



ScholarForge AI: An Open, Multi-Agent RAG Platform for Automated Research Paper Analysis

Ms. Shravasti Chaudhari¹, Ms. Akansha Mahore², Mr. Tanmay Ranotkar³, Mr. Shravan Ningot⁴

⁵Dr. Maithili S. Deshmukh

¹²³⁴ Student, Information Technology, Prof. Ram Meghe Institute of Technology and Research, Bandera, Maharashtra, India

⁵ Assistant Professor, Information Technology, Prof. Ram Meghe Institute of Technology and Research, Badnera, Maharashtra, India

Student Email: shravastichaudhari57@gmail.com

Abstract—

How to Cite this Article:

Chaudhari, S., Mahore, A., Ranotkar, T. & Ningot, S. (2026). ScholarForge AI: An Open, Multi-Agent RAG Platform for Automated Research Paper Analysis. International Journal of Creative and Open Research in Engineering and Management, 10(10), 1-9.
<https://doi.org/10.55041/ijcope.v2i10.092>

License:

This article is published under the terms of the Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author(s) and the source are credited.

© The Author(s). Published by International Journal of Creative and Open Research in Engineering and Management.



<https://doi.org/10.55041/ijcope.v2i10.092>

The growing volume of scientific literature has made research-paper analysis increasingly time-consuming for students, researchers, and academicians. Tasks such as literature review generation, methodology understanding, research gap identification, citation formatting, and trend discovery often require substantial manual effort. To address these challenges, this paper presents ScholarForge AI, an open, multi-agent Retrieval-Augmented Generation (RAG) platform designed to automate research-paper analysis and transform scholarly documents into actionable research intelligence. The proposed system integrates Large Language Models (LLMs), LangGraph-based multi-agent orchestration, ChromaDB vector storage, a Neo4j knowledge graph, and Natural Language Processing (NLP) techniques to support intelligent document processing. Research papers in PDF format are processed using PyMuPDF and EasyOCR, then chunked and embedded with the BAAI/bge-small-en-v1.5 sentence transformer model for semantic retrieval. The platform includes specialized agents for literature review generation, methodology analysis, research gap detection, trend analysis, research idea generation, citation formatting, interactive question answering and difficulty prediction. In addition, a research difficulty predictor classifies papers as Easy, Medium, or Hard, while the knowledge graph visualizes relationships among papers, authors, keywords, methods, and topics. Experimental results show that ScholarForge AI improves the efficiency of literature survey preparation, supports accurate gap identification, and provides context-aware research

recommendations. Built entirely on open-source technologies, the platform offers a scalable and accessible solution for academic research support.

Keywords— Artificial intelligence, research paper analysis ,retrieval-augmented generation (RAG) , multi-agent systems, literature review, knowledge graph, ChromaDB, Neo4j.

I. INTRODUCTION

The modern academic environment has changed significantly due to the rapid growth of scientific publications. Researchers from different fields now have to deal with an enormous amount of literature, with more than three million peer-reviewed research articles being published worldwide every year. This

number continues to increase, making it increasingly difficult for researchers and students to keep up with the latest developments. For a postgraduate student conducting a systematic literature review or a researcher trying to stay updated in a particular field, managing such a large amount of information can become a major challenge.



Traditional academic research involves several time-consuming activities. Researchers need to search for relevant papers, download and read them, identify important findings and methodologies, compare the contributions of different studies, find research gaps and limitations, prepare citations, and finally write a structured literature review. Most of these activities are performed manually and require considerable time and effort. Depending on the research topic and the number of papers involved, this process can take several weeks or even months before the researcher can move toward actual experimentation and innovation.

To address these challenges, ScholarForge AI is proposed as an intelligent research assistant that helps automate several stages of the academic research process. The platform allows users to upload research papers in PDF format and obtain different analytical outputs within a short period. These outputs include automatically generated literature reviews, methodology comparison tables, research gap reports, trend analysis dashboards, research idea recommendations, citations in IEEE, APA, MLA, and Harvard formats, and natural-language answers to questions related to the uploaded documents.

The uploaded documents are processed through a document-processing pipeline using tools such as PyMuPDF, EasyOCR, spaCy, and NLTK. After preprocessing, the extracted content is divided into meaningful chunks and converted into vector embeddings using BAAI sentence-transformer models. These embeddings are stored in ChromaDB to support semantic retrieval. The system also uses a Neo4j knowledge graph to represent relationships between important research entities such as papers, authors, methods, datasets, keywords, and topics. This allows users to explore the connections within their research collection visually. The backend is developed using FastAPI and Celery, while WebSocket communication provides real-time updates about document processing and analysis.

