**TextMorph: A Text Summarization and Paraphrasing Web Application**

**LAKSHITA SETIA**

**Abstract**

In an era of information overload, individuals often face challenges in comprehending lengthy articles, research papers, and online content. The demand for efficient text summarization and paraphrasing tools has grown significantly, driven by the need to condense large volumes of information while retaining meaning. This project, titled **TextMorph**, is designed to automate extractive and abstractive text summarization and provide paraphrased outputs using state-of-the-art Natural Language Processing (NLP) models. Built with **Streamlit**, the web application integrates **Hugging Face Transformers** and **Groq API** to generate human-like summaries and reworded text. The project demonstrates the synergy of machine learning models, API-based architectures, and user-friendly interfaces in producing efficient language-processing systems.

**1. Introduction**

The explosion of digital content has made it increasingly difficult to read and comprehend every piece of text in full. Whether for students, researchers, or professionals, quickly understanding the core ideas of lengthy documents is essential. **Text summarization** and **paraphrasing** are two fundamental NLP tasks that address this issue.  
TextMorph aims to provide a lightweight, interactive, and accurate summarization and paraphrasing platform accessible via a web interface. Unlike many traditional summarizers, TextMorph allows users to choose between **extractive** and **abstractive** summarization approaches and also generates multiple paraphrased outputs to suit different contexts.

This project demonstrates how AI and NLP can be integrated to develop intelligent applications that enhance content comprehension, productivity, and learning efficiency.

**2. Problem Statement**

With the exponential growth of online text sources such as research papers, blogs, and news articles, manually summarizing or rewriting text has become impractical. Traditional summarization methods, like simple keyword extraction, often fail to preserve semantic meaning. Similarly, paraphrasing manually can introduce redundancy or change the context unintentionally.

Therefore, there is a need for a **comprehensive NLP-based tool** that can:

1. Automatically condense large texts while preserving meaning and readability.
2. Generate paraphrased sentences that maintain the original semantics.
3. Provide an intuitive, browser-based interface for users without technical expertise.

**3. Objectives**

The primary objectives of the TextMorph project are:

* To design and develop a web-based NLP application for summarization and paraphrasing.
* To implement both **extractive** and **abstractive** summarization techniques.
* To integrate the **Groq API** for enhanced text generation performance.
* To allow users to interactively select summarization length and type.
* To maintain flexibility through YAML-based configuration and environment management.

**4. System Architecture / Methodology**

The proposed system follows a modular architecture comprising four major components:

**a. Configuration Management**

A centralized configuration file (config.yaml) is used to store model names, summarization parameters, API keys, and other environment variables. The ConfigManager class loads these configurations dynamically, improving maintainability and scalability.

**b. Summarization Pipeline**

The core logic resides in the mvp\_pipeline.py file, which integrates:

* **Extractive Summarizer:** Uses algorithms like TextRank and NLTK-based sentence scoring to extract key sentences directly from the input text.
* **Abstractive Summarizer:** Utilizes Transformer-based models such as T5 or BART to generate natural language summaries that rephrase the content.
* **Paraphraser Module:** Generates multiple reworded versions of text using Transformer models fine-tuned for paraphrasing.

**c. Web Application Interface**

The frontend is built using **Streamlit**, providing an interactive, minimal, and responsive interface. Users can:

* Paste input text.
* Select summarization type (extractive or abstractive).
* Adjust summary length.
* Generate paraphrased outputs.
* Download results as text files.  
  Custom **CSS styling** ensures an aesthetic and modern look.

**d. API and Environment Handling**

The **Groq API** enhances text generation and speed. API credentials are securely managed using .env files loaded through python-dotenv, preventing sensitive information from being exposed in the repository.

**5. Implementation**

The implementation involved the following steps:

1. **Data Input & Preprocessing:**  
   Input text is processed using NLTK for sentence tokenization and cleaning.
2. **Extractive Summarization:**  
   Key sentences are selected based on similarity and frequency metrics using the TextRank algorithm implemented via **NetworkX**.
3. **Abstractive Summarization:**  
   The Hugging Face Transformer models (like BART/T5) are used to generate concise rephrased summaries.
4. **Paraphrasing Module:**  
   The paraphrasing component uses a transformer-based model to generate multiple reworded outputs for the same input, ensuring grammatical correctness and semantic consistency.
5. **Frontend & Integration:**  
   The backend models are integrated into a Streamlit interface. CSS is used to style buttons, text areas, and layout. The app is deployed locally and can be extended for online deployment using platforms like Hugging Face Spaces or Streamlit Cloud.

**6. Results and Discussion**

The TextMorph application successfully combines summarization and paraphrasing into a unified platform.

* The **Extractive Summarizer** produces concise summaries directly from input text, maintaining factual accuracy.
* The **Abstractive Summarizer** generates fluent, human-like summaries using Transformer models.
* The **Paraphraser** produces multiple creative rewordings, offering users stylistic flexibility.

The system is modular and can easily integrate new models or APIs in the future. YAML-based configurations ensure reproducibility, and .env management enhances security.

Performance testing showed that abstractive summaries closely resembled human-written ones, while paraphrasing produced syntactically diverse yet semantically stable results.

**7. Conclusion**

This project demonstrates how modern NLP models can be integrated into a user-friendly web application to automate summarization and paraphrasing. The architecture balances functionality, flexibility, and usability by leveraging Hugging Face Transformers, Groq API, and Streamlit.  
**TextMorph** provides an effective solution for students, researchers, and professionals who need concise summaries or reworded text. Future work can focus on expanding language support, fine-tuning models for domain-specific texts, and deploying the app for public access.