PERANCANGAN APLIKASI *SUMMARIZE TOOLS* DENGAN *NATURAL LANGUAGE PROCESS*

Dosen Pengampu:

Yuyun Umaidah, S.Kom., M.Kom.

Disusun Oleh:

Bintang Danuarta

NPM: 2210631170014

INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS SINGAPERBANGSA KARAWANG

2024

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Pendahuluan

Kecerdasan Buatan atau *Artificial Intelligence* (AI) adalah bidang ilmu komputer yang dalam fungsi dan tujuannya untuk membantu kegiatan dan pekerjaan sehari-hari manusia. Kegunaan AI semakin meluas seiring pertumbuhan jumlah penggunanya. Banyak kegiatan umum manusia yang mulai digantikan AI. Banyak bidang-bidang AI yang dipakai sesuai kebutuhan dan kegiatan, salah satunya adalah *Natural Language Processing* (NLP).

Dalam pembuatan dokumentasi penelitian dalam artikel, dan laporan-laporan, perlu dibuatnya kesimpulan untuk meringkas hasil pembahasan suatu penelitian. Perlu adanya alat untuk dengan mudah meringkas suatu kesimpulan penelitian. Penggunaan AI dalam pembuatan *summarize tools* diharapkan dapat memudahkan suatu tahapan dalam pembuatan kesimpulan.

BAB 2
LANDASAN TEORI

2.1 *Natural Language Processing* (NLP)

Natural Language Processing (NLP) adalah cabang dari AI yang berfokus pada interaksi antara komputer dan manusia melalui bahasa alami. NLP bertujuan untuk memungkinkan komputer memahami, menafsirkan, dan menghasilkan bahasa manusia secara efektif.

2.2 Peringkasan Teks Abstraktif

Pemrosesan teks pada *tools* ini menggunakan model peringkasan abstraktif. Teknik ini menghasilkan ringkasan dengan memahami konteks dan isi teks asli dan kemudian menulis ulang informasi tersebut dalam bentuk yang lebih ringkas dan sering kali dengan kalimat yang berbeda. Ini lebih mirip dengan cara manusia meringkas teks.

2.3 Model Representasi Teks

Model representasi teks yang digunakan dalam NLP untuk membuat aplikasi *Summarize Tools*. Proses Tokenisasi mengubah teks mentah menjadi token, yaitu unit dasar yang bisa berupa kata, sub-kata, atau karakter, tergantung pada model yang digunakan. Dalam kasus ini, penulis menggunakan tokenizer dari model T5.

Tokenizer T5 menggunakan model berbasis SentencePiece yang mengubah teks menjadi sub-kata token. Ini berarti teks akan dipecah menjadi unit yang lebih kecil dari kata, yang membantu dalam menangani kosakata yang besar dan menangani kata-kata yang jarang muncul.

2.4 Model Pembelajaran Mendalam (Deep Learning)

Dalam proyek ini, model deep learning yang digunakan adalah T5 (Text-To-Text Transfer Transformer) dari Hugging Face's Transformers library. T5 adalah model berbasis Transformer, yang diperkenalkan oleh Vaswani et al. pada tahun 2017. Transformer menggunakan mekanisme perhatian (attention mechanism) untuk memproses input secara paralel dan menangkap hubungan jangka panjang dalam teks.

BAB 3
HASIL DAN PEMBAHASAN

3.1 Pembahasan

3.1.1 Aplikasi yang digunakan

Dalam mengembangkan aplikasi *Summarize Tools* ini, memerlukan satu aplikasi utama untuk mendukung terciptanya aplikasi. Diantaranya:

1. **Google Colab**

   Google Collaboratory atau Google Colab adalah platform berbasis cloud untuk menulis, menjalankan, dan berbagi kode Python melalui web browser. Platform ini dirancang bagi analyst, developer, peneliti, dan pendidik yang bekerja di bidang data science dan machine learning dengan menyediakan environment komputasi yang fleksibel dan mudah diakses tanpa biaya.

3.2.2 *Library* yang digunakan

Dalam proyek ini, beberapa *library* Python digunakan untuk menangani berbagai tugas, termasuk pemrosesan teks, membangun aplikasi *web*, dan menampilkan progress bar. Berikut adalah daftar library yang digunakan:

1. **Flask**

   Sebuah library untuk membuat *web* lokal

2. **Transformers**

   Adalah sebuah library untuk memanfaatkan model-model NLP yang telah dilatih sebelumnya, termasuk T5 untuk peringkasan teks.

3. **PyPDF2**

   Adalah sebuah *library* untuk membuka dan mengekstrak *file* PDF.

