Indian Journal of Computer Science and Technology
https://www.doi.org/10.59256/indjcst.20260503103
Volume 5, Issue3 (September-December 2026), PP: 868-873.
www.indjcst.com
ISSN No: 2583-5300

AI-Based Research Paper Analysis and Assistance System

Nitish Kumar Saw[1], Md. Irfan Alam[2]
[1]*Master of Computer Application (MCA), Jharkhand Rai University, Ranchi, Jharkhand, India.*
[2]*Faculty of Computer Science Engineering & Information Technology, Jharkhand Rai University, Ranchi, Jharkhand, India.*

To Cite this Article: *Nitish Kumar Saw[1], Md. Irfan Alam[2], "AI-Based Research Paper Analysis and Assistance System", Indian Journal of Computer Science and Technology, Volume 05, Issue 03 (September-December 2026), PP: 868-873.*

Copyright: ©2026 This is an open access journal, and articles are distributed under the terms of the Creative Commons Attribution License; Which Permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abstract: *The growing volume of scholarly publications makes it difficult for students and early-stage researchers to examine every paper in depth. This paper presents an AI-based research paper analysis and assistance system that supports the initial examination of research articles through three integrated functions: paper summarization, AI-assisted research-gap suggestions, and context-based question answering. The application accepts a research paper in PDF format, extracts its machine-readable text, performs basic text preparation, and submits the processed content to the Google Gemini large language model (LLM) for task-specific analysis. A Streamlit interface provides access to the analysis functions, while Supabase supports user authentication and paper-history management. Python, pypdf, NLTK, and environment-based configuration are used in the implementation. The system is intended to help users locate key information and develop an initial understanding of a paper; it does not replace close reading, systematic literature review, or expert assessment. The current implementation is text-oriented and may not fully interpret figures, equations, scanned pages, or very long documents. Since no controlled benchmark or user study is reported in the present implementation, the results are discussed in terms of implemented functionality rather than quantitative performance. The paper describes the system architecture, processing workflow, implementation choices, observed capabilities, limitations, and directions for further development.*

Key Word: *Research Paper Analysis, Large Language Models, Scientific Document Summarization, Research-Gap Exploration, Question Answering, PDF Processing, Gemini.*

I. INTRODUCTION

Research articles are a primary means of communicating new knowledge, methods, and findings. However, understanding a technical paper requires more than reading its abstract. A reader may need to examine the research problem, objectives, methodology, experimental findings, limitations, and references before deciding how the work relates to an ongoing study. This process can be demanding for undergraduate and postgraduate students, as well as researchers who are new to a subject area.

Natural language processing (NLP) and large language models have created new ways to interact with lengthy textual material. Transformer-based models use attention mechanisms to represent relationships between words and text segments [1]. Subsequent language models have demonstrated their usefulness in tasks such as language understanding and question answering [2], while broader surveys describe the rapid development and expanding capabilities of LLMs [3]. These developments provide a foundation for tools that help readers navigate scholarly documents.

Scientific articles nevertheless have characteristics that distinguish them from ordinary prose. They follow recognizable structures, use domain-specific terminology, and communicate important information through tables, figures, equations, and citations. Research on scientific-document summarization has highlighted these structural and content-specific challenges [4]. A useful research assistant should therefore do more than produce a generic short summary: it should help readers identify the main components of a paper and ask questions grounded in the document.

This work presents a web-based application that accepts a PDF research paper and offers three core analytical services: (i) concise summarization, (ii) suggestions of possible research gaps based on the submitted paper, and (iii) question answering using the paper as context. It also includes user authentication and paper-history management. The intention is to make preliminary paper examination more accessible through a straightforward interface.

A. Problem Statement and Motivation

Manual paper examination can be time-consuming, particularly when readers are screening several publications or entering an unfamiliar research area. Keyword searches alone may not explain how a paper's problem, method, findings, and limitations connect. The proposed application addresses this practical need by bringing selected first-pass reading tasks into one workflow. It is designed as an assistance tool, not as an automated authority on research quality or novelty.

B. Objectives

i. To develop a web application that accepts research papers in PDF format and extracts their available text.
ii. To generate a concise, structured overview of a submitted paper using an LLM.
iii. To suggest potential areas for further investigation by examining the paper's stated limitations and content.
iv. To support interactive questions whose answers are generated with reference to the submitted paper.
v. To provide authenticated access and maintain user-specific paper activity/history.

