

Title : AI powered Text Summarizer

1. Project Overview

The AI Text Summarizer is a web-based system designed to automatically condense long texts into concise, meaningful summaries. It provides two types of summarization:

1. **Abstractive Summarization** – Generates a new text that conveys the key points of the original content, similar to how a human would summarize.
2. **Extractive Summarization** – Selects important sentences directly from the original text to form a summary.

The system supports input via **pasted text** or **file upload** (PDF and TXT formats), and can handle multiple files simultaneously. It also evaluates the quality of summaries using standard **ROUGE metrics**, which measure similarity between the generated summary and the original text.

2. System Architecture

2.1 Backend

The backend is built using a web API framework (**FastAPI**) that exposes endpoints for uploading files, submitting text, and retrieving summaries. Its main responsibilities include:

- Accepting text input or uploaded files.
- Extracting text from PDF or TXT files.
- Performing both abstractive and extractive summarization.
- Calculating ROUGE scores to evaluate summary quality.
- Saving summaries and ROUGE results in an organized manner.

Key Features:

- Handles single and multiple file uploads.
 - Supports combining multiple files into a single merged summary.
 - Performs text chunking to manage long documents efficiently.
-

2.2 Summarization Engine

The system uses a dual-method summarization approach:

1. **Abstractive Summarization**
 - Uses a state-of-the-art transformer model designed for text summarization.

- Capable of generating natural language summaries that paraphrase the original content.
- Automatically handles long texts by breaking them into smaller manageable chunks.

2. Extractive Summarization

- Uses an algorithm that identifies and selects the most important sentences from the original text.
- Ensures that key information is retained exactly as written.

3. Quality Evaluation

- ROUGE metrics are calculated for both summary types.
 - Measures include ROUGE-1, ROUGE-2, and ROUGE-L, which quantify overlap in terms of words, phrases, and sentence structure.
-

2.3 Frontend

The frontend is a user-friendly interface that allows users to interact with the summarization system. It provides:

- **Text Input Area:** Users can paste text directly.
- **File Upload:** Users can upload PDF or TXT files for summarization.
- **Single and Multi-File Handling:** Users can summarize one or multiple files simultaneously.
- **Merged Summaries:** Users have the option to combine multiple files into a single summary.
- **Summary Display:** Both abstractive and extractive summaries are displayed with clear formatting.
- **ROUGE Metrics Display:** Users can view quality scores alongside the summaries.
- **Navigation Bar :** Contains Navigation for single file and separate for quick display of summaries of multiple files with cta to multiple files

The interface is designed for usability, with clear labeling, responsive layout, and visual separation between inputs and outputs.

3. Workflow

1. Input Submission

- The user either pastes text or uploads one or more files.
- For files, the system extracts textual content from PDF or TXT formats.

2. Text Processing

- For long documents, the text is split into smaller sections to ensure optimal summarization performance. Note the provided input file **sample.txt** takes **5 mins**

- On laptop and **research pdf** provided takes **9 -11 mins** to process

3. Summary Generation

- The abstractive summarizer generates paraphrased summaries.
- The extractive summarizer selects key sentences from the text.

4. Quality Evaluation

- ROUGE metrics are calculated for each summary type.
- Scores indicate how well the summary represents the original text.

5. Output Delivery

- Summaries and scores are returned to the frontend for display.
- Summaries are also saved in an organized output folder for future reference.

4. Features and Capabilities

- **Multiple Input Types:** Supports both direct text and file uploads.
- **Single and Multi-File Summarization:** Can handle one or several files simultaneously.
- **Merged Summaries:** Optional feature to combine multiple files into a single summary.
- **Dual Summarization Approach:** Provides both abstractive and extractive summaries for comprehensive coverage.
- **Quality Assessment:** ROUGE metrics give users insight into summary accuracy.
- **User-Friendly Interface:** Clean layout with separate areas for input, output, and quality metrics.

