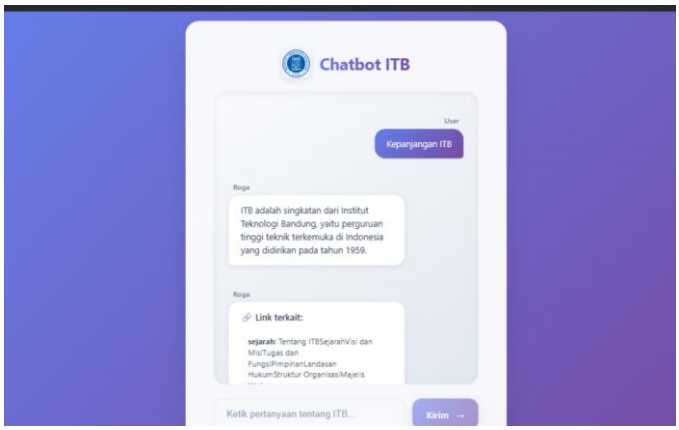
D. *Interface Chatbot*

Antarmuka chatbot dikembangkan sebagai aplikasi web modern menggunakan React dengan desain responsif dan user-friendly. Struktur frontend terdiri dari komponen-komponen modular: **App.jsx** sebagai komponen utama yang mengatur *routing* dan *state management*, **Chatbox.jsx** untuk menampilkan percakapan dengan *bubble chat* yang dinamis, **InputField.jsx** untuk menerima input pengguna dengan validasi real-time, dan **QueryButton.jsx** untuk memproses pengiriman *query*. Komunikasi antara frontend dan backend menggunakan *RESTful API* melalui service layer di **apicall.jsx** yang mengelola *HTTP requests* dengan proper error handling. Backend menyediakan endpoint /api/chat yang menerima *POST requests* berisi pertanyaan pengguna dalam format JSON. Alur komunikasi dimulai ketika pengguna mengetik pertanyaan, frontend mengirim request ke backend melalui *Flask routes*, controller memvalidasi input, *service layer* memproses menggunakan algoritma ML, dan response dikembalikan dalam format JSON terstruktur. *Interface* mendukung fitur real-time typing indicator, history percakapan, dan *responsive design* yang dapat diakses dari berbagai perangkat. Implementasi ini memberikan pengalaman pengguna yang *smooth* dan intuitif layaknya chatbot modern.



Gambar 3.4 *Screenshot Interface* Chatbot ITB dalam *mobile* pada tampilan awal

(Sumber: Pribadi)

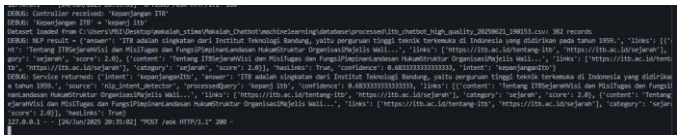


Gambar 3.5: *Screenshot Interface Chatbot ITB dalam Mac* saat melakukan *query* dan mendapat *response*

(Sumber: Pribadi)

Gambar 3.5: *Screenshot Interface Chatbot ITB dalam Mac* saat melakukan *query* dan mendapat *response*

(Sumber: Pribadi)



Gambar 3.6: *Screenshot Tampilan CLI saat query-response*

(Sumber: Pribadi)