C. Main Contributions

The contribution of this work is an integrated, lightweight application that combines PDF text extraction, LLM-based analysis, and user-oriented document management. In particular, it places summarization, exploratory research-gap suggestions, and paper-focused question answering within one interface. The implementation also makes its boundaries explicit: outputs depend on the quality of extracted text and model responses, and proposed gaps require independent verification.

II. RELATED WORK

Automated summarization has a long history in information retrieval and NLP. Nenkova and McKeown reviewed extractive and abstractive approaches, summarization strategies, and evaluation challenges [5]. Mihalcea and Tarau introduced TextRank, a graph-based ranking method for identifying important sentences and words in a document [6]. These approaches provide a foundation for condensing text, although scientific papers require attention to document structure and the relationship between sections.

The summarization of scientific articles has been studied as a distinct problem because research papers contain domain-specific language and information distributed across prose, tables, figures, and citations. Altmami and Menai surveyed automatic scientific-article summarization methods and discussed the limitations of existing approaches, including the need for suitable evaluation resources [4]. Domain-adapted language models such as SciBERT further demonstrate the value of training language representations on scientific publications [7]. The present system takes a practical application-oriented approach: it uses an available general-purpose LLM to analyze extracted paper text rather than training a new scientific-domain model.

Transformer architectures underpin many current language models. Vaswani et al. introduced the Transformer architecture, based on attention rather than recurrent sequence processing [1]. Devlin et al. demonstrated the effectiveness of bidirectional transformer pre-training for language understanding tasks, including question answering [2]. Zhao et al. provide a broad survey of large language models and their development [3]. Together, these studies explain the technical context in which LLM-based document assistance has become feasible.

Question answering over documents is closely related to machine reading comprehension. The SQuAD dataset introduced by Rajpurkar et al. helped establish a widely used benchmark in which systems answer questions using evidence from a supplied passage [8]. Retrieval-augmented generation (RAG), introduced by Lewis et al., combines language generation with retrieved external passages and offers a way to ground answers in accessible evidence [9]. The current application uses the submitted paper text as its context; it does not claim to implement a full retrieval-augmented pipeline.

LLM output quality remains an important concern. Ji et al. survey hallucination in natural language generation and explain how fluent outputs may still be unsupported by the source material [10]. Liu et al. also show that language models can struggle to use relevant information consistently when it appears within long input contexts [11]. These findings are directly relevant to research-paper assistants: a plausible answer may still misrepresent a paper, and performance may degrade when a complete article exceeds the model's effective context capacity.

A. Research Gap and Positioning

The literature establishes strong foundations for summarization, scientific text processing, language understanding, and document question answering. However, a practical user-facing workflow must connect these capabilities with document upload and personal paper management. The system presented here focuses on this integration. Its distinguishing scope is not a new summarization algorithm or a validated method for discovering scientific novelty; rather, it is a compact application that combines three first-pass reading functions with authentication and paper-history support. Compared with RAG-oriented approaches, the present version uses the extracted text as model context and leaves chunk retrieval and evidence-level citation as future improvements.

III. PROPOSED SYSTEM

The proposed system is a browser-accessible application in which an authenticated user uploads a research paper in PDF format and selects an analysis task. The application extracts the document text, prepares it for the language model, and presents the generated response through the Streamlit interface. Supabase is used to manage authentication and user-associated paper activity.

A. System Architecture

The architecture is organized into four logical layers:

i. User interface layer: Streamlit provides the interface for sign-in, PDF upload, analysis-task selection, question entry, and result display.

ii. PDF processing layer: pypdf extracts machine-readable text from the uploaded document. The extracted text is combined and prepared for downstream processing.
iii. AI analysis layer: the processed text and task-specific instruction are supplied to the Google Gemini LLM to generate a summary, possible research-gap suggestions, or an answer to a user question.
iv. Data management layer: Supabase supports authentication and the storage or management of user-related paper history and activity.

The logical flow can be represented as follows:
User authentication → PDF upload → Text extraction → Text preparation → Task-specific Gemini prompt → Generated response → Result display / paper history

B. System Workflow

The workflow starts when a user signs in and uploads a PDF. The application extracts the text available in the file and applies basic preparation before analysis. The user then selects one of the supported functions. For summarization, the model is instructed to present the paper's central problem, objectives, method, findings, and conclusion in a concise form. For exploratory gap analysis, it is asked to examine the paper's stated limitations and identify areas that may merit further study.

For question answering, the user's question is paired with the paper content so that the model can formulate a context-based response. The output is returned to the interface for the user to review.

C. Functional Components

i. PDF upload: accepts a research paper in PDF format.
ii. Text extraction: retrieves the text layer available in the PDF.
iii. Paper summarization: generates a concise overview of the submitted document.
iv. Research-gap assistance: suggests possible unexplored areas or follow-up questions based on the available content.
v. Context-based question answering: responds to user questions using the submitted paper as context.
vi. Paper history: supports access to previous paper-related activity for authenticated users.

IV. METHODOLOGY

A. PDF Text Extraction

The uploaded PDF is processed using the pypdf library. Let the raw text extracted from the document be represented by T_raw. The extraction stage produces a sequence of text content that is passed to the preparation stage. The process can be represented as:

$$T_{clean} = Clean(T_{raw}) \quad (1)$$

Here, Clean(·) denotes the basic text-preparation operations applied by the application. The extraction process is dependent on the PDF's internal text layer. A scanned document that contains page images rather than selectable text may require OCR, which is not part of the current implementation.

B. Text Preprocessing

The extracted content is prepared before it is sent to the LLM. The purpose of preprocessing is to make the text more suitable for analysis by reducing avoidable formatting noise and organizing the extracted content into a usable text sequence. NLTK is included in the text-processing environment. The current paper does not claim a complex linguistic normalization pipeline or a trained preprocessing model; the preparation stage is limited to the operations implemented in the application.

C. LLM-Based Paper Analysis

The prepared text is supplied to the Google Gemini model with an instruction describing the requested task. The model is used as a generative language component rather than as a classifier trained specifically for research-paper analysis. The information sought from a paper can be represented as a set of analytical components:

$$A = \{S, P, O, M, F, L, C, K, G\} \quad (2)$$

where S denotes summary, P the research problem, O the objectives, M the methodology, F the findings, L the limitations, C the conclusion, K the key concepts, and G possible research-gap suggestions. This representation describes the information categories that guide analysis; it is not a claim that every PDF contains all categories explicitly.

D. Research-Gap Suggestion

The research-gap function asks the model to examine the available paper content, particularly the problem statement, discussion, limitations, and future-work statements. Based on these elements, it generates possible directions for further investigation. Such suggestions are exploratory. A limitation stated by one paper does not automatically represent a gap in the wider research field, and an LLM cannot establish novelty solely from one document. Each suggestion must therefore be checked against recent literature, relevant databases, and expert knowledge before being used to formulate a research problem.

E. Context-Based Question Answering

A user may ask a question about the uploaded paper, such as its objective, dataset, methodology, principal findings, or stated limitations. The paper text is supplied as context together with the question. The response can be expressed conceptually as:

Answer = LLM (Context_paper, Question) **(3)**

The quality of the response depends on text-extraction quality, the amount and relevance of context supplied, the clarity of the question, and the model's ability to follow the instruction. Document-grounded answers should be checked against the original paper, especially when they include numerical values, technical claims, or interpretations.

F.Output Review and Responsible Use

The application presents generated content as assistance for the reader. It does not independently verify factual claims, citations, or novelty. Users should compare important statements with the source paper and consult the cited literature. This review step is essential because generative systems can produce unsupported or inaccurate statements [10].

V. IMPLEMENTATION

A. Technologies Used

The application is implemented using the technologies listed in Table I. Each component has a specific role in the workflow.

Technology	Role in the application
Python	Core application development and integration
Streamlit	Web interface and user interaction
Pypdf	Extraction of machine-readable text from PDF files
Google Gemini	LLM-based summarization, gap suggestions, and question answering
NLTK	Text-processing support
Supabase	User authentication and paper-history/data management
python-dotenv	Loading configuration values from environment variables

B. Database and Paper History

Supabase is used for application data management. The database layer associates user activity and paper-related history with authenticated accounts. This allows the application to support a continuing workflow rather than treating every visit as an isolated interaction. The exact data schema and retention policy should be documented in the deployment version of the system, particularly if uploaded documents or extracted text are retained.