5. Benefits

- **Time Efficiency:** Quickly condenses large documents into short, readable summaries.
- **Enhanced Understanding:** Extractive summaries retain key sentences, while abstractive summaries provide paraphrased insights.
- **Quality Feedback:** ROUGE metrics allow users to assess the reliability of summaries.
- **Flexible Input Handling:** Works with both raw text and document uploads, including multiple files at once.

6. Use Cases

1. **Students and Academics:** Summarize long articles, research papers, or lecture notes.
2. **Business Professionals:** Condense reports, meeting notes, or project documentation.

3. **Content Creators:** Generate summaries for blogs, news articles, or research content.
 4. **General Users:** Quickly understand large texts without reading the full content.
-

7. System Requirements

- **Backend:**
 - Python environment with machine learning libraries.
 - Access to sufficient memory and CPU/GPU resources for model inference.
 - **Frontend:**
 - Modern web browser.
 - JavaScript-enabled interface for interactive summarization.
 - **Optional:**
 - GPU acceleration for faster processing of large documents.
-

8. Limitations and Considerations

- Extremely large PDFs may take longer to process.
 - ROUGE scores are useful for comparison but do not always reflect human-like understanding.
 - Abstractive summaries may occasionally miss minor details from the original text.
 - File types are limited to PDF and TXT formats.
 - **Summaries are strictly restricted to 5 lines as per problem statement**
-

9. Future Enhancements

- Support for additional file formats (DOCX, HTML, etc.).
- Batch summarization for large-scale document processing.
- Integration with cloud storage for automatic file input/output.
- User customization of summary length and style.

10. API Routes

The backend exposes several API routes that can be accessed via frontend or directly using tools like **Swagger UI** or **Postman**.

1. Root Route

- **Path:** /

- **Method:** GET
 - **Purpose:** Health check to confirm the backend is running.
 - **Usage:** Returns a simple message indicating the service is operational.
-

2. Single File or Text Upload

- **Path:** /api/files/extract
 - **Method:** POST
 - **Purpose:** Upload a single file (PDF/TXT) or submit raw text for extraction.
 - **Details:**
 - Accepts either a text input or a single file.
 - Extracts the text from the file if provided.
 - Returns the extracted text in JSON format.
 - **Notes:** This route is specifically for **single file or text input**. It can be used directly without a frontend through Swagger UI or similar API testing tools.
-

3. Multiple File Upload and Summarization

- **Path:** /api/files/summarize
 - **Method:** POST
 - **Purpose:** Upload multiple files for summarization.
 - **Details:**
 - Accepts multiple files (PDF/TXT).
 - Optionally merges all files into a single summary if the merge option is enabled.
 - Returns both **abstractive** and **extractive summaries** for each file or the merged summary, along with ROUGE scores.
 - **Notes:** Supports summarization of multiple files simultaneously. Can also be accessed directly without a frontend for automated batch processing.
-