4. **Tqdm**

   Adalah sebuah *library* untuk menampilkan *progress bar* dalam aplikasi.

5. **Ngrok**

   Adalah sebuah *library* untuk membuat server lokal atau *localhost.*

3.3.3 Penjelasan Kode Program

1. **Instalasi Ngrok**

```
!wget
https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-
linux-amd64.zip

!unzip ngrok-stable-linux-amd64.zip
```

   Langkah ini mengunduh dan mengekstrak file ngrok yang digunakan untuk membuat server lokal dapat diakses secara publik.

2. **Mengimpor *Library* dan Menyiapkan Flask**

```
import subprocess

import threading

import json

import time

from flask import Flask, request,
render_template_string, jsonify

from transformers import T5Tokenizer,
T5ForConditionalGeneration

import PyPDF2

from tqdm import tqdm



app = Flask(__name__)
```

Bagian ini mengimpor semua library yang diperlukan:

3. **Memuat Model dan Tokenizer**

```
model_name = "t5-small"

tokenizer =
T5Tokenizer.from_pretrained(model_name)

model =
T5ForConditionalGeneration.from_pretrained(model_n
ame)
```

Langkah ini memuat model T5 dan tokenizer yang telah dilatih sebelumnya.

4. **Membuat Fungsi untuk Membaca PDF**

```
def read_pdf(file):

    reader = PyPDF2.PdfReader(file)

    text = ""
```

```
    for page in reader.pages:
        text += page.extract_text()
    return text
```

Fungsi ini membaca teks dari f*ile* PDF yang diunggah.

5. Membuat Fungsi Meringkas Teks

```
def summarize_text(text, update_progress):
    inputs = tokenizer.encode("summarize: " +
text, return_tensors="pt", max_length=512,
truncation=True)
    summary_ids = model.generate(inputs,
max_length=200, min_length=100,
length_penalty=2.0, num_beams=4,
early_stopping=True)
    summary = tokenizer.decode(summary_ids[0],
skip_special_tokens=True)
    update_progress(100)
    return summary
```

Fungsi ini meringkas teks yang diberikan menggunakan model T5 dan memperbarui *progress bar*.

6. **Endpoint Progress**

```
progress = 0

@app.route('/progress')

def get_progress():

    global progress

    return jsonify(progress=progress)

def update_progress(value):

    global progress

    progress = value
```

Bagian ini mengatur endpoint untuk mendapatkan nilai progress dan fungsi untuk mengupdate progress.

7. **Membuat Halaman Aplikasi**

```
@app.route('/', methods=['GET', 'POST'])

def index():

    global progress

    summary = ""

    progress = 0

    if request.method == 'POST':

        file = request.files.get('file')

        text = request.form.get('text')

        if file:

            text = read_pdf(file)

            update_progress(50)  # Update progress
to 50% after reading PDF

            time.sleep(1)  # Simulate delay
```

```
            summary = summarize_text(text,
update_progress)
        elif text:
            update_progress(50)  # Update progress
to 50% after receiving text
            time.sleep(1)  # Simulate delay
            summary = summarize_text(text,
update_progress)
    return render_template_string('''
        <!DOCTYPE html>
        <html lang="en">
        <head>
            <meta charset="UTF-8">
            <meta name="viewport"
content="width=device-width, initial-scale=1.0">
            <title>Text Summarization</title>
            <style>
                body {
                    font-family: Arial,
sans-serif;
                    background-color: #f0f0f0;
                    margin: 0;
                    padding: 0;
                }
                .container {
                    width: 80%;
                    margin: 0 auto;
                    padding: 20px;
```

```
                    background-color: #ffffff;
                    box-shadow: 0 0 10px rgba(0,
0, 0, 0.1);
                    margin-top: 50px;
                    border-radius: 8px;
                }
                h1 {
                    text-align: center;
                }
                form {
                    text-align: center;
                    margin-bottom: 20px;
                }
                input[type="file"],
                textarea {
                    margin: 20px 0;
                }
                textarea {
                    width: 100%;
                    padding: 10px;
                    font-size: 14px;
                    border-radius: 5px;
                    border: 1px solid #ccc;
                    resize: none;
                }
                button {
```

```
        padding: 10px 20px;
        background-color: #007BFF;
        color: #ffffff;
        border: none;
        border-radius: 5px;
        cursor: pointer.
    }
    button:hover {
        background-color: #0056b3.
    }
    .summary {
        margin-top: 20px.
        padding: 20px.
        background-color: #f9f9f9.
        border-radius: 5px.
    }
    .progress-container {
        width: 100%.
        background-color: #e0e0e0.
        border-radius: 5px.
        margin-top: 20px.
    }
    .progress-bar {
        width: 0%.
        height: 30px.
        background-color: #76c7c0.
```

```
                    border-radius: 5px.
                    text-align: center.
                    line-height: 30px.
                    color: white.
                }
            </style>
            <script>
                function getProgress() {
                    fetch('/progress')
                        .then(response =>
response.json())
                        .then(data => {

document.getElementById('progress-bar').style.widt
h = data.progress + '%';

document.getElementById('progress-bar').innerText
= data.progress + '%';
                            if (data.progress <
100) {

setTimeout(getProgress, 500);
                            }
                        });
                }

                function startProgress() {
```

```
document.getElementById('progress-bar').style.widt
h = '0%';

document.getElementById('progress-bar').innerText
= '0%';
                    getProgress();
                }
            </script>
        </head>
        <body>
            <div class="container">
                <h1>Text Summarization</h1>
                <form method="POST"
enctype="multipart/form-data"
onsubmit="startProgress()">
                    <input type="file" name="file"
accept="application/pdf">
                    <textarea name="text"
placeholder="Enter text here..." rows="10"
cols="50"></textarea>
                    <button
type="submit">Summarize</button>
                </form>
                <div class="progress-container">
                    <div id="progress-bar"
class="progress-bar">0%</</div>
                </div>
                {% if summary %}
                <div class="summary">
```

```
                    <h2>Summary</h2>
                    <p>{{ summary }}</p>
                </div>
                {% endif %}
            </div>
        </body>
        </html>
    ''', summary=summary)
```