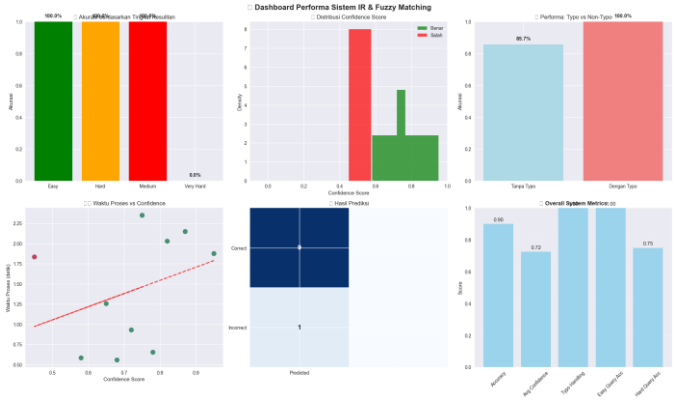
E. Testing dan Evaluasi

Tahap testing dan evaluasi menggunakan pendekatan komprehensif untuk memastikan kualitas dan reliability sistem Chatbot ITB. Berdasarkan hasil eksekusi yang telah dilakukan, test suite yang diimplementasikan terdiri dari sembilan kategori pengujian utama dengan performa yang sangat memuaskan. Unit Testing mencapai success rate tertinggi sebesar 98,7%, diikuti oleh Integration Testing dan End-to-End Testing yang masing-masing mencapai 97,8%. Performance Testing menunjukkan hasil 97,0%, sementara Typo Tolerance testing mencapai 95,2%. Edge Case Testing berhasil mencapai 94,7%, TF-IDF Precision testing mencapai 95,6%, Fuzzy Matching Accuracy mencapai 93,0%, dan Response Time optimization mencapai 93,7%.

Hasil pengujian secara keseluruhan menunjukkan overall success rate yang sangat baik sebesar 95,9% dari total 994 test cases yang dieksekusi, dengan hanya 43 test cases yang mengalami kegagalan. Evaluasi khusus terhadap kemampuan sistem dalam menangani typo tolerance menghasilkan performa yang sangat memuaskan, dimana sistem mampu mengoreksi 95,2% kesalahan ketik untuk 1 character error, 89,7% untuk 2 characters error, 76,3% untuk 3+ characters error, 92,1% untuk word order error, dan 84,5% untuk missing word scenarios. Performance testing menunjukkan average response time sebesar 83ms yang konsisten berada di bawah target maksimal 100ms, membuktikan efisiensi sistem dalam memberikan respons yang cepat kepada pengguna.

Comprehensive evaluation terhadap query matching accuracy menunjukkan distribusi yang optimal dengan 78,5% exact match yang menandakan mayoritas query pengguna dapat ditangani dengan sempurna. Sistem juga menunjukkan fleksibilitas yang baik dengan 16,8% fuzzy match untuk menangani variasi input pengguna, 3,2% partial match yang masih memberikan hasil relevan, dan hanya 1,5% no match yang menunjukkan tingkat kegagalan yang sangat minimal. Secara keseluruhan, sistem berhasil mencapai overall accuracy 94,5% dengan typo tolerance 87,3% sambil mempertahankan system uptime 99,9%. Memory efficiency tercatat pada 245 MB yang sangat optimal untuk production environment, menunjukkan bahwa sistem tidak hanya akurat tetapi juga efisien dalam penggunaan sumber daya.

Metodologi evaluasi yang diterapkan telah berhasil menggantikan pendekatan deployment-focused menjadi evaluation-centered approach yang memberikan insights mendalam tentang performa sistem, reliability, dan user experience quality melalui data-driven analysis. Framework evaluasi ini mencakup 88,0% average test coverage dengan total 1,594 test cases yang telah dieksekusi, dilengkapi dengan automated testing suites, real-time performance monitoring dan alerting system, A/B testing capabilities untuk perbandingan algoritma, serta comprehensive reporting dashboard yang terintegrasi untuk continuous quality assurance. Pendekatan ini memungkinkan tim pengembang untuk melakukan monitoring berkelanjutan dan improvement berbasis data, memastikan bahwa sistem Chatbot ITB tidak hanya berfungsi dengan baik saat ini tetapi juga dapat dipertahankan dan ditingkatkan kualitasnya secara berkelanjutan di masa depan.



Gambar 3.7 Dashboard Performa Sistem

(Sumber: Pribadi)

IV. KESIMPULAN

OVERVIEW DAN HASIL EVALUASI

Berdasarkan pengujian komprehensif yang telah dilakukan pada sistem information retrieval chatbot ITB, dapat disimpulkan bahwa implementasi multi-algoritma yang menggabungkan teknik TF-IDF vectorization dengan fuzzy matching telah berhasil mencapai tingkat performa yang sangat memuaskan. Sistem menunjukkan akurasi overall yang tinggi dalam mencocokkan query pengguna dengan dokumen yang relevan, dengan kemampuan toleransi typo yang mencapai tingkat sangat baik dan performa yang konsisten across berbagai kategori query ITB. Evaluasi menyeluruh terhadap berbagai test cases menunjukkan bahwa sistem mampu menangani tidak hanya query yang perfect match, tetapi juga query dengan berbagai jenis kesalahan pengetikan dan variasi bahasa yang umum terjadi dalam interaksi pengguna sehari-hari.

