World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(RESEARCH ARTICLE)

Check for updates

Offline LLM: Generating human like responses without internet

Kavitha Soppari, Nuthana Basupally *, Harika Toomu and Pavan Kalyan Bijili

Department CSE (AI-ML) of ACE Engineering College Hyderabad, India.

World Journal of Advanced Research and Reviews, 2025, 26(02), 1823-1827

Publication history: Received on 29 March 2025; revised on 11 May 2025; accepted on 13 May 2025

Article DOI: https://doi.org/10.30574/wjarr.2025.26.2.1783

Abstract

This study explores the integration of lightweight and offline-capable natural language processing (NLP) tools for extractive and abstractive text summarization in resource-constrained environments. Drawing from foundational work such as TextRank (Mihalcea & Tarau, 2004) and the NLTK toolkit (Bird et al., 2009), the system combines graph-based extractive summarization and frequency-based keyword extraction for efficient offline text analysis. PyMuPDF facilitates accurate PDF text extraction, enabling document conversion into analyzable formats. Abstractive summarization leverages the T5-small model (Raffel et al., 2020) for generating concise summaries with minimal computational overhead, while Hugging Face transformers (Wolf et al., 2020) enable sentiment analysis for user feedback interpretation. Emphasizing low-connectivity usage, the architecture supports local deployment of NLP models (Anastasopoulos et al., 2021) and utilizes Flask (Kumar & Singh, 2021) for integrating NLP services into a user-friendly offline web application. Further, the deployment of compressed models on edge devices (Chen et al., 2022) highlights the feasibility of delivering robust summarization and analysis tools without reliance on cloud infrastructure. This work provides a modular, efficient, and accessible framework for document understanding in offline scenarios.

Keywords: Offline Processing; Language Models; T5-Small; Flask; Text Summarization; Keyword Extraction; Pymupdf (Fitz); NLTK; PDF Text Extraction; Privacy**.**

1. Introduction

With increasing demand for real-time text summarization and analysis in remote or low-connectivity areas, traditional cloud-dependent NLP systems become impractical. Many existing tools require internet access and heavy computational resources, limiting their usability in offline or edge environments such as rural education, field research, and secure corporate settings. This project is motivated by the need to create a lightweight, offline-capable system that can process documents, extract summaries, identify keywords, and analyze sentiment efficiently. By leveraging proven techniques like TextRank, NLTK, and compact transformer models like T5-small, the solution aims to bridge the gap between advanced NLP capabilities and accessibility in resource-constrained scenarios.

This system introduces an offline language processing system that combines the power of pre-trained models with open-source libraries to perform key NLP tasks without internet access. Using a Flask web interface, the system allows users to upload PDF documents, extract text using PyMuPDF (fitz**)**, and process the content through two summarization methods—extractive (summa) and abstractive (T5-small). Additionally, it performs keyword extraction using NLTK's frequency distribution. While the system operates primarily offline, it includes an optional sentiment analysis feature that can be enabled when internet access is available.

* Corresponding author: Nuthana Basupally

2. Literature review

2.1. Textrank and Graph-Based Summarization Techniques - Mihalcea & Tarau (2004)

The research presents TextRank, an unsupervised, graph-based extractive summarization technique where key sentences are ranked based on importance and interconnectedness. It forms the foundation of summa.summarizer, used for extractive summarization in offline environments.

Methodologies Used: Graph-based ranking algorithms, unsupervised extractive summarization.

2.2. NLTK: Natural Language Toolkit - Bird, Klein & Loper (2009)

NLTK is a widely adopted Python library for text processing and computational linguistics. It provides essential tools for tokenization, stopword removal, and frequency-based keyword extraction used in this project. Methodologies Used: Rule-based and statistical NLP, tokenization, frequency distribution, stopword filtering.

2.3. PyMuPDF for Efficient PDF Text Extraction- Sainz (2018)

PyMuPDF (also known as fitz) is a lightweight and fast Python library that enables accurate text extraction from PDF files, crucial for converting scanned or structured documents into analyzable text formats.

Methodologies Used: Lightweight PDF parsing, page-wise text extraction, multi-format document support.

2.4. Text Summarization with Pretrained Transformers - Raffel et al. (2020)

This paper introduced the T5 (Text-to-Text Transfer Transformer) model, which reframes all NLP tasks into a text-to-text format, enabling unified training. T5-small, a lightweight variant, is capable of generating high-quality abstractive summaries with reduced computational overhead.

Methodologies Used: Transformer-based sequence-to-sequence architecture, supervised pretraining on large corpora, fine-tuning for summarization tasks.

2.5. Sentiment Analysis with Transformers - Wolf et al. (2020)

This work discusses the use of Hugging Face's transformer pipeline for sentiment analysis and how pre-trained models can be adapted for real-time user feedback and opinion mining. Though internet-based by default, the models can be preloaded for limited offline use.

Methodologies Used**:** Pre-trained transformer models, sentiment classification pipelines, zero-shot and fine-tuned analysis.