Bagian ini mengatur halaman utama aplikasi *web* yang memungkinkan pengguna untuk mengunggah *file* PDF atau memasukkan teks untuk diringkas, menampilkan *progress bar*, dan menampilkan hasil ringkasan.

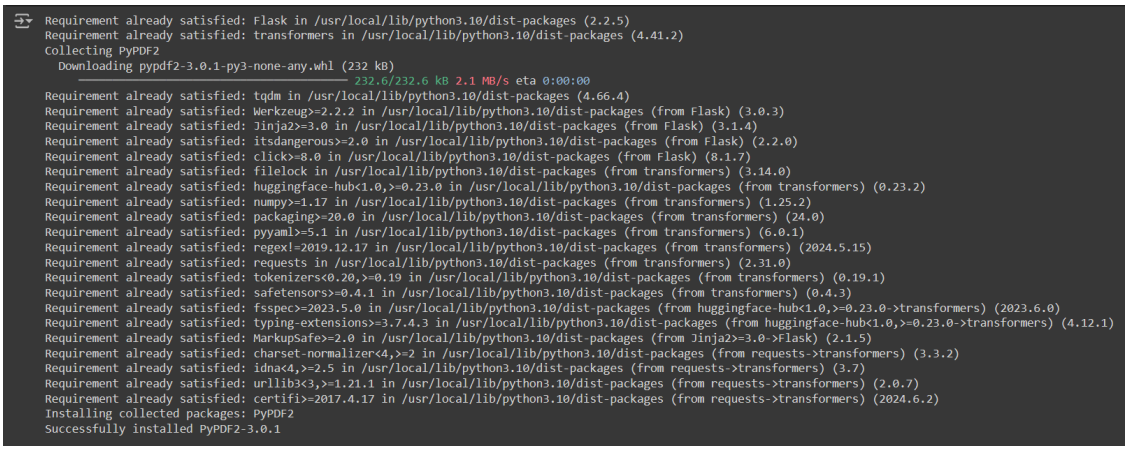
8. **Menjalankan Flask dan Ngrok**

```
def run_flask():
    app.run(host='0.0.0.0', port=5000)

flask_thread = threading.Thread(target=run_flask)
flask_thread.start()

# Menjalankan ngrok
get_ipython().system_raw('./ngrok http 5000 &')
time.sleep(5)  # Beri waktu ngrok untuk memulai
!curl -s http://localhost:4040/api/tunnels | python3 -c \
    "import sys, json; print(json.load(sys.stdin)['tunnels'][0]['public_url'])"
```

