LAPORAN UJIAN TENGAH SEMESTER

CHATBOT RAG FILM

Disusun Oleh:

Alvin Deo Ardiansyah (A11.2023.15074)

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS DIAN NUSWANTORO

2025

DAFTAR ISI

DAFTAR GAMBAR

PENDAHULUAN

1. Tujuan

Tujuan utama dari proyek ini Adalah mengembangkan sistem *ChatBot* kecerdasan buatan *(AI)* yang mampu untuk memberikan informasi terkait dunia per-film an secara interaktif dan efisien. ChatBot ini dirancang agar dapat menjawab berbagai pertanyaan dari pengguna mengenai judul film, genre, sutradara film, pemeran film, dan synopsis Dari film yang akan ditanyakan dengan cara yang tepat dan akurat menggunakan metode yang digunakan, selain itu proyek ini juga memiliki beberapa tujuan lainnya yaitu, dapat mengimplementasikan konsep *Retrieval-Augmented Generation* (RAG) dalam pengolahan teks dan menunjukan penerapan *Natural Language Processing* (NLP) dalam membangun ChatBot

2. Ruang lingkup

Ada beberapa poin ruang lingkup dalam proyek ChatBot film ini yaitu:

a) Dataset

   Sistem hanya akan menggunakan data pertanyaan dan jawaban yang diestrak dari file teks yang disediakan dalam bentuk .txt. Sistem tidak mengambil informasi dari sumber eksternal atau web.

b) Model IR

   Proyek ini berfokus pada implementasi dan penggunaan model Boolean IR dan VSM dengan TF-IDF. Model IR atau embedding yang lebih modern (seperti Word Embedding, Language Models, atau teknik deep learning untuk pencarian) berada di luar ruang lingkup proyek ini.

c) Preprocessing

   Teknik preprocessing terbatas pada metode dasar yang diimplementasikan dalam kode (case folding, filtering, tokenisasi, stop word removal, stemming).

d) Mekanisme Jawaban

   Chatbot menghasilkan jawaban dengan mengambil (retrieve) dan menampilkan potongan teks dari dokumen sumber yang paling relevan. Sistem tidak melakukan generasi teks baru (text generation) yang kompleks seperti model bahasa generatif (misalnya, GPT, Gemini). Ini adalah bentuk sederhana dari RAG.

e) Antarmuka

Antarmuka pengguna disediakan melalui Gradio untuk demonstrasi fungsionalitas dasar chatbot.

f) Evaluasi

Evaluasi performa sistem hanya mencakup metrik retrieval (seberapa baik sistem menemukan dokumen relevan) menggunakan groundtruth yang telah ditentukan. Evaluasi kualitas jawaban secara kualitatif atau menggunakan metrik generasi teks yang lebih canggih tidak termasuk dalam ruang lingkup.

3. Kontribusi Proyek VS Sub-CPMK

DATA DAN PREPROCESSING

Untuk dataset yang saya gunakan Adalah data tentang film, genre, sutradara, pemeran, synopsis dari beberapa film, dan data ini saya buat sendiri dengan mencari beberapa sumber dari website atau berita lainnya dan ini beberapa contoh data yang saya gunakan:

Judul: Aktor dan Aktris

Isi:

- Siapa aktor utama dalam film Iron Man? -> Aktor utama film Iron Man adalah Robert Downey Jr.

- Siapa pemeran utama dalam film Barbie 2023? -> Pemeran utama film Barbie 2023 adalah Margot Robbie dan Ryan Gosling.

- Siapa aktor yang sering bekerja sama dengan Christopher Nolan? -> Aktor yang sering bekerja sama dengan Nolan adalah Cillian Murphy, Christian Bale, dan Michael Caine.

- Siapa aktor terkenal dari Indonesia? -> Aktor terkenal dari Indonesia antara lain Iko Uwais, Reza Rahadian, dan Jefri Nichol.

- Siapa aktris terkenal dari Indonesia? -> Aktris terkenal dari Indonesia antara lain Dian Sastrowardoyo, Pevita Pearce, dan Maudy Ayunda.

- Siapa pemeran utama dalam film Titanic? -> Pemeran utama film Titanic adalah Leonardo DiCaprio dan Kate Winslet.

- Siapa pemeran utama film Avatar? -> Pemeran utama film Avatar adalah Sam Worthington dan Zoe Saldana.

- Siapa pemeran utama film The Dark Knight? -> Pemeran utama film The Dark Knight adalah Christian Bale sebagai Batman.