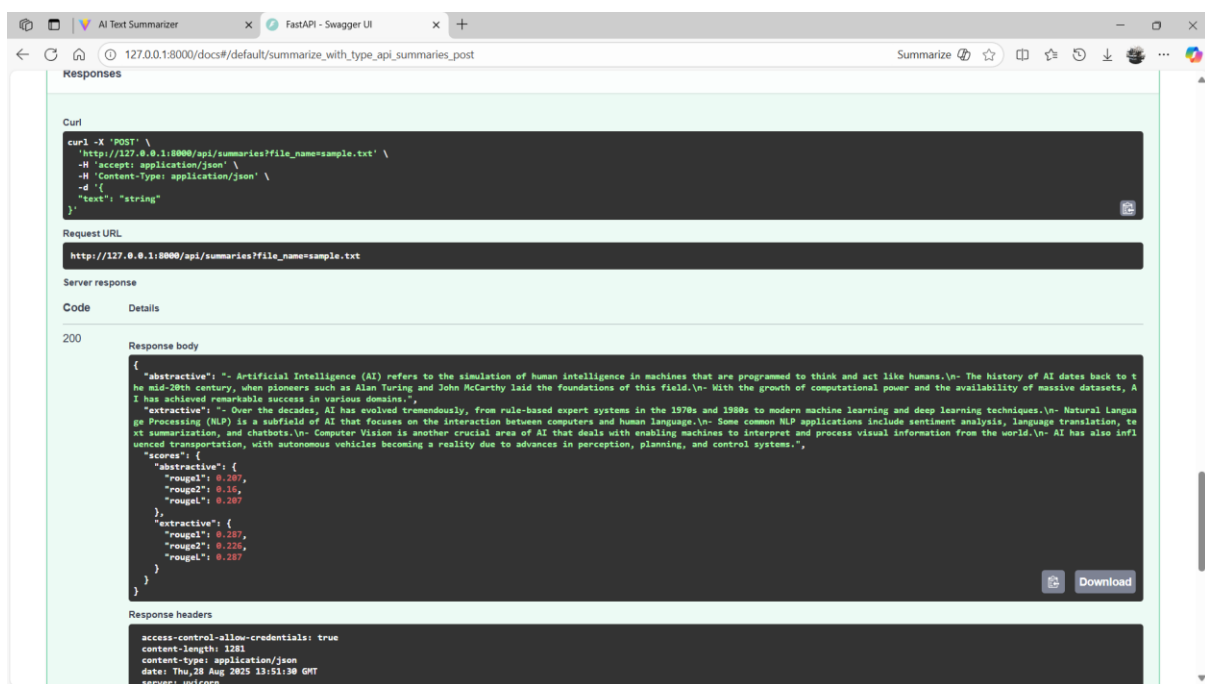
4. Summarization by Text or Existing File

- **Path:** /api/summaries
- **Method:** POST
- **Purpose:** Generate summaries from either raw text or a previously uploaded file.
- **Details:**

- Accepts a text input or the name of a file already stored on the backend.
- Returns both **abstractive** and **extractive summaries**, along with ROUGE evaluation scores.
- **Notes:** Useful for summarization without uploading new files. Can also be used directly via Swagger UI or API clients.

11. Snapshots

Swagger UI of FASTAPI



AI Text Summarizer x FastAPI - Swagger UI +

127.0.0.1:8000/docs#/default/upload_file_or_text_api_files_extract_post

Summarize

default

GET / Root

POST /api/files/extract Upload File Or Text

This endpoint allows uploading a single PDF/TXT file or providing raw text for extraction.

Parameters

No parameters

Try it out

Request body

multipart/form-data

file Upload a single file (PDF/TXT)

string | (string | null) (binary)

text Provide raw text instead of file

string | (string | null)