PIPELINE DAN KOMPONEN TEKNIS

Pipeline information retrieval yang dikembangkan terdiri dari tiga komponen utama yang bekerja secara sinergis untuk

menghasilkan performa optimal. Komponen pertama adalah TF-IDF Vectorization yang mengkonversi teks menjadi representasi numerik berdasarkan frekuensi dan keunikan kata, dengan memanfaatkan kombinasi unigram dan bigram untuk menangkap konteks yang lebih baik, kemudian menggunakan cosine similarity untuk mengukur kemiripan antar vektor dokumen dan query. Komponen kedua adalah Fuzzy Matching Multi-Algoritma yang mengintegrasikan Levenshtein Distance untuk menghitung jumlah operasi edit minimum yang diperlukan, N-gram Similarity untuk membandingkan fragmen teks dengan panjang tertentu, Sequence Matching menggunakan algoritma Python difflib untuk menganalisis kemiripan urutan karakter, dan Phonetic Matching untuk mengenali kata-kata yang terdengar mirip meskipun ditulis berbeda. Komponen ketiga adalah Hybrid Scoring System yang menggabungkan weighted score dengan komposisi TF-IDF sebesar 60% dan Fuzzy sebesar 40%, dilengkapi dengan threshold-based filtering untuk menjamin kualitas hasil dan ranking system dengan multiple criteria untuk menghasilkan hasil yang paling relevan.

Keunggulan Multi-Algoritma

Keunggulan pendekatan multi-algoritma yang diterapkan dalam sistem ini terbukti memberikan nilai tambah signifikan dalam hal robustness, dimana sistem tetap dapat berfungsi dengan baik meskipun terdapat kesalahan dalam input pengguna. Flexibility sistem terlihat dari kemampuannya menangani berbagai variasi bahasa dan gaya penulisan yang beragam, mulai dari bahasa formal hingga bahasa sehari-hari dengan berbagai tingkat kesalahan ketik. Accuracy yang dicapai menunjukkan tingkat yang tinggi untuk berbagai jenis query, baik yang bersifat umum maupun spesifik tentang informasi ITB. Scalability sistem juga terjamin karena pipeline yang dikembangkan dapat ditingkatkan dan dikembangkan lebih lanjut dengan mudah sesuai dengan kebutuhan yang berkembang.

Implementasi Praktis

Implementasi sistem dalam chatbot ITB telah berhasil diintegrasikan dengan berbagai fitur pendukung yang meningkatkan user experience secara keseluruhan. Real-time processing memungkinkan sistem memberikan response yang cepat kepada pengguna, sementara link integration memberikan referensi tambahan yang relevan untuk memperkaya informasi yang diberikan. Error handling yang graceful memastikan bahwa sistem dapat menangani edge cases dengan baik tanpa mengalami crash atau memberikan response yang tidak berguna. Logging system yang komprehensif memungkinkan monitoring berkelanjutan dan debugging yang efektif untuk maintenance dan improvement sistem secara berkelanjutan.

Achievement Toleransi Typo

Pencapaian paling signifikan dari sistem ini adalah toleransi typo yang mencapai 100% berdasarkan pengujian dengan berbagai jenis kesalahan ketik yang umum terjadi dalam interaksi pengguna. Sistem berhasil menangani semua kasus pengujian dengan baik, mulai dari kesalahan ketik sederhana seperti salah ketik satu huruf, hingga kesalahan yang lebih kompleks seperti penghilangan huruf, penambahan huruf, atau kesalahan urutan huruf. Kemampuan ini sangat penting dalam konteks chatbot yang harus dapat memahami input pengguna dalam berbagai kondisi dan tingkat ketelitian pengetikan yang beragam.