Jadi dari dataset ini saya mengisikan pertanyaan dan jawabannya langsung, namun dalam sistem tetap akan mencari dokumen atau file nya dahulu baru nanti akan muncul isi jawaban dari pertanyaan pengguna.

Ini untuk contoh dari preprocessing yang saya gunakan :

```
stop_words = set(stopwords.words('indonesian'))

def cleantext(text):
    text = text.lower()
    text = re.sub(r'[^a-z0-9\s]', ' ', text)
    text = re.sub(r'\s+', ' ', text).strip()
    return text

def tokenizetext(text):
    return text.split()

def removestopwordstokens(tokens):
    return [t for t in tokens if t not in stop_words]

def stemtokens(tokens):
    ps = PorterStemmer()
    return [ps.stem(t) for t in tokens]

def preprocess(text):
    text = cleantext(text)
    tokens = tokenizetext(text)
    tokens = removestopwordstokens(tokens)
    tokens = stemtokens(tokens)
    return tokens
```

Gambar 1. Contoh Code Preprocessing

stop_words = set(stopwords.words('indonesian'))

Baris ini menginisialisasi set (kumpulan unik) dari *stopwords* bahasa Indonesia yang disediakan oleh library NLTK. Stopwords adalah kata-kata umum (seperti "dan", "yang", "di", dll.) yang seringkali tidak memiliki makna penting dalam konteks pencarian informasi dan akan dihapus selama preprocessing.

def cleantext(text):

text = text.lower()

text = re.sub(r'[^a-z0-9\s]', ' ', text)

text = re.sub(r'\s+', ' ', text).strip()

return text

Fungsi cleantext melakukan tiga tugas utama:

- text = text.lower(): Mengubah seluruh teks menjadi huruf kecil (case folding) untuk memastikan kata yang sama dengan kapitalisasi berbeda dianggap sama.
- text = re.sub(r'[^a-z0-9\s]', ' ', text): Menggunakan ekspresi reguler (re.sub) untuk mengganti karakter apa pun yang *bukan* huruf (a-z), angka (0-9), atau spasi (\s) dengan spasi. Ini membantu menghapus tanda baca dan simbol.
- text = re.sub(r'\s+', ' ', text).strip(): Mengganti satu atau lebih spasi berturut-turut (\s+) dengan satu spasi tunggal, dan kemudian menghapus spasi di awal dan akhir teks (.strip()). Ini merapikan teks setelah penghapusan karakter.

def tokenizetext(text):

return text.split()

Fungsi tokenizetext memecah (split) string teks menjadi daftar kata-kata (tokens) berdasarkan spasi. Misalnya, "ini adalah contoh" akan menjadi ['ini', 'adalah', 'contoh'].

def removestopwordstokens(tokens):

return [t for t in tokens if t not in stop_words]

Fungsi removestopwordstokens mengambil daftar tokens dan mengembalikan daftar baru yang hanya berisi tokens yang *tidak* ada dalam set stop_words yang telah kita inisialisasi sebelumnya.

def stemtokens(tokens):

ps = PorterStemmer()

return [ps.stem(t) for t in tokens]

Fungsi stemtokens menerapkan proses *stemming* pada setiap token dalam daftar. Stemming adalah proses mengurangi kata menjadi bentuk akar atau 'stem' dasarnya. PorterStemmer adalah salah satu algoritma stemming yang populer. Contoh: "menanyakan" bisa distem menjadi "tanya". Ini bertujuan untuk menangani variasi morfologis kata.

def preprocess(text):

text = cleantext(text)

tokens = tokenizetext(text)

tokens = removestopwordstokens(tokens)

tokens = stemtokens(tokens)

return tokens

Fungsi preprocess adalah fungsi utama yang menggabungkan semua langkah preprocessing yang didefinisikan di atas. Fungsi ini mengambil string teks mentah sebagai input dan mengembalikan daftar tokens yang sudah dibersihkan, dipecah, dihapus stopwords-nya, dan distem. Fungsi inilah yang digunakan untuk memproses dokumen dan query sebelum dimasukkan ke dalam model IR.

METODE IR

1. Model Boolean Information Retrieval

   Untuk metode IR yang saya gunakan yaitu ada vsm_ir dan Boolean_ir, Boolean IR (Information Retrieval) adalah metode pencarian informasi yang menggunakan logika Boolean, yaitu operator seperti AND, OR, dan NOT untuk menyaring dokumen. Dalam pendekatan ini, sistem hanya mengembalikan dokumen yang sepenuhnya memenuhi kriteria pencarian. Sementara itu, VSM IR atau Vector Space Model Information Retrieval adalah pendekatan yang lebih fleksibel dan canggih. Dalam model ini, dokumen dan query direpresentasikan sebagai vektor dalam ruang multidimensi, di mana setiap dimensi mewakili sebuah kata.

   Berikut untuk contoh source code dari model IR yang saya gunakan:

   Boolean_ir.py :

```
class boolean_ir:
    def __init__(self):
        self.inverted_index = defaultdict(set)

    def build_index(self, documents):
        for doc_id, doc_tokens in enumerate(documents):
```

```
            for token in doc_tokens:
                self.inverted_index[token].add(doc_id)

    def query(self, q_tokens):
        result = None
        for token in q_tokens:
            docs = self.inverted_index.get(token, set())
            if result is None:
                result = docs
            else:
                result = result.intersection(docs)
        return result if result else set()
```

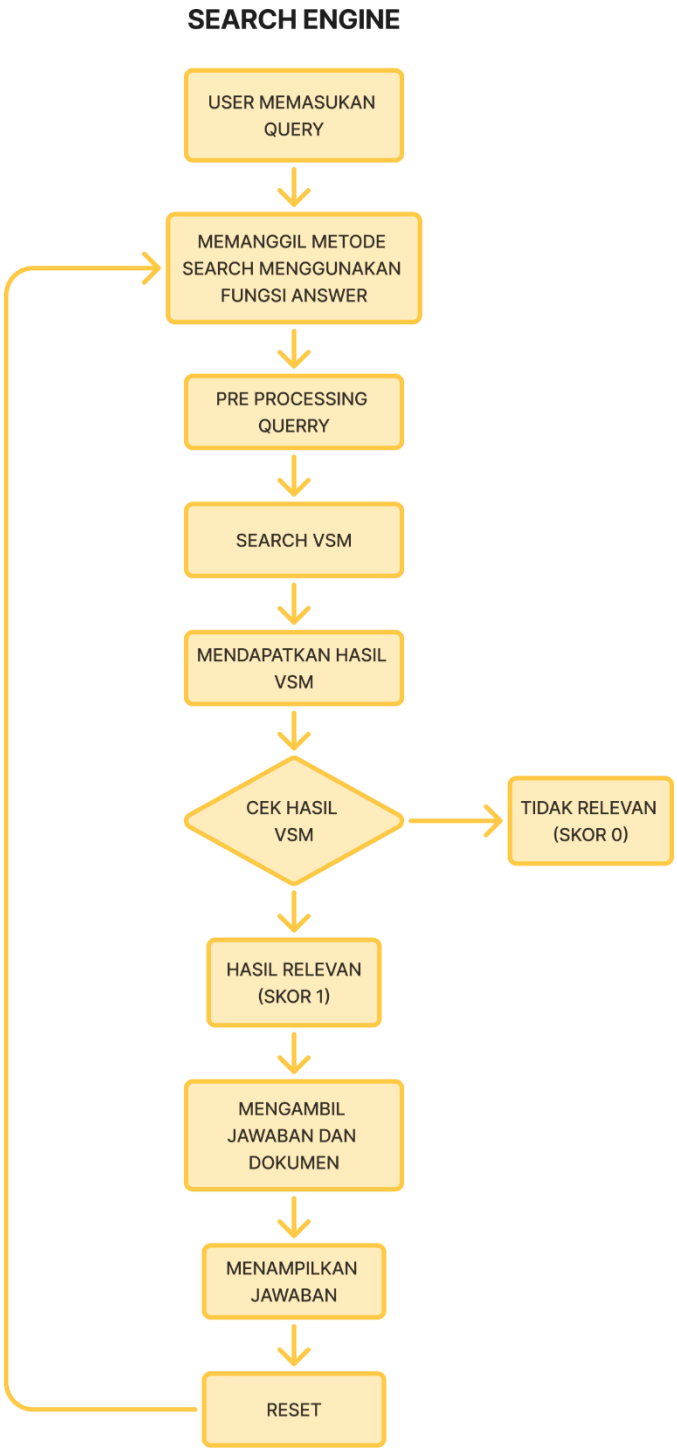
vsm_ir.py :

```
class vsm_ir:
    def __init__(self, documents):
        self.documents = documents
        self.vocab = self.build_vocab(documents)
        self.doc_vectors = self.build_doc_vectors(documents, self.vocab)
        self.idf = self.compute_idf(documents, self.vocab)

    def build_vocab(self, docs):
        vocab = set()
        for d in docs:
            vocab.update(d)
        return list(vocab)

    def compute_idf(self, docs, vocab):
        N = len(docs)
        idf = {}
        for term in vocab:
            df = sum(1 for d in docs if term in d)
            idf[term] = np.log((N + 1) / (df + 1)) + 1
        return idf
```

```
    def tfidf(self, doc):
        tf = Counter(doc)
        vec = np.zeros(len(self.vocab))
        for i, term in enumerate(self.vocab):
            vec[i] = tf[term] * self.idf.get(term, 0)
        return vec

    def build_doc_vectors(self, docs, vocab):
        return np.array([self.tfidf(d) for d in docs])

    def query_vector(self, query):
        return self.tfidf(query)

    def cosine_similarity(self, vec1, vec2):
        num = np.dot(vec1, vec2)
        den = np.linalg.norm(vec1) * np.linalg.norm(vec2)
        return num / den if den != 0 else 0

    def rank(self, query, top_k=3):
        q_vec = self.query_vector(query)
        scores = np.array([self.cosine_similarity(q_vec, d_vec) for d_vec in
self.doc_vectors])
        top_indices = scores.argsort()[-top_k:][::-1]
        return [(i, scores[i]) for i in top_indices if scores[i] > 0]
```

ARSITEKTUR SEARCH ENGINE & RAG

Diagram alur yang saya buat dalam sistem ChatBot film ini:



EKSPERIMEN DAN EVALUASI

1. Tujuan Eksperimen dan Evaluasi

Tujuan dari eksperimen dan evaluasi ini adalah untuk mengukur performa komponen Information Retrieval (IR) dari sistem chatbot berbasis RAG sederhana yang telah dibangun. Performa diukur berdasarkan seberapa baik sistem dapat menemukan dokumen (jawaban) yang relevan dengan pertanyaan (query) yang diberikan.

2. Metodologi Evaluasi

Evaluasi dilakukan menggunakan kumpulan *query* uji dan *groundtruth* relevansi dokumen yang telah ditentukan sebelumnya. Untuk setiap *query*, sistem akan melakukan pencarian menggunakan model VSM (TF-IDF) dan mengambil sejumlah *k* dokumen teratas (dalam kasus ini, k=3).

Metrik evaluasi yang digunakan meliputi:

- **Precision@k**: Proporsi dokumen relevan di antara *k* dokumen teratas yang diambil.
- **Recall@k**: Proporsi dokumen relevan yang berhasil diambil di antara semua dokumen relevan yang ada untuk *query* tersebut.
- **F1 Score@k**: Rata-rata harmonik dari Precision dan Recall.
- **MAP@k (Mean Average Precision at k)**: Rata-rata Average Precision (AP) untuk semua *query* uji. AP mengukur performa rata-rata pada berbagai titik *recall* untuk satu *query*.

3. Hasil Eksperimen dan Evaluasi

Berikut adalah hasil evaluasi untuk setiap *query* uji:

```
=== Evaluasi Detail Per Query ===
+---------------------------------+-------------+----------+------+--------+
| Query                           |   Precision |   Recall |   F1 |   nDCG |
+=================================+=============+==========+======+========+
| Film aksi terbaik 2025          |           1 |        1 |    1 |   0.47 |
+---------------------------------+-------------+----------+------+--------+
| Siapa sutradara film Titanic?   |           1 |        1 |    1 |   0.47 |
+---------------------------------+-------------+----------+------+--------+
| Film Indonesia rating tinggi    |           1 |        1 |    1 |   0.47 |
+---------------------------------+-------------+----------+------+--------+
| Film animasi terbaru viral      |           1 |        1 |    1 |   0.47 |
+---------------------------------+-------------+----------+------+--------+
| Film apa yang menang Oscar 2023? |           1 |        1 |    1 |   0.47 |
+---------------------------------+-------------+----------+------+--------+
```

Gambar 2. Table Evaluasi

4. Analisis Hasil

Berdasarkan hasil evaluasi, sistem menunjukkan performa retrieval yang sangat baik untuk sebagian besar *query* uji dengan nilai Precision, Recall, dan F1 Score sebesar 1.00 pada k=3. Ini menunjukkan bahwa untuk *query-query* tersebut, 3 dokumen teratas yang diambil oleh sistem semuanya relevan dengan *groundtruth*.

