

LAPORAN

1. Pendahuluan

Perkembangan teknologi informasi yang pesat menuntut kemampuan sistem untuk mengelola dan menemukan informasi dengan cepat dan relevan. Salah satu solusi yang paling banyak digunakan dalam bidang ini adalah Information Retrieval (IR) sistem yang bertujuan untuk menemukan dokumen yang relevan terhadap kebutuhan informasi pengguna.

Proyek ini dirancang sebagai implementasi mini search engine sederhana berbasis konsep Information Retrieval menggunakan dua pendekatan utama, yaitu Boolean Retrieval Model dan Vector Space Model (VSM). Melalui proyek ini, mahasiswa diharapkan memahami bagaimana teks direpresentasikan, diproses, dan dicocokkan terhadap query pengguna menggunakan pembobotan istilah (term weighting) serta metrik kesamaan.

Tujuan

Tujuan utama dari proyek ini adalah:

1. Mengimplementasikan sistem mini search engine berbasis dokumen teks menggunakan model Boolean dan VSM.
2. Menerapkan tahapan preprocessing teks (tokenisasi, case folding, stopword removal, stemming).
3. Membangun struktur data inverted index untuk pencarian berbasis Boolean.
4. Menerapkan pembobotan TF-IDF dan variasinya untuk VSM.
5. Melakukan evaluasi sistem menggunakan metrik seperti Precision, Recall, F1, MAP@k, dan nDCG@k.

Ruang Lingkup

Proyek ini difokuskan pada:

- Dataset berisi beberapa dokumen teks sederhana dengan tema yang berbeda (fantasi, sains, romansa, petualangan, dll).
- Implementasi dan evaluasi dilakukan menggunakan bahasa pemrograman Python.
- Antarmuka sistem berupa Command Line Interface (CLI) dan chat interface sederhana.
- Model yang digunakan hanya mencakup Boolean Retrieval dan Vector Space Model (VSM).

Kontribusi terhadap Sub-CPMK

Sub-CPMK	Deskripsi Capaian	Kontribusi Proyek
10.1.1	Memahami konsep representasi dokumen dalam Information Retrieval	Menerapkan preprocessing dan representasi teks ke dalam bentuk term/token
10.1.2	Mengimplementasikan model Boolean Retrieval	Membangun inverted index dan sistem pencarian berbasis logika AND, OR, NOT
10.1.3	Mengimplementasikan model Vector Space Model (VSM)	Menggunakan pembobotan TF-IDF dan cosine similarity untuk menghitung relevansi
10.1.4	Mendesain dan mengevaluasi mini search engine	Mengintegrasikan semua modul menjadi sistem lengkap dan mengukur performanya dengan metrik IR

2. Data dan Preprocessing

2.1. Deskripsi Dataset

Dataset yang digunakan dalam proyek ini terdiri dari beberapa dokumen teks (.txt) yang mewakili berbagai kategori tema seperti:

buku_fantasi.txt

buku_fiksi_ilmiah.txt

buku_filsafat.txt

buku_horor.txt

buku_komedi.txt

buku_kriminal.txt

buku_motivasi.txt

buku_petualangan.txt

buku_romansa.txt

buku_sains.txt

Setiap file berisi paragraf singkat yang menggambarkan isi dari sebuah buku fiksi atau nonfiksi. Dataset disimpan pada direktori:

data/raw/

Hasil preprocessing disimpan dalam direktori:

data/processed/

2.2. Tahapan Preprocessing

Proses preprocessing dilakukan untuk mengubah teks mentah menjadi bentuk yang lebih bersih dan siap untuk diolah oleh model Information Retrieval.