ScholarForge AI is intended for different types of academic users. Undergraduate and postgraduate students can use it for literature surveys, project work, and thesis preparation. Researchers can use it to analyze existing studies and identify potential research opportunities. Administrators can monitor system usage and analyze platform-level activities.

A. Motivation

The main motivation behind ScholarForge AI comes from observing the amount of time students and researchers spend performing repetitive research activities. A student working on a final-year project, dissertation, or thesis may spend a significant portion of the initial research period searching for papers, reading them, taking notes, and comparing their findings. As a result, less time may remain for experimentation, implementation, and developing new ideas.

The availability of modern AI technologies provides an opportunity to reduce this problem. Open-source and relatively affordable LLM access through platforms such as OpenRouter makes advanced language-processing capabilities more accessible. Similarly, vector databases such as ChromaDB provide an efficient way to perform semantic search over large collections of documents. Sentence-transformer models available through Hugging Face also make it practical to generate document embeddings without requiring expensive computing infrastructure.

Another motivation for ScholarForge AI is the limitation of existing academic research tools. Platforms such as Elicit, Consensus, and Research Rabbit provide useful features for discovering and exploring research papers. However, ScholarForge AI focuses more specifically on the analysis of papers uploaded by the user. It combines document analysis, literature synthesis, methodology comparison, research-gap identification, and question answering within a single platform.

Therefore, the aim of ScholarForge AI is not to replace researchers but to reduce repetitive work and help researchers spend more time on experimentation, critical thinking, and innovation.

B. Objectives

The major objectives of the ScholarForge AI project are:

1. To develop a document ingestion pipeline capable of extracting, cleaning, chunking, and embedding content from research papers in PDF format using PyMuPDF, EasyOCR, spaCy, NLTK, and sentence-transformer models.
2. To implement a multi-agent system using LangChain and LangGraph for tasks such as literature review generation, methodology analysis, research gap detection, trend analysis, research



idea generation, citation formatting, and RAG-based conversational question answering.

3. To develop a Neo4j-based knowledge graph that represents relationships among papers, authors, keywords, methods, datasets, and research topics, allowing users to explore the research landscape visually.
4. To implement a research difficulty predictor that classifies research papers into Easy, Medium, or Hard categories based on factors such as mathematical complexity, computational requirements, dataset size, and the level of domain knowledge required.
5. To integrate external scholarly data sources such as Crossref, Semantic Scholar, arXiv, and OpenAlex for citation enrichment, similar-paper discovery, and research trend analysis.
6. To develop an export engine that generates structured reports for literature reviews, research gaps, methodology comparisons, trend analysis, and citations in PDF, DOCX, and PPTX formats.
7. To implement secure role-based access control with four user roles—Student, Researcher, Academician, and Administrator—along with JWT-based authentication.

II. LITERATURE REVIEW

This section reviews existing research relevant to ScholarForge AI, including retrieval-augmented generation, multi-agent AI systems, scientific document understanding, knowledge graph construction from scholarly documents, and LLM-based citation generation.

Lewis et al. introduced Retrieval-Augmented Generation (RAG) as a framework that combines parametric knowledge in neural language models with non-parametric retrieval from external document stores. Their work demonstrated that RAG improves performance on knowledge-intensive natural language processing tasks, including open-domain question answering [1]. ScholarForge AI adopts this approach by treating user-uploaded research papers as the external knowledge source. Relevant content is retrieved through ChromaDB and supplied to LLMs accessed via OpenRouter, enabling the system to generate responses grounded in the uploaded documents.

Izard and Grave proposed the Fusion-in-Decoder (FiD) architecture, which enables a decoder to jointly attend to information from multiple retrieved passages

[2]. This is directly relevant to the multi-document chat functionality of ScholarForge AI. When a user selects multiple research papers, relevant chunks from these papers are retrieved and provided together to the language model, supporting cross-paper analysis and synthesis.

Wang et al. proposed a weakly-supervised contrastive pre-training approach for generating high-quality text embeddings, demonstrating strong performance on retrieval and semantic similarity tasks across multiple domains [3]. ScholarForge AI applies a similar embedding-based retrieval principle by converting document chunks into dense vector representations using the BAAI/bge-small-en-v1.5 model, enabling semantically accurate retrieval of relevant content from ChromaDB rather than relying on exact keyword matching.