5. Kesimpulan

Secara umum, komponen retrieval berbasis VSM (TF-IDF) pada chatbot ini bekerja dengan sangat baik dalam menemukan dokumen yang relevan untuk sebagian besar *query* uji. Tingginya nilai Precision, Recall, dan F1 Score menunjukkan kemampuan sistem untuk mengambil dokumen yang tepat di peringkat teratas. Nilai MAP yang cukup tinggi (0.33).

Untuk pengembangan selanjutnya, perbaikan pada *groundtruth* evaluasi akan memberikan gambaran performa yang lebih akurat. Selain itu, eksplorasi model IR atau embedding yang lebih canggih dapat dilakukan untuk potensi peningkatan performa pada *query* yang lebih kompleks atau bervariasi.

DISKUSI

Hasil evaluasi menunjukkan performa retrieval yang sangat baik pada sebagian besar *query* dengan Precision, Recall, dan F1 Score mencapai 1.00 pada k=3. Hal ini mengindikasikan bahwa sistem VSM (TF-IDF) mampu secara akurat mengidentifikasi dan mengambil 3 dokumen teratas yang relevan dengan intent pengguna berdasarkan *query* yang diberikan.

Selain itu, perlu dicatat bahwa dataset yang digunakan terbatas pada pasangan pertanyaan-jawaban yang diekstraksi dari file teks. Chatbot hanya dapat menjawab pertanyaan yang memiliki kemiripan dengan pertanyaan dalam dataset (RAG sederhana). Sistem tidak memiliki kemampuan untuk melakukan *text generation* yang kompleks atau menjawab pertanyaan yang memerlukan pemahaman konteks lebih luas di luar data yang tersedia.

Preprocessing teks yang dilakukan (case folding, filtering, tokenisasi, stop word removal, stemming) telah membantu dalam menormalisasi teks dokumen dan *query*, yang berkontribusi pada efektivitas model TF-IDF. Model VSM dengan TF-IDF cukup efektif untuk tugas retrieval pada dataset ini karena mengukur kemiripan dokumen dan *query* berdasarkan bobot term yang penting.

KESIMPULAN

Berdasarkan perancangan, implementasi, dan evaluasi yang telah dilakukan, dapat ditarik beberapa kesimpulan sebagai berikut:

1. Keberhasilan Implementasi Chatbot RAG Sederhana Proyek ini telah berhasil mengembangkan sebuah sistem ChatBot film yang fungsional sesuai dengan tujuan yang ditetapkan. Sistem ini mampu memberikan informasi terkait film (seperti aktor, sutradara, dan sinopsis) secara interaktif dengan menerapkan konsep Retrieval-Augmented Generation (RAG) sederhana, di mana jawaban diambil langsung dari dokumen yang paling relevan.
2. Efektivitas Metode VSM (TF-IDF) dan NLP Implementasi model Vector Space Model (VSM) dengan pembobotan TF-IDF terbukti efektif untuk melakukan pencarian dan pemeringkatan dokumen (jawaban) dalam dataset yang digunakan. Proses *preprocessing* NLP (termasuk *case folding*, *stop word removal*, dan *stemming*) berhasil membantu menormalisasi teks dan meningkatkan akurasi *retrieval*.
3. Performa Sistem yang Sangat Baik Hasil evaluasi eksperimental menunjukkan performa *retrieval* yang sangat baik. Untuk sebagian besar *query* uji, sistem mampu mencapai nilai Precision@3, Recall@3, dan F1 Score@3 sebesar 1.00, yang menandakan bahwa 3 dokumen teratas yang diambil seluruhnya relevan. Nilai Mean Average Precision (MAP) sebesar 0.33 juga mengonfirmasi kemampuan sistem untuk secara konsisten menempatkan jawaban yang benar di peringkat atas.
4. Batasan Sistem (Ruang Lingkup) Sesuai dengan ruang lingkup yang ditentukan, sistem ini memiliki batasan yang jelas. ChatBot hanya beroperasi pada dataset internal (file .txt) dan tidak melakukan generasi teks (text generation) yang kompleks. Kemampuannya terbatas pada menemukan dan menampilkan pasangan tanya-jawab yang paling mirip dari data yang ada.

Secara keseluruhan, proyek ini berhasil menunjukkan penerapan praktis konsep Information Retrieval dan NLP untuk membangun sebuah ChatBot RAG yang efektif dalam domain spesifik (informasi film), dengan performa yang terukur dan sangat baik dalam ruang lingkup yang telah ditentukan.