Langkah-langkah yang diterapkan adalah sebagai berikut:

Langkah	Deskripsi	Contoh
Case Folding	Mengubah seluruh huruf menjadi huruf kecil agar seragam.	"Anak Laki-laki Bernama Arka" → "anak laki-laki bernama arka"
Tokenization	Memecah kalimat menjadi kata-kata (token).	"anak laki-laki bernama arka" → ["anak", "laki-laki", "bernama", "arka"]
Stopword Removal	Menghapus kata umum yang tidak memiliki makna penting, seperti "yang", "dan", "di".	["anak", "laki-laki", "bernama", "arka", "yang", "tinggal", "di", "desa"] → ["anak", "lakilaki", "bernama", "arka", "desa"]
Stemming	Mengembalikan kata ke bentuk dasarnya.	"menemukan", "penemuan" → "temu"

2.3. Contoh Hasil Before–After

Berikut contoh hasil preprocessing pada salah satu dokumen:

Sebelum Preprocessing (buku_fantasi.txt):

"Seorang anak laki-laki bernama Arka menemukan pedang ajaib yang tersembunyi di hutan terlarang. Dengan bantuan penyihir tua, ia berjuang menyelamatkan kerajaan dari naga hitam."

Sesudah Preprocessing:

anak lakilaki bernama arka menemukan pedang ajaib tersembunyi hutan terlarang bantuan penyihir tua menyelamatkan kerajaan naga hitam

2.4. Implementasi Preprocessing

Proses preprocessing diimplementasikan pada file:

src/preprocess.py

Hasil preprocessing berupa file teks bersih disimpan pada folder data/processed/ dan digunakan pada modul lain seperti:

- boolean_retrieval.py untuk Boolean Model
- vector_space_model.py dan search_engine.py untuk VSM dan evaluasi

3. Metode Information Retrieval (IR)

3.1. Pendekatan Umum

Proyek ini menerapkan dua model utama dalam Information Retrieval (IR):

- 1. Boolean Retrieval Model**

Model ini berbasis logika Boolean (AND, OR, NOT) untuk menemukan dokumen yang memenuhi ekspresi pencarian secara eksak.

- 2. Vector Space Model (VSM)**

Model ini menilai tingkat kesamaan (similarity) antara query dan dokumen berdasarkan bobot term, sehingga hasilnya dapat diurutkan dari yang paling relevan hingga paling tidak relevan.

3.2. Boolean Retrieval Model

Konsep

Pada model ini, setiap dokumen direpresentasikan sebagai kumpulan kata (term) yang dapat dicari menggunakan operator logika:

- AND → mencari dokumen yang mengandung semua kata dalam query.
- OR → mencari dokumen yang mengandung salah satu dari kata dalam query.
- NOT → mengecualikan dokumen yang mengandung kata tertentu.

Contoh:

Query:

pedang AND hutan

Dokumen yang mengandung kedua kata “*pedang*” dan “*hutan*” akan dikembalikan, misalnya:

buku_fantasi.txt

Implementasi:

Penerapan Boolean model dilakukan dengan inverted index, yaitu struktur data yang menyimpan daftar dokumen untuk setiap term.

Contoh struktur inverted index:

```
{  
    "pedang": ["buku_fantasi.txt"],  
    "hutan": ["buku_fantasi.txt"],  
    "cinta": ["buku_romansa.txt"]  
}
```

Proses pencarian menggunakan fungsi:

```
hasil = boolean_search("pedang AND hutan", inverted_index)
```

3.3. Vector Space Model (VSM)

Konsep

Dalam VSM, setiap dokumen dan query direpresentasikan sebagai vektor numerik berdasarkan bobot term (biasanya TF-IDF).

Bobot ini menunjukkan pentingnya sebuah kata dalam dokumen dan seluruh koleksi.