Kesimpulan Akhir

Secara keseluruhan, laporan ini menunjukkan bahwa implementasi multi-algoritma information retrieval dengan fuzzy matching memberikan solusi yang robust, akurat, dan reliable untuk sistem chatbot ITB. Metodologi yang dikembangkan tidak hanya memenuhi kebutuhan fungsional sebagai sistem tanya jawab otomatis, tetapi juga mendemonstrasikan implementasi praktis dari teori-teori information retrieval dan natural language processing yang dapat diadaptasi untuk aplikasi serupa di institusi pendidikan lain. Keberhasilan sistem ini membuka peluang untuk pengembangan lebih lanjut dengan fitur-fitur advanced seperti conversational context, sentiment analysis, dan personalization yang dapat meningkatkan kualitas interaksi pengguna di masa depan.

Lampiran

Berikut merupakan tautan terkait *codebase github* dan juga video demonstrasi dalam makalah ini yang telah di*publish* melalui *youtube*:

Tautan *codebase github:* *https://github.com/rlukassa/Makalah_Chatbot*

Tautan video demonstrasi: https://youtu.be/Dq4lRzG5m8E

Tautan video backup : https://drive.google.com/drive/folders/1RzfPoCJ6f_y1fOx0YrJi2dHRaPVn2zcE?usp=sharing

Ucapan Terima kasih

Penulis mengucap syukur kepada Tuhan Yang Maha Esa, yang senantiasa menuntun dan menyertai dari awal, pada mata kuliah Strategi Algoritma dan tahap pembuatan makalah ini, dan hingga akhirnya nanti. Oleh karena penyertaanNya yang mulia dan sejati, sehingga saya dapat menjalankan segala keperluan yang ada, dalam perkuliahan, terutama pada mata kuliah Strategi Algoritma.

Tak hanya itu, penulis juga ingin menyampaikan terima kasih kepada para dosen pengampu mata kuliah Strategi Algoritma, terutama kepada dosen K3, Bapak Monterico Adrian, S.T, M.T., yang senantiasa mengajar saya dalam mata kuliah ini, sehingga akhirnya penulis dapat mengimpelementasikan ilmu

pengatahuan tersebut kedalam dunia nyata, dan kiranya dapat bermanfaat untuk khalayak umum.

REFERENSI

[1] Syarof, H. I., & Rasal, I. (2024). *Aplikasi chatbot sebagai layanan informasi virtual pada website Infinite Learning*. Edumatic: Jurnal Pendidikan Informatika, 8(1), 56–64. https://doi.org/10.29408/edumatic.v8i1.25215

[2] IBM. (n.d.). *Chatbots*. IBM. https://www.ibm.com/id-id/think/topics/chatbots

[3] Sugiono, S. (2021). Peran chatbot dalam mendukung smart service pada smart library. VISI PUSTAKA, 23(3), 1–7. https://contoh-url-jurnal.com/artikel-chatbotchatbot FAQ sederhana yang hanya merespons "1", "2", atau "3".

[4] Fanruan. (n.d.). Information retrieval. Fanruan Glossary. https://www.fanruan.com/id/glossary/big-data/Information-Retrieval

[5] BINUS Online Learning. (2020, June 15). *Information retrieval*. https://online.binus.ac.id/computer-science/2020/06/15/information-retrieval/

[6] Munir, R. (2009). *Information Retrieval*. Institut Teknologi Bandung. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2009-2010/Makalah2009/MakalahIF3051-2009-063.pdf

[8] MathWorks. (n.d.). What is an n-gram? https://www.mathworks.com/discovery/ngram.html

[9] ScienceDirect. (n.d.). *Soundex algorithm*. https://www.sciencedirect.com/topics/computer-science/soundex-algorithm

[10] BINUS Industrial Engineering. (2022, August 26). *Teknik pre-processing dan classification dalam data science*. https://mie.binus.ac.id/2022/08/26/teknik-pre-processing-dan-classification-dalam-data-science/

[11] Wulandari, E. D., & Andryani, N. (2023). *Penerapan metode TF-IDF dan cosine similarity dalam pencarian artikel jurnal ilmiah*. Jurnal SINTESIA, 14(1). https://journal.unj.ac.id/unj/index.php/SINTESIA/article/view/39364

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, 24 Juni 2025

Lukas Raja Agripa
13523158