Responses

AI Text Summarizer x FastAPI - Swagger UI +

127.0.0.1:8000/docs#/

FastAPI 0.10.0 OAS 3.1

default

GET / Root

POST /api/files/extract Upload File Or Text

POST /api/files/summarize Upload And Summarize Files

POST /api/summaries Summarize With Type

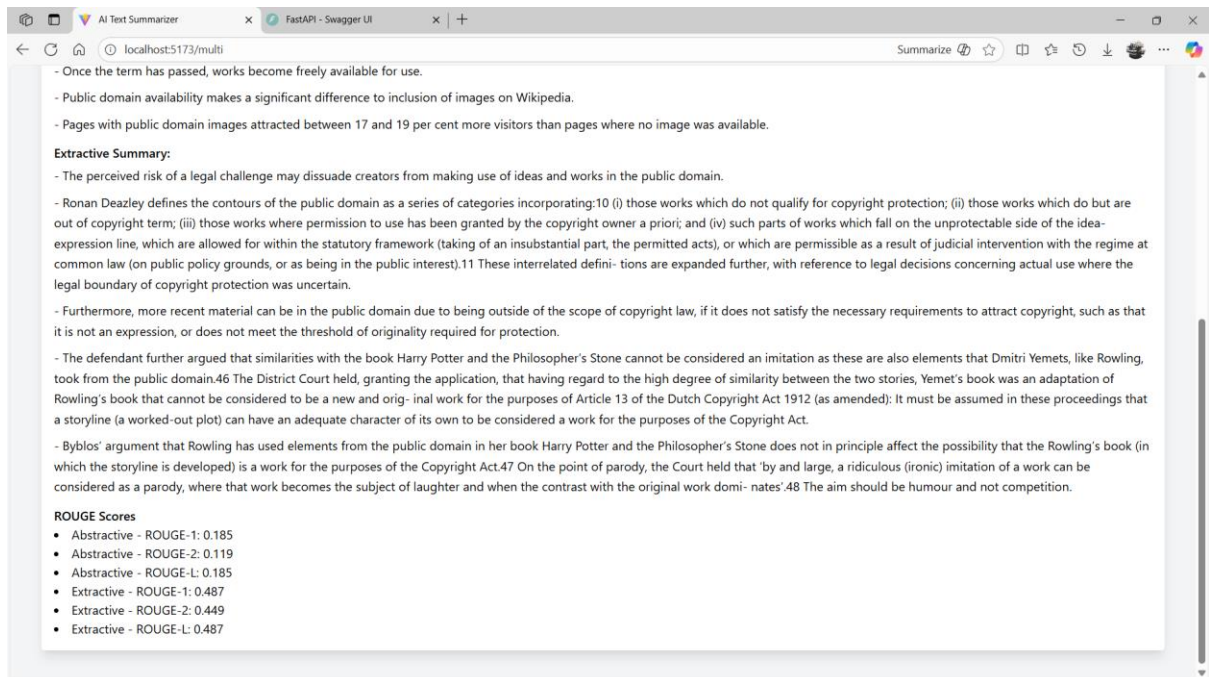
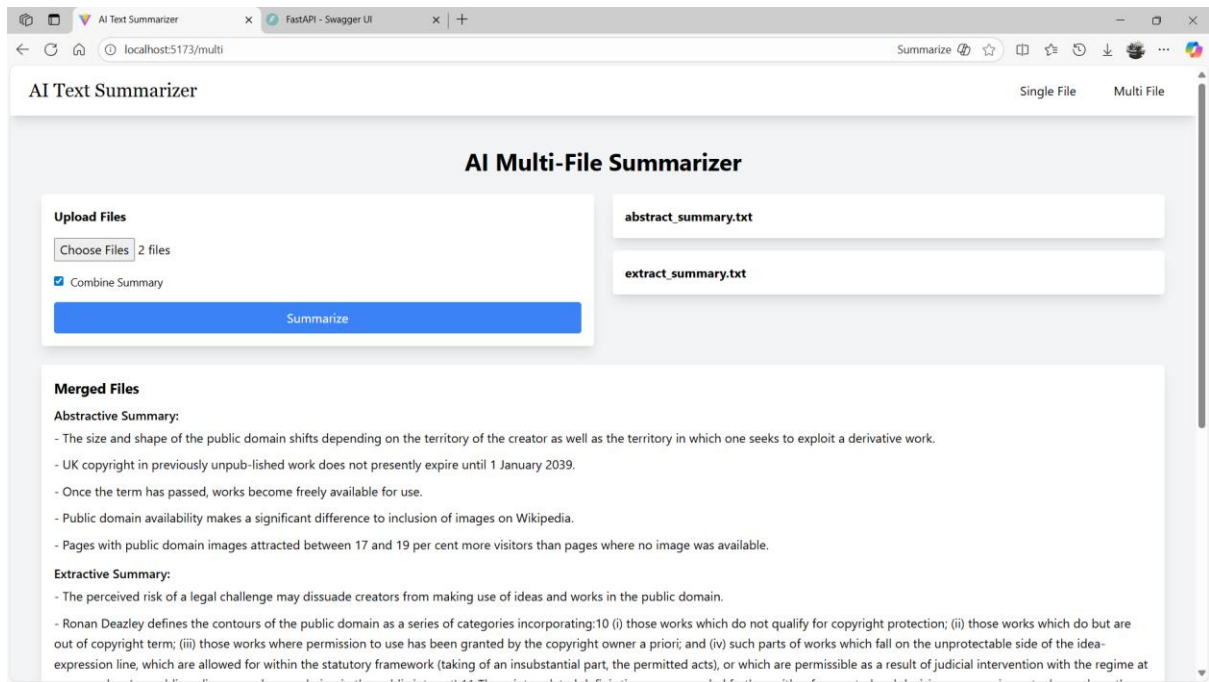
Schemas