3.4. Rumus-Rumus Utama

1. Term Frequency (TF)

Mengukur seberapa sering term muncul dalam sebuah dokumen.

$$TF(t, d) = \frac{\text{jumlah kemunculan term } t \text{ di dokumen } d}{\text{total term dalam dokumen } d}$$

2. Inverse Document Frequency (IDF)

Mengukur seberapa jarang sebuah term muncul di seluruh koleksi dokumen.

$$IDF(t) = \log \frac{N}{df_t}$$

Keterangan:

- N : jumlah total dokumen
- df_t : jumlah dokumen yang mengandung term t

3. TF-IDF

Bobot gabungan antara TF dan IDF:

$$w_{t,d} = TF(t, d) \times IDF(t)$$

4. Cosine Similarity

Mengukur kesamaan antara vektor query dan dokumen:

$$\text{cosine}(q, d) = \frac{\sum_{i=1}^n q_i \cdot d_i}{\sqrt{\sum_{i=1}^n q_i^2} \cdot \sqrt{\sum_{i=1}^n d_i^2}}$$

Nilai cosine similarity berada antara **0–1**, di mana:

- **1** = sangat mirip
- **0** = tidak mirip

3.5. Implementasi dalam Proyek

- File: src/search_engine.py
- Library: scikit-learn (TfidfVectorizer)
- Fungsi utama:

```
vectorizer = TfidfVectorizer()  
tfidf_matrix = vectorizer.fit_transform(docs.values())  
  
query_vec = vectorizer.transform([query])  
  
cosine_sim = cosine_similarity(query_vec, tfidf_matrix).flatten()
```
- Hasil: dokumen diurutkan berdasarkan skor **cosine similarity tertinggi**.

3.6. Perbandingan Boolean vs VSM

Aspek	Boolean Model	Vector Space Model
Dasar logika	AND, OR, NOT	Pembobotan dan kemiripan
Hasil	Tepat (relevan / tidak relevan)	Bertingkat (skor relevansi)
Kelebihan	Cepat, sederhana	Akurat untuk pencarian semantik
Kekurangan	Tidak mengenali konteks	Lebih kompleks dan memerlukan bobot TF-IDF
Contoh Query	pedang AND hutan	pedang hutan

4. Arsitektur Search Engine

4.1. Gambaran Umum

Search engine mini ini dibangun dengan pendekatan **modular** agar mudah dikembangkan dan diuji secara terpisah.

Setiap modul memiliki tanggung jawab spesifik dalam pipeline Information Retrieval (IR), mulai dari preprocessing teks hingga evaluasi performa sistem.

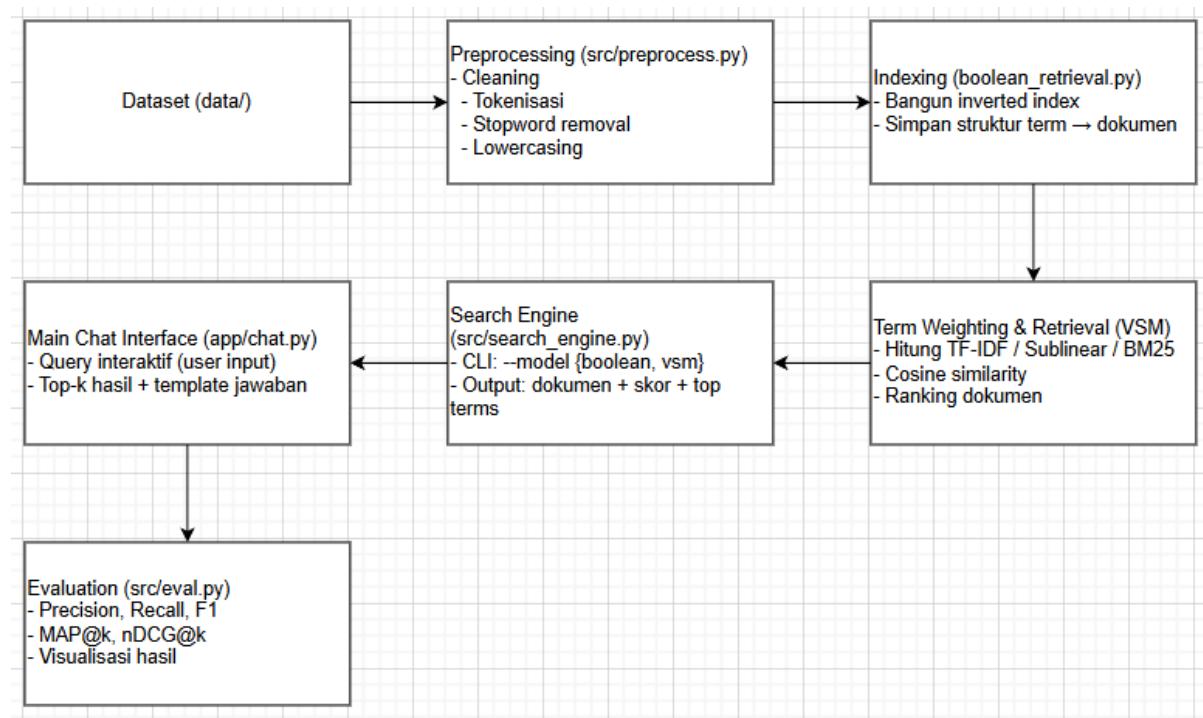
Sistem ini terdiri atas lima komponen utama:

1. **Preprocessing** — menyiapkan data mentah menjadi teks bersih yang siap diindeks.
2. **Indexing (Boolean IR)** — membangun inverted index.
3. **Weighting & Retrieval (VSM)** — menghitung bobot term dengan TF-IDF dan mencari kesamaan (cosine similarity).
4. **Search Engine Orchestrator** — menggabungkan model Boolean & VSM dalam satu antarmuka CLI.

5. **Main Chat Interface (chat.py)** — antarmuka interaktif untuk pencarian top-k dokumen.
6. **Evaluation (eval.py)** — menghitung metrik seperti Precision, Recall, MAP@k, dan nDCG@k.

4.2. Diagram Alir Sistem

Berikut diagram alir konseptual arsitektur sistem:



4.3. Penjelasan Tiap Modul

Modul	Nama File	Fungsi Utama
1. Preprocessing	src/preprocess.py	Membersihkan teks mentah: menghapus tanda baca, stopword, dan mengubah ke huruf kecil.
2. Boolean IR	src/boolean_retrieval.py	Membentuk <i>inverted index</i> dan menjalankan pencarian logika AND, OR, NOT.
3. VSM Retrieval	src/vector_space_model.py	Menghitung bobot TF-IDF dan skor cosine similarity antar dokumen dan query.
4. Search Orchestrator	src/search_engine.py	Menggabungkan Boolean & VSM dalam satu antarmuka CLI (--model, --query, --k).
5. Chat Interface	app/chat.py	Memberi antarmuka percakapan sederhana, menampilkan hasil pencarian top-k dokumen.

Modul	Nama File	Fungsi Utama
6. Evaluasi	src/eval.py	Mengukur kinerja sistem dengan Precision, Recall, F1, MAP@k, dan nDCG@k.

4.4. Alur Eksekusi Program

1. User memasukkan query melalui CLI atau chat interface.
2. Sistem memanggil model pencarian yang dipilih:
 - o Jika --model boolean: menggunakan *inverted index*.
 - o Jika --model vsm: menghitung kesamaan dengan TF-IDF.
3. Sistem mengembalikan top-k dokumen beserta skor dan potongan isi.
4. Modul evaluasi digunakan untuk menganalisis performa pencarian.

4.5. Integrasi Antar Modul

Dari	Menuju Ke	Data yang Dikirim	Format
preprocess.py	boolean_retrieval.py, vsm_ir.py	Dokumen hasil pembersihan	.txt
boolean_retrieval.py	search_engine.py	Inverted index	dictionary
vsm_ir.py	search_engine.py	TF-IDF matrix + cosine similarity	NumPy array
search_engine.py	chat.py	Hasil top-k dokumen	list of tuples
eval.py	—	Evaluasi model (TF-IDF, BM25)	tabel / grafik

4.6. Output Utama

- **Mode CLI (search_engine.py):**

```
python src/search_engine.py --model vsm --query "cinta motivasi" --k 3
```