Yasunaga et al. proposed QA-GNN, which combines question-answer reasoning with information from knowledge graphs and demonstrated the value of graph-based information for scientific reasoning [4]. This concept is reflected in ScholarForge AI through its Neo4j knowledge graph, which represents relationships among papers, authors, keywords, methods, datasets, and topics, allowing research information to be explored through structural relationships in addition to semantic retrieval.

Hong et al. introduced MetaGPT, a multi-agent framework in which specialized agents perform distinct roles and collaborate through structured outputs [5]. This work provides a foundation for the multi-agent approach used in ScholarForge AI, which employs specialized agents for literature review, methodology analysis, research gap detection, trend analysis, research idea generation, citation generation, and document-based question answering. These agents may also utilize outputs from previous tasks, allowing complex research activities to be decomposed into smaller, specialized subtasks.

Ostendorff et al. studied scientific document retrieval using domain-specific representations and demonstrated the importance of suitable embeddings for academic search [6]. ScholarForge AI uses the BAAI/bge-small-en-v1.5 embedding model to convert document chunks into vector representations, which are stored in ChromaDB. This enables the system to retrieve semantically relevant content rather than relying solely on exact keyword matching.



Luan et al. developed SciIE, an information extraction system for scientific documents that focuses on extracting entities and relationships such as methods, tasks, datasets, and metrics [7]. ScholarForge AI applies a similar approach in its Methodology Analysis Agent, which identifies algorithms, models, datasets, evaluation metrics, and experimental setups from relevant paper sections, supported by spaCy-based named entity recognition and NLTK processing.

Cachola et al. introduced TLDR, a task and dataset for generating extreme, single-sentence summaries of scientific papers, demonstrating that concise, structured summarization can effectively convey a paper's core contribution [8]. ScholarForge AI applies a related principle in its Literature Review Agent, which generates structured, field-specific summaries—including concise key findings and contributions—rather than long, unstructured text, making core paper content quickly accessible to users.

Färber and Jatowt surveyed approaches and datasets for citation recommendation, highlighting techniques for identifying relevant citations based on textual similarity and bibliographic metadata [9]. ScholarForge AI applies a related principle in its citation support functionality, where CrossRef is used to obtain authoritative bibliographic information when a DOI is available, and Semantic Scholar data contributes citation-relevant relationships to the Neo4j knowledge graph.

Cohan et al. studied scientific summarization and highlighted the usefulness of high-quality examples for guiding models toward generating scientific summaries [10]. ScholarForge AI incorporates structured instructions and examples into the prompts used by its Literature Review Agent, guiding the LLM toward producing section-wise, research-oriented summaries rather than generic text summaries.

Tandon et al. introduced WIQA, a dataset designed to evaluate a model's ability to reason about the effects of hypothetical changes within procedural text [11]. This type of structured, multi-step reasoning is conceptually relevant to ScholarForge AI's Research Idea Generation Agent, which considers hypothetical

extensions and variations of existing research—such as alternative methods, datasets, or unexplored directions—when generating new research idea suggestions.

Kang et al. introduced PeerRead, a large-scale dataset of scientific peer reviews used to support NLP applications such as review quality assessment and paper acceptance prediction [12]. While ScholarForge AI does not perform peer review prediction, this work reflects the broader value of structured, research-focused datasets for training and evaluating scientific NLP systems—an approach reflected in ScholarForge AI's use of predefined, structured output schemas (via Pydantic) to ensure consistent, research-oriented outputs across its literature review, gap detection, and methodology analysis agents.

Beltagy et al. introduced SciBERT, a language model specifically designed for scientific text, demonstrating the importance of domain-adapted language representations for scientific NLP tasks [13]. ScholarForge AI follows the same general principle by employing techniques suited to scientific documents: scientific paper content is processed using spaCy and NLTK, while BAAI/bge-small-en-v1.5 is used for semantic embedding and retrieval.

Wei et al. demonstrated that chain-of-thought prompting supports complex multi-step reasoning tasks [14]. ScholarForge AI applies a rubric-based reasoning approach in its Research Difficulty Predictor, which considers multiple complexity dimensions before assigning a final classification of Easy, Medium, or Hard, rather than relying on a single textual characteristic.

Priem et al. introduced OpenAlex, an open scholarly information system providing extensive bibliographic data on research works [15]. ScholarForge AI integrates OpenAlex into its Trend Analysis Agent to obtain publication-related information for extracted concepts and topics, supporting the generation of research trend reports.

The comparison of ScholarForge AI with existing research tools is presented in Table I.