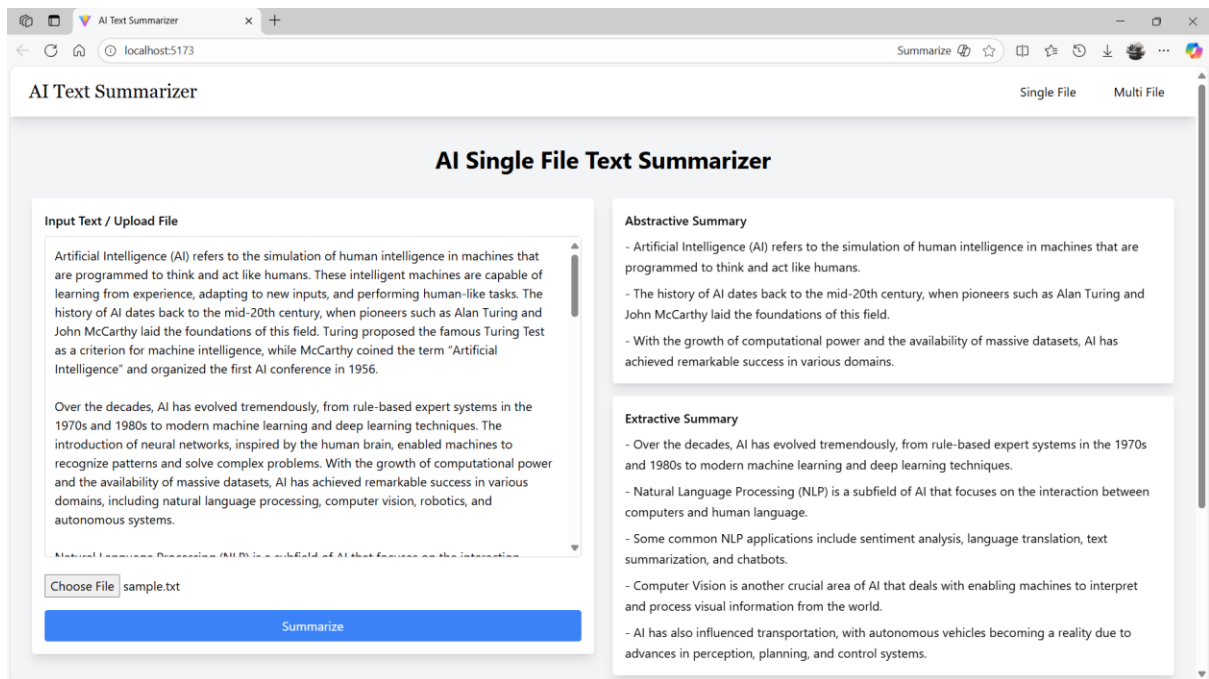
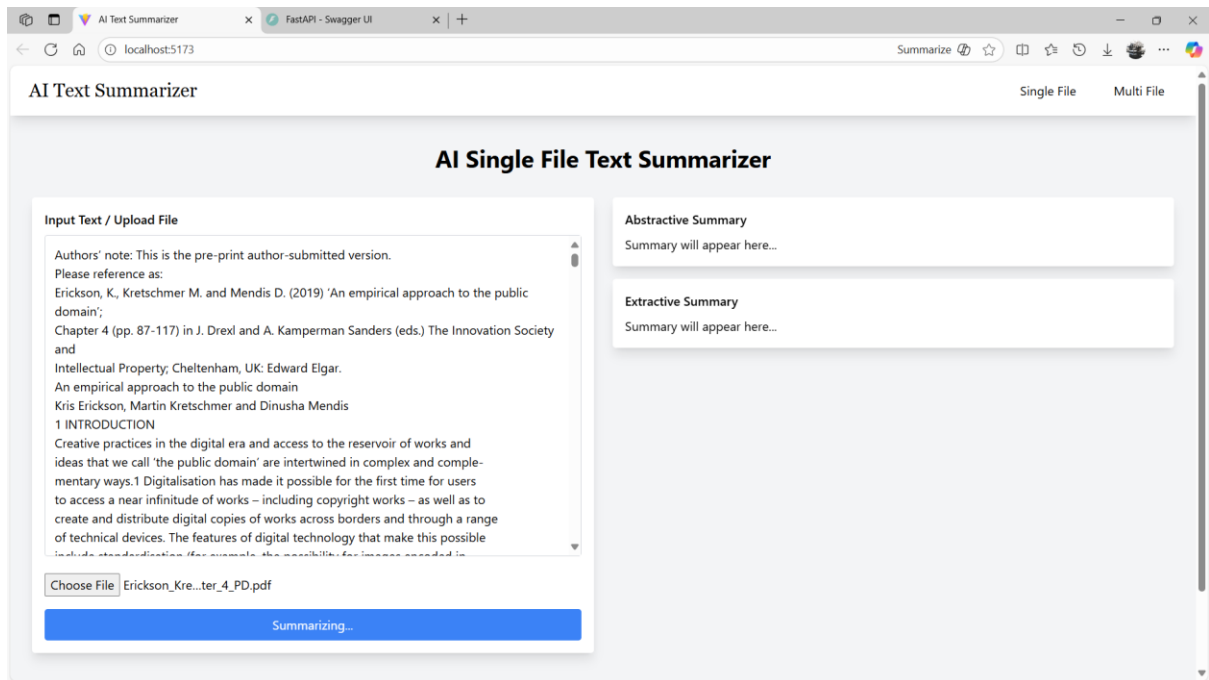
Body_upload_and_summarize_files_api_files_summarize_post > Expand all object