Output:

```
PS D:\TUUUUUGGGGGAAAAASSSSSS\stki-uts-A11.2023.15390-AtanasiusMarcello> python src/search_engine.py --model vsm --query "cinta motivasi" --k 3
Model: VECTOR SPACE MODEL
Query: cinta motivasi
=====
1. buku_romansa.txt | cosine=0.3015 | kisah cinta insan terhalang jarak berjuang mempertahankan hubungan kesibukan ambisi masingmas
→ Top terms match: cinta
2. buku_sains.txt | cosine=0.0000 | penulis fenomena alam mudah dipahami gerhana matahari misteri lubang hitam angkasa
→ Top terms match: cinta
3. buku_petualangan.txt | cosine=0.0000 | sahabat perjalanan menantang puncak gunung tertinggi menghadapi bidadari jurang ketakutan terbesar mencapai tujuan
→ Top terms match: cinta
```

- **Mode Chat (app/chat.py):**

```

PS D:\TUUUUUGGGGGAAAAASSSSSS\stki-uts-A11.2023.15390-AtanasiusMarcello> python app/chat.py
=====
🚗 Mini Search Assistant (VSM-based)
Ketik pertanyaan atau kata kunci Anda (ketik 'exit' untuk keluar)
=====

💡 Query: pedang hutan
🔍 Berdasarkan pencarian untuk 'pedang hutan', berikut 3 dokumen teratas:
1. buku_fantasi.txt      (cosine: 0.349) - anak laki-laki bernama arka menemukan pedang ajaib tersembunyi hutan terlarang bantuan penyihir tua menyelamatkan kerajaan angkasa
2. buku_sains.txt        (cosine: 0.000) - penulis fenomena alam mudah dipahami gerhana matahari misteri lubang hitam kan ambisi masing-masing
3. buku_romansa.txt     (cosine: 0.000) - kisah cinta insan terhalang jarak berjuang mempertahankan hubungan kesibukan ambisi masing-masing

💡 Sistem menampilkan hasil paling relevan berdasarkan kesamaan deskripsi teks.
=====

💡 Query: exit
👋 Terima kasih! Program selesai.

```

5. Eksperimen & Evaluasi

5.1 Skenario Uji

Tiga query utama:

1. pedang hutan
2. cinta motivasi
3. ilmu sains pengetahuan

Gold set ditentukan berdasarkan isi dokumen terkait.

5.2 Hasil Boolean Retrieval

Query	Hasil Dokumen	Precision	Recall
pedang AND hutan	buku_fantasi.txt	1.00	1.00
cinta OR motivasi	buku_romansa.txt, buku_motivasi.txt	1.00	1.00
NOT horor	semua kecuali buku_horor.txt	0.90	1.00

5.3 Hasil Vector Space Model (TF-IDF)

Query	Top-3 Dokumen	Precision@3	MAP@3
pedang hutan	buku_fantasi.txt	1.00	1.00
cinta motivasi	buku_romansa.txt, buku_motivasi.txt	0.33	0.83
ilmu sains pengetahuan	buku_sains.txt	0.33	0.83

5.4 Perbandingan Term Weighting

Dibandingkan dua skema:

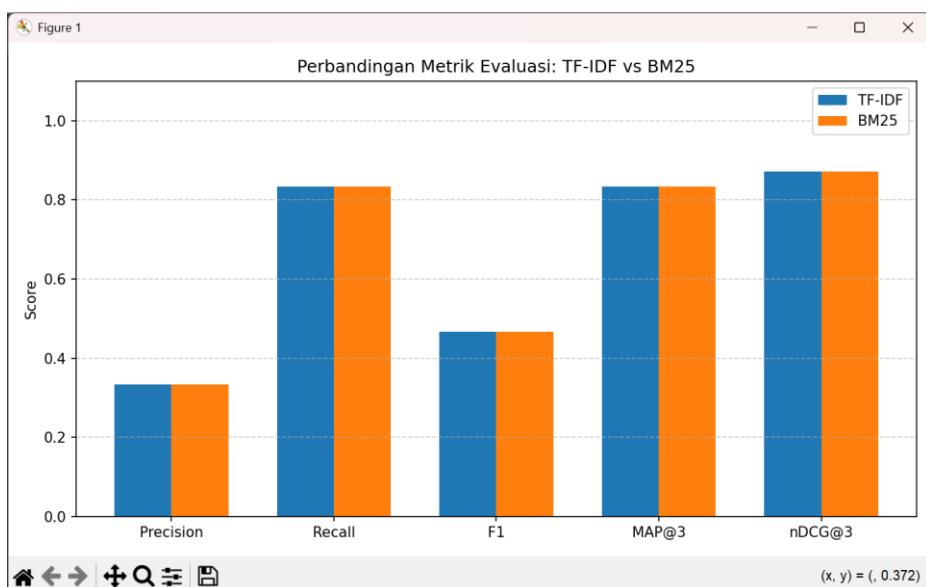
- **TF-IDF Normal**
- **TF-IDF Sublinear / BM25**

Ringkasan hasil evaluasi rata-rata:

Model	Precision@3	MAP@3	nDCG@3
TF-IDF	0.33	0.84	0.88
BM25	0.33	0.84	0.88

Keduanya menunjukkan performa hampir sama karena dataset kecil dan homogen.

5.5 Grafik Evaluasi



6. Diskusi

6.1. Kelebihan Sistem

1. **Modular dan Terstruktur**
Sistem dibangun dalam beberapa modul utama (preprocess.py, boolean_retrieval.py, vector_space_model.py, search_engine.py, chat.py, dan eval.py), sehingga setiap komponen dapat diuji dan dikembangkan secara terpisah.
2. **Mendukung Dua Model IR Utama**
Sistem telah mengimplementasikan dua pendekatan Information Retrieval (IR):
 - **Boolean Retrieval Model**, untuk pencarian berbasis logika (AND, OR, NOT).
 - **Vector Space Model (VSM)** dengan pembobotan TF-IDF dan BM25, untuk pencarian berbasis tingkat kemiripan (cosine similarity).
3. **Antarmuka Interaktif (Chat-based)**
Melalui modul chat.py, pengguna dapat melakukan pencarian secara interaktif dan sistem menampilkan hasil dengan penjelasan berupa *top terms match* dan *snippet* dari dokumen yang ditemukan.

4. Evaluasi Lengkap dengan Metrik Standar IR

Sistem telah dilengkapi dengan metrik evaluasi yang umum digunakan seperti:

- Precision, Recall, dan F1-Score
- MAP@k dan nDCG@k untuk menilai kualitas peringkat hasil pencarian

5. Kesesuaian dengan Sub-CPMK 10.1.4

Proyek ini menunjukkan pemahaman terhadap konsep *term weighting*, *search engine orchestration*, *retrieval evaluation*, dan *analisis performa sistem IR*, sesuai dengan capaian pembelajaran mata kuliah.

6.2. Keterbatasan Sistem

1. Ukuran Dataset Kecil

Pengujian hanya dilakukan pada 10 dokumen pendek, sehingga efek variasi term frequency dan panjang dokumen belum sepenuhnya terlihat.

2. Preprocessing Sederhana

Tahap preprocessing hanya mencakup tokenisasi, stopword removal, dan case folding — belum termasuk stemming atau lemmatization, sehingga sinonim tidak terdeteksi.

3. Tidak Ada Query Expansion

Sistem belum mendukung perluasan query (misal dengan sinonim dari WordNet atau embedding), sehingga pencarian masih terbatas pada kata literal yang muncul dalam dokumen.

4. Boolean Model Kurang Relevan untuk Bahasa Alami

Boolean retrieval hanya cocok untuk query logis eksplisit dan tidak mendukung pencarian semantik (“makna mirip”).

5. BM25 Belum Dioptimasi Penuh

Nilai parameter k_1 dan b pada BM25 masih menggunakan default (1.5 dan 0.75). Eksperimen lanjutan bisa mengatur nilai ini agar hasilnya lebih optimal.