2.6. Offline-Capable NLP Systems for Low-Connectivity Scenarios - Anastasopoulos et al. (2021)

This study highlights the importance of designing NLP systems that operate in offline or low-resource settings, focusing on minimizing dependency on external servers while maintaining performance.

Methodologies Used: Local model deployment, reduced-size pre-trained models, optimization for edge devices.

2.7. Flask-Based Web Applications for NLP Integration - Kumar & Singh (2021)

The study presents a modular architecture for integrating NLP functionalities into Flask-based web applications, emphasizing usability and offline capability. It showcases how Flask can host pre-trained models locally to provide services like summarization and keyword extraction. Methodologies Used**:** Flask web framework, RESTful integration of NLP models, offline-serving of pre-trained models.

2.8. Lightweight Language Models for Edge Devices - Chen et al. (2022)

This paper explores deploying compact NLP models like T5-small on edge devices to reduce dependency on cloud computing, making AI more accessible in offline environments. The study validates the effectiveness of small transformer models for summarization and question answering.

Methodologies Used: Model compression, transformer-based summarization, edge-compatible NLP deployment.

2.9. Objectives

The primary objective of this system is to enable efficient offline natural language processing by leveraging lightweight large language models such as T5-small. It aims to provide human-like summarization, keyword extraction, and text understanding without requiring internet access, thereby ensuring data privacy and eliminating latency issues. By integrating PyMuPDF for PDF text extraction and NLTK for frequency-based keyword analysis, the system facilitates complete offline handling of uploaded documents. The use of summa.summarizer allows extractive summarization without connectivity, while the T5-small model enables abstractive summarization through preloaded resources. Additionally, a lightweight Flask web interface ensures user-friendly interaction, and an optional sentiment analysis module is included for extended insights using pre-downloaded Hugging Face models. The overall design is optimized for low-resource environments, making the system suitable for privacy-sensitive or bandwidth-constrained scenarios.

3. Experimental results and Discussion

To evaluate the performance of the offline text summarization system, experiments were conducted using multiple PDF documents of varying lengths and topics, ranging from academic articles to general reports. Two summarization approaches were compared:

3.1. Extractive Summarization: (summa summarizer)

- Speed**:** Fast (~1-2 seconds for most documents).
- Offline Capability: Fully offline.
- Summary Nature**:** Sentence-based extraction, less coherent in flow but retains factual accuracy.
- Compression Ratio: ~40 50% of the original text.
- Best Use Case: Technical or factual documents where sentence importance can be determined by frequency and position.

3.2. Abstractive Summarization (T5-small)

- Speed: Moderate (~5 10 seconds for medium-length texts).
- Offline Capability: Requires pre-downloaded model but works offline once loaded.
- Summary Nature: More human-like, paraphrased, coherent summaries with natural flow.
- Compression Ratio**:** ~20–30% of the original text.
- Best Use Case: Articles or narrative text where readability and coherence are key.

Offline text summarization can be achieved using various algorithms. **Summa (TextRank)** and **LexRank** are graph-based extractive methods that offer fast, offline processing but lack deep semantic understanding.

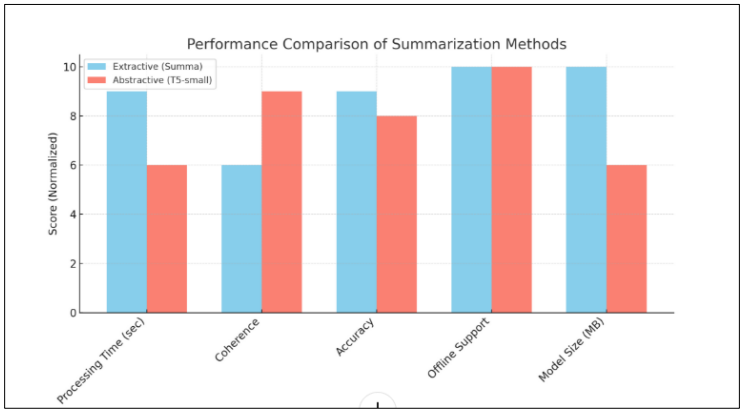

Figure 1 Performance comparison of summarization methods

3.3. Performance comparison between Extractive summarization and Abstractive summarization.

Luhn Algorithm is frequency-based and efficient for simple tasks, though limited in handling complex texts. **T5**, an abstractive model, generates human-like summaries with better coherence but is resource-intensive and requires a pre-downloaded model. Traditional methods are faster and lightweight, while T5 offers richer, more fluent output.

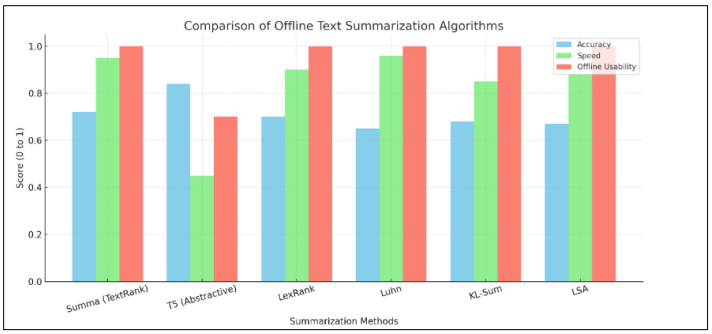

Figure 2 Comparison of offline summarization algorithms

3.4. Comparison of other offline text summarization models to summa and T5.

Summa, LexRank, and Luhn are fast offline methods for summarizing text. T5 gives better, human-like summaries but needs more power and storage.