Body_upload_file_or_text_api_files_extract_post > Expand all object

HTTPValidationError > Expand all object

TextRequest > Expand all object





AI Text Summarizer

localhost:5173

Summarize

learning from experience, adapting to new inputs, and performing human-like tasks. The history of AI dates back to the mid-20th century, when pioneers such as Alan Turing and John McCarthy laid the foundations of this field. Turing proposed the famous Turing Test as a criterion for machine intelligence, while McCarthy coined the term "Artificial Intelligence" and organized the first AI conference in 1956.

Over the decades, AI has evolved tremendously, from rule-based expert systems in the 1970s and 1980s to modern machine learning and deep learning techniques. The introduction of neural networks, inspired by the human brain, enabled machines to recognize patterns and solve complex problems. With the growth of computational power and the availability of massive datasets, AI has achieved remarkable success in various domains, including natural language processing, computer vision, robotics, and autonomous systems.

Natural Language Processing (NLP) is a subfield of AI that focuses on the interaction between computers and human language.

Choose File sample.txt

Summarize

Summarize

The history of AI dates back to the mid-20th century, when pioneers such as Alan Turing and John McCarthy laid the foundations of this field.

- With the growth of computational power and the availability of massive datasets, AI has achieved remarkable success in various domains.

Extractive Summary

- Over the decades, AI has evolved tremendously, from rule-based expert systems in the 1970s and 1980s to modern machine learning and deep learning techniques.
- Natural Language Processing (NLP) is a subfield of AI that focuses on the interaction between computers and human language.
- Some common NLP applications include sentiment analysis, language translation, text summarization, and chatbots.
- Computer Vision is another crucial area of AI that deals with enabling machines to interpret and process visual information from the world.
- AI has also influenced transportation, with autonomous vehicles becoming a reality due to advances in perception, planning, and control systems.

ROUGE Scores

- Abstractive - ROUGE-1: 0.207
- Abstractive - ROUGE-2: 0.16
- Abstractive - ROUGE-L: 0.207

- Extractive - ROUGE-1: 0.287
- Extractive - ROUGE-2: 0.226
- Extractive - ROUGE-L: 0.287