C. Authentication and Configuration Security

Authentication is provided so that paper-related activity can be associated with individual users. API credentials and other sensitive configuration values are loaded through environment variables rather than being embedded directly in the source code. This is a basic security practice; a production deployment should additionally apply access controls, secure secret management, input validation, file-size limits, appropriate data-retention rules, and clear user information about how uploaded documents are stored and processed[13,14,15].

VI. RESULTS AND DISCUSSION

The current implementation demonstrates an integrated workflow for initial analysis of PDF research papers. The results reported in this section describe the functions implemented in the application. No controlled experiment, benchmark dataset, expert-rating exercise, or quantitative user study was included in the supplied implementation; therefore, numerical claims about accuracy, time savings, or superiority over other systems are not made.

A. Automated Summarization

The summarization function produces a concise representation of the submitted paper. By presenting the central topic and selected components in a shorter form, it can help a reader decide which sections require closer attention. The output should be treated as a reading aid: details such as assumptions, experimental settings, and qualifications may be compressed or omitted during generation. Scientific summarization research similarly emphasizes that the structure and content of scholarly articles create challenges beyond ordinary text summarization [4], [5].

B. Research-Gap Identification

The system generates possible research directions from the content of the submitted paper. This function may help users notice limitations or future-work statements that deserve further examination. However, its scope is limited to the information supplied to the model. It cannot confirm whether a suggested direction is genuinely unexplored without comparison against the wider literature. The generated output should therefore be described as a candidate research direction, not a verified research gap.

C. Context-Based Question Answering

The question-answering feature offers an interactive way to locate and interpret information in the uploaded document. It can reduce the need to manually search through every page for an initial response. Nevertheless, context-based generation is not guaranteed to be fully faithful to the source. Long-context studies report that models may not use all parts of an input equally well [11], while hallucination research documents the risk of fluent but unsupported generated content [10]. Users should verify important answers against the original text.

D. User and Paper Management

Authentication and Supabase integration provide a foundation for user-specific access and paper-history management. This improves continuity for users who analyze multiple documents. For publication and deployment, the authors should specify precisely which metadata or document content is stored, how long it is retained, and how users can remove it.

E. Qualitative Feature Comparison

Table II compares the broad functional scope described in selected approaches and the present application. The comparison is qualitative and is based on the functionality reported in the source descriptions available to the authors; it is not an independently reproduced benchmark. A dash indicates that the feature was not established from the information available, rather than proving that the system lacks it.

Approach	Summarization	Gap exploration	Paper-based Q&A
Proposed application	Yes	Yes (suggestive)	Yes
Scientific summarization literature [4], [5]	Core focus	Not its primary focus	Not its primary focus
RAG framework [9]	Generation supported	Not a dedicated gap detector	Retrieval-grounded generation
General QA benchmark [8]	Not its focus	Not its focus	Reading-comprehension evaluation

The comparison indicates that the proposed application brings together functions that are often studied separately. Its practical value lies in integration and accessibility. A stronger comparative claim would require a reproducible evaluation using the same documents, questions, metrics, and user tasks across competing systems.

VII. LIMITATIONS

i. The present version has several limitations that should be considered when interpreting its outputs:
ii. Text-only extraction: the application primarily processes the text layer of PDFs. It does not fully interpret figures, plots, equations, complex tables, or scanned pages.
iii. Long-document handling: supplying an entire paper as context may be constrained by token limits and by the model's ability to use information across long inputs [11].
iv. Potential hallucination: summaries and answers may include inaccurate or unsupported statements. Generated text requires checking against the source [10].
v. Unverified research novelty: gap suggestions are derived from the submitted paper and are not validated against the complete body of published literature.
vi. Limited evaluation: the current implementation description does not include quantitative accuracy measures, a benchmark dataset, comparative experiments, or structured user feedback.
vii. Model and service dependency: response quality, availability, latency, and cost may depend on the external LLM service and its configuration.