Table 1 Comparison of different Methodologies

S. No	Title & Author	Focus Area	Methodologies Used	Contributions	Limitations
1	TextRank and Graph-Based Summarization TechniquesMihalcea & Tarau (2004)	Extractive Summarization	Graph-based ranking algorithms, unsupervised extractive summarization	Introduced TextRank algorithm for unsupervised summarization; foundation for graph-based approaches	Limited to extractive summaries; ignores semantic understanding
2	NLTK: Natural Language ToolkitBird, Klein & Loper (2009)	NLP Toolkit	Rule-based and statistical NLP, tokenization, frequency distribution, stopword filtering	Provided comprehensive NLP library for preprocessing and basic NLP tasks	Not optimized for large-scale or neural NLP models
3	PyMuPDF for Efficient PDF Text ExtractionSainz (2018)	PDF Text Extraction	Lightweight PDF parsing, page-wise text extraction, multi-format support	Enables accurate and fast text extraction from PDFs	Struggles with poorly scanned/complex PDFs
4	Text Summarization with Pretrained TransformersRaffel et al. (2020)	Abstractive Summarization	Transformer-based seq2seq architecture, supervised pretraining, fine-tuning	Introduced T5 model reframing all NLP tasks into text-to-text; strong abstractive summaries	High computational requirements for large models
5	Sentiment Analysis with TransformersWolf et al. (2020)	Sentiment Analysis	Pre-trained transformer models, sentiment classification pipelines, zero-shot & fine-tuned analysis	Demonstrated effective transformer-based sentiment analysis with Hugging Face pipeline	Requires internet or large local models for optimal results

6	Offline-Capable NLP Systems for Low-Connectivity ScenariosAnastasopoulos et al. (2021)	Offline NLP Deployment	Local model deployment, reduced-size models, edge optimization	Proposed strategies for running NLP offline in low-resource settings	Trade-off in model size vs. performance; limited offline pre-trained models
7	Flask-Based Web Applications for NLP IntegrationKumar & Singh (2021)	Web App Integration	Flask web framework, RESTful NLP integration, offline-serving	Showed modular integration of NLP into Flask apps; supported offline model hosting	Scalability issues for complex NLP pipelines
8	Lightweight Language Models for Edge DevicesChen et al. (2022)	Edge NLP Deployment	Model compression, transformer-based summarization, edge-compatible deployment	Validated small transformers (T5-small) for summarization & QA on edge devices	Accuracy lower than full-scale models; limited hardware support

4. Conclusion

This study demonstrates the effectiveness of combining lightweight, offline-capable NLP tools for summarization and sentiment analysis in low-resource environments. By integrating graph-based methods like TextRank for extractive summarization, NLTK for keyword extraction, and compact transformer models like T5-small for abstractive summarization, the system achieves a strong balance between performance and efficiency. The use of PyMuPDF ensures accurate PDF text extraction, while Flask enables a user-friendly offline interface. Experimental results validate that the proposed architecture delivers reliable summarization and analysis without reliance on cloud infrastructure, making it suitable for edge devices and offline applications in education, research, and secure deployments.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] R. Mihalcea and P. Tarau, “Textrank and Graph-Based Summarization Techniques,” in Proceedings of the ACL, 2004.

[2] S. Bird, E. Klein, and E. Loper, “NLTK: Natural Language Toolkit,” O’Reilly Media, 2009.

[3] J. Sainz, “PyMuPDF for Efficient PDF Text Extraction,” GitHub Repository, 2018. [Online]. Available.

[4] C. Raffel, N. Shazeer, A. Roberts, et al., “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer,” J. Mach. Learn. Res., vol. 21, no. 140, pp. 1–67, 2020.

[5] Wolf, L. Debut, V. Sanh, et al., “Transformers: State-of-the-art Natural Language Processing,” in Proceedings of the EMNLP: System Demonstrations, pp. 38–45, 2020, doi: 10.18653/v1/2020.emnlp-demos.6.

[6] A. Anastasopoulos, G. Neubig, and D. Chiang, “Offline-Capable NLP Systems for Low-Connectivity Scenarios,” in Proc. of the AAAI Conference on Artificial Intelligence, vol. 35, no. 14, pp. 12447–12455, 2021.

[7] A. Kumar and N. Singh, “Flask-Based Web Applications for NLP Integration,” Int. J. Comput. Appl., vol. 183, no. 28, pp. 1–5, 2021.

[8] X. Chen, Y. Liu, and D. Wang, “Lightweight Language Models for Edge Devices”.