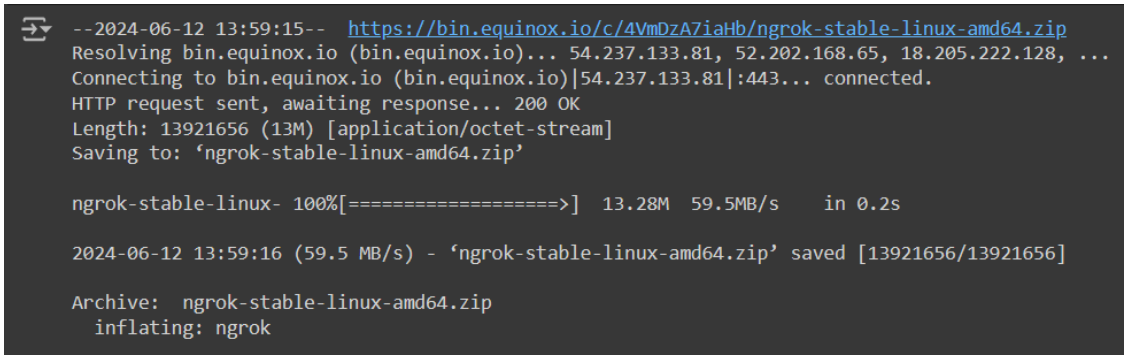
Bagian ini memulai server Flask pada *port* 5000 dan menggunakan ngrok untuk membuat server lokal dapat diakses secara publik. URL publik dari ngrok akan ditampilkan di *output*.

3.3.4 Hasil dan *Output*

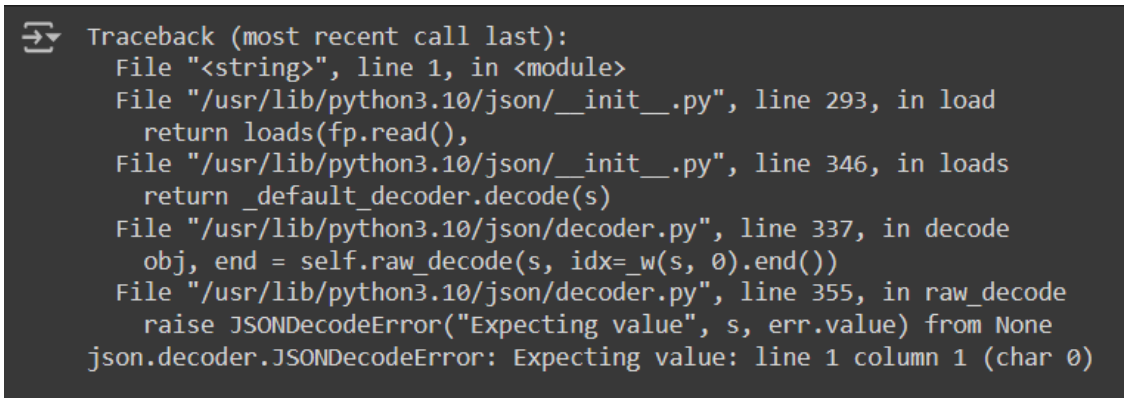
Berikut ini adalah *output* dan hasil antarmuka yang ditampilkan dari kode program setelah dijalankan:



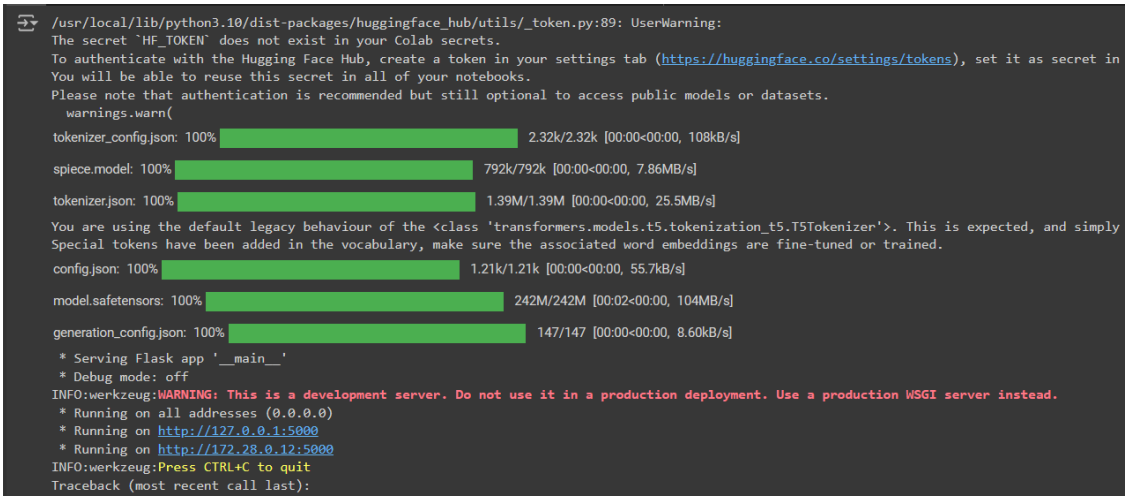
Gambar 1 (*Output* program)



Gambar 2 (*Output* program)