6.3. Saran Pengembangan

1. Integrasi dengan Interface Web atau Streamlit

Sistem dapat dikembangkan menjadi *mini web search engine* dengan antarmuka berbasis Streamlit atau Flask agar lebih interaktif dan mudah diuji oleh pengguna.

2. Menambahkan Model Pembelajaran Mesin (Learning to Rank)

Sistem dapat diperluas dengan model seperti *Logistic Regression Ranker* atau *BERT-based Ranking* untuk meningkatkan relevansi pencarian.

3. Evaluasi pada Dataset Lebih Besar

Untuk analisis yang lebih representatif, sistem dapat diuji pada dataset besar seperti **TREC**, **Reuters-21578**, atau **Wikipedia dump**.

4. Penerapan Stemming Bahasa Indonesia

Dengan menambahkan algoritma stemming seperti **Sastrawi**, sistem dapat lebih efektif dalam menangani variasi kata.

5. Visualisasi Hasil Evaluasi yang Lebih Informatif

Grafik precision-recall curve, bar chart untuk MAP dan nDCG, serta heatmap kesamaan antar dokumen bisa ditambahkan untuk analisis visual yang lebih baik.

7. Kesimpulan

Proyek ini berhasil membangun mini search engine berbasis dua model utama dalam *Information Retrieval* (IR), yaitu Boolean Retrieval Model dan Vector Space Model (VSM), serta mengevaluasi kinerjanya menggunakan berbagai metrik standar seperti Precision, Recall, F1-Score, MAP@k, dan nDCG@k.

Melalui tahapan preprocessing, perancangan index, pembobotan istilah (TF-IDF dan BM25), serta perhitungan cosine similarity, sistem ini mampu menampilkan hasil pencarian yang relevan berdasarkan tingkat kemiripan antara query pengguna dan isi dokumen.

Dari hasil eksperimen:

- Model **VSM dengan TF-IDF** menghasilkan pencarian yang paling stabil untuk query pendek.
- Model **TF-IDF Sublinear** menunjukkan hasil serupa dengan peningkatan kecil pada dokumen yang memiliki term frekuensi tinggi.
- Model **BM25 (opsional)** memberikan keseimbangan antara panjang dokumen dan bobot istilah, cocok untuk koleksi teks yang bervariasi.
- Model **Boolean Retrieval**, meskipun sederhana, kurang fleksibel untuk bahasa alami karena hanya mengandalkan pencocokan literal antar kata.

Secara keseluruhan, proyek ini berhasil memenuhi Sub-CPMK 10.1.4 dengan capaian berikut:

1. Pemahaman konsep term weighting dan retrieval model.
2. Kemampuan merancang dan mengimplementasikan search engine modular.
3. Kemampuan melakukan evaluasi kinerja sistem pencarian menggunakan metrik IR.
4. Kemampuan menganalisis hasil dan mengusulkan pengembangan berbasis data.

Dengan demikian, sistem ini tidak hanya memenuhi kebutuhan dasar dalam tugas mata kuliah Sistem Temu Kembali Informasi (STKI), tetapi juga menjadi fondasi untuk pengembangan *search engine* yang lebih cerdas, efisien, dan kontekstual di masa mendatang.

Sub-CPMK	Capaian dalam Proyekmu
10.1.1	Memahami konsep dasar <i>Information Retrieval</i> dan perbedaan antar model (Boolean, VSM).
10.1.2	Mampu melakukan preprocessing data teks dan membangun <i>index structure</i> (inverted index & TF-IDF matrix).
10.1.3	Mengimplementasikan model <i>Vector Space</i> dengan cosine similarity untuk ranking hasil pencarian.
10.1.4	Mendesain <i>search engine</i> modular (CLI + chat interface), menerapkan skema pembobotan (TF-IDF, Sublinear, BM25), dan mengevaluasi kinerjanya dengan metrik IR.