VIII. FUTURE WORK

i. Several extensions could improve the usefulness, reliability, and research value of the application:
ii. Retrieval-augmented generation: split long papers into meaningful sections or chunks, retrieve passages relevant to each question, and provide those passages to the model. RAG offers a well-established direction for combining retrieval with generation [9].
iii. OCR and layout-aware extraction: add OCR for scanned PDFs and improve extraction of multi-column text, tables, equations, and captions.
iv. Multimodal document understanding: process figures, diagrams, tables, and mathematical expressions so that important non-textual evidence is not ignored.
v. Evidence-linked answers: display the page, section, or supporting passage used to produce each answer, allowing users to verify responses more easily.
vi. Systematic research-gap exploration: compare multiple papers and their references, identify recurring limitations, and connect candidate gaps to current scholarly literature.
vii. Quantitative evaluation: assess summaries using measures such as ROUGE [12], complemented by expert judgments of factual consistency, coverage, and usefulness. Evaluate question answering with a curated question set and human-verified answers.
viii. User-centred evaluation: conduct a study with students and researchers to measure usability, perceived usefulness, task completion time, and the frequency of errors.
ix. Multilingual support: extend the interface and analysis

IX.CONCLUSION

This paper presented an AI-based research paper analysis and assistance system designed to support the initial reading of scholarly PDFs. The application combines PDF text extraction, basic text preparation, Gemini-based summarization, exploratory research-gap suggestions, and context-based question answering within a Streamlit interface. Supabase provides authentication and paper-history management. The system offers a practical way for students and researchers to obtain an

initial overview of a paper and interact with its content. Its outputs, however, should not be treated as verified scientific conclusions. Text extraction constraints, long-context limitations, and the possibility of hallucinated responses require users to check generated information against the original document. Future work will focus on retrieval and evidence linking, multimodal PDF processing, multi-paper analysis, and systematic evaluation. These extensions can help move the application from a useful first-pass assistant toward a more reliable research-support environment.

REFERENCES

1. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in Advances in Neural Information Processing Systems, vol. 30, 2017. Available: https://research.google/pubs/attention-is-all-you-need/
2. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proc. NAACL-HLT, 2019, pp. 4171–4186, doi: 10.18653/v1/N19-1423.
3. W. X. Zhao et al., "A Survey of Large Language Models," arXiv:2303.18223, 2023. doi: 10.48550/arXiv.2303.18223.
4. N. I. Altmami and M. E. B. Menai, "Automatic summarization of scientific articles: A survey," Journal of King Saud University – Computer and Information Sciences, vol. 34, no. 4, pp. 1011–1028, 2022, doi: 10.1016/j.jksuci.2020.04.020.
5. A. Nenkova and K. McKeown, "Automatic Summarization," Foundations and Trends in Information Retrieval, vol. 5, nos. 2–3, pp. 103–233, 2011, doi: 10.1561/1500000015.
6. R. Mihalcea and P. Tarau, "TextRank: Bringing Order into Text," in Proc. EMNLP, 2004, pp. 404–411. Available: https://aclanthology.org/W04-3252/
7. I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A Pretrained Language Model for Scientific Text," in Proc. EMNLP-IJCNLP, 2019, pp. 3615–3620, doi: 10.18653/v1/D19-1371.
8. P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ Questions for Machine Comprehension of Text," in Proc. EMNLP, 2016, pp. 2383–2392, doi: 10.18653/v1/D16-1264.
9. P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in Advances in Neural Information Processing Systems, vol. 33, 2020. Available: https://arxiv.org/abs/2005.11401
10. Z. Ji et al., "Survey of Hallucination in Natural Language Generation," ACM Computing Surveys, vol. 55, no. 12, Article 248, 2023, doi: 10.1145/3571730.
11. N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, "Lost in the Middle: How Language Models Use Long Contexts," Transactions of the Association for Computational Linguistics, vol. 12, pp. 157–173, 2024, doi: 10.1162/tacl_a_00638.
12. C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in Text Summarization Branches Out, 2004, pp. 74–81. Available: https://aclanthology.org/W04-1013/
13. Bihari, S., Alam, M. I., & Khalid, M. S. (2024). Secure data using DNA cryptography in cloud computing environment and maintaining data integrity using SHA-256. Pramana Research Journal, 14(3), 48–58.
14. Alam, M. I. (2020). Enhancing cloud security using multi-level DNA cryptography. Splint International Journal of Professionals, 7(1), 75–82.
15. Alam, I., & Singh, S. N. (2015). Enhancement of cloud security through two level cryptographic algorithms. International Journal of IT, Engineering and Applied Sciences Research, 4(9).