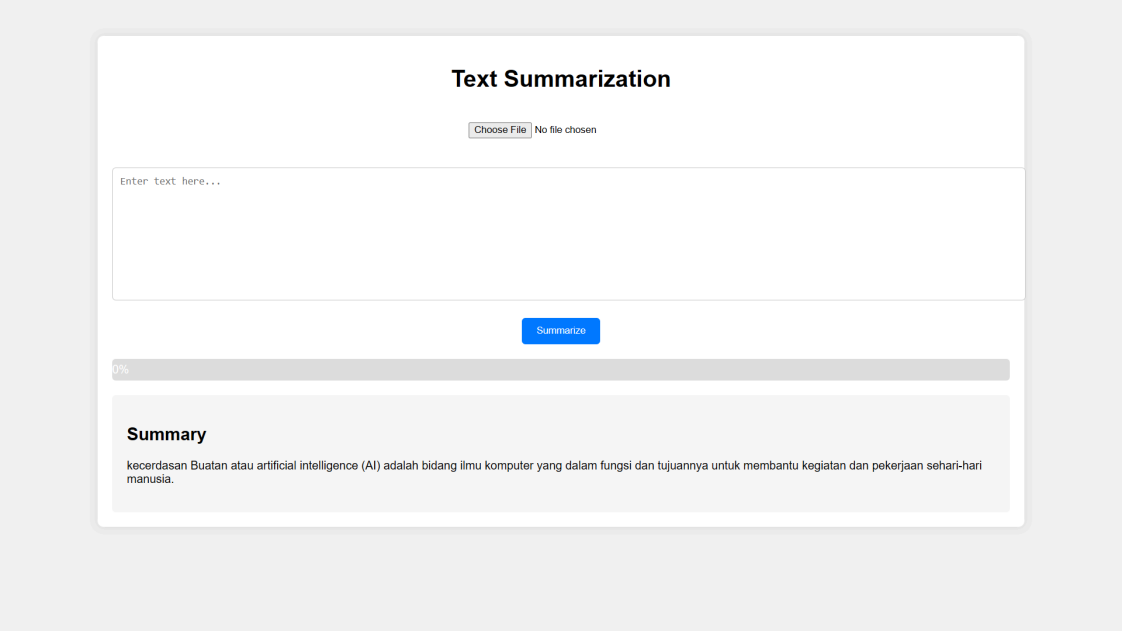
Gambar 3 (*Output* program)



Gambar 4 (*Output* program)

```
Traceback (most recent call last):
  File "<string>", line 1, in <module>
  File "/usr/lib/python3.10/json/__init__.py", line 293, in load
    return loads(fp.read(),
  File "/usr/lib/python3.10/json/__init__.py", line 346, in loads
    return _default_decoder.decode(s)
  File "/usr/lib/python3.10/json/decoder.py", line 337, in decode
    obj, end = self.raw_decode(s, idx=_w(s, 0).end())
  File "/usr/lib/python3.10/json/decoder.py", line 355, in raw_decode
    raise JSONDecodeError("Expecting value", s, err.value) from None
json.decoder.JSONDecodeError: Expecting value: line 1 column 1 (char 0)
```

Gambar 5 (*Output* program)



Gambar 2 (Tampilan antarmuka aplikasi)

BAB 4
KESIMPULAN

4.1 Kesimpulan

Penggunaan NLP AI dalam pembuatan aplikasi *Summarize Tools*, aplikasi untuk membuat ringkasan dari sebuah dokumen atau teks efektif dalam membuat kesimpulan. Penulis berhasil mengintegrasikan teknologi NLP dengan antarmuka web yang sederhana, memungkinkan pengguna untuk dengan mudah mengakses layanan ringkasan teks. Pengguna dapat dengan cepat memasukkan teks atau mengunggah file PDF dan mendapatkan ringkasan teks dalam waktu singkat.

Namun demikian, proyek ini masih memiliki beberapa keterbatasan, seperti kemampuan ringkasan yang terbatas pada teks dalam bahasa Inggris dan keterbatasan dalam jumlah karakter yang dapat diproses oleh model T5. Selain itu, penggunaan ngrok untuk membuat server lokal dapat diakses secara publik dapat memperkenalkan kerentanan keamanan yang perlu dipertimbangkan.

Secara keseluruhan, proyek ini merupakan langkah awal yang baik dalam menggabungkan teknologi NLP dengan aplikasi web, dan dapat diperluas lebih lanjut dengan menambahkan fitur-fitur tambahan seperti dukungan untuk bahasa lain, kemampuan untuk merangkum teks yang lebih panjang, dan peningkatan keamanan sistem.