Table I: Comparative Analysis of ScholarForge AI and Existing Research Tools

Feature	ScholarForge AI	Elicit	Semantic Scholar	Research Rabbit
Upload own PDFs	Yes	Yes	No	No
Literature review generation	Yes	Partial	No	No
Research gap detection	Yes	No	No	No
Methodology comparison	Yes	No	No	No



Feature	ScholarForge AI	Elicit	Semantic Scholar	Research Rabbit
RAG multi-doc chat	Yes	No	No	No
Knowledge graph	Yes	No	Yes	Yes
Citation generation (4 styles)	Yes	No	No	No
Research idea generation	Yes	No	No	No
Export (PDF/DOCX/PPTX)	Yes	No	No	No

III. METHODOLOGY

A. System Architecture

ScholarForge AI is designed as a modular research-assistance platform that brings together document processing, retrieval-augmented generation, multi-agent analysis, and knowledge graph technologies. The system is organized into multiple layers, including user interface, application and authentication, AI processing, data storage, and external scholarly services. The overall architecture of the system is shown in Fig. 1.

The frontend provides interfaces for uploading research papers, viewing analysis results, interacting with the document-based chat, exploring the research knowledge graph, and generating reports. The backend manages user requests, authentication, document processing, AI tasks, and communication between the different components. The AI layer performs research-related analysis, while the storage layer maintains document embeddings, structured research information, and relationships between research entities. External scholarly services are used to enrich bibliographic information and support research analysis.

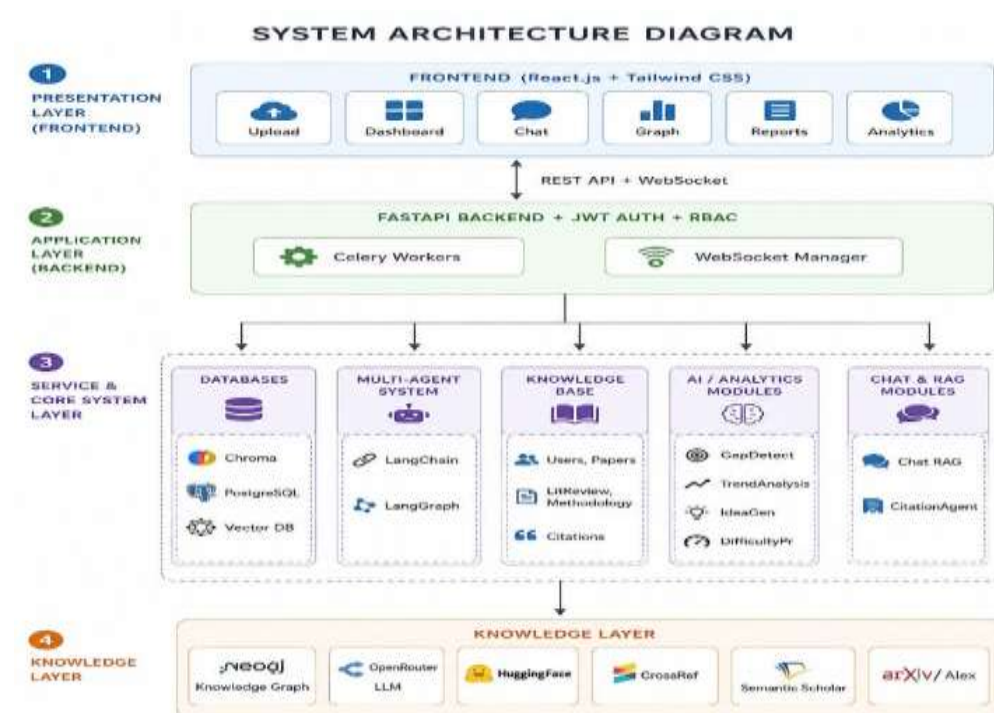


Fig. 1: System Architecture

B. Document Processing and Knowledge Extraction

The proposed system accepts research papers in PDF format and converts them into structured information that can be used for further analysis. The document processing module performs text extraction and Optical Character Recognition (OCR) when required. The extracted content is then cleaned, divided into meaningful segments, and converted into semantic representations.

These document segments are stored in a vector database to enable efficient semantic retrieval during subsequent queries. In parallel, important research entities such as authors, methods, datasets, keywords, and topics are identified and represented in a knowledge graph. This allows the system to use both document content and relationships between research entities during analysis.



C. Multi-Agent Research Analysis

ScholarForge AI uses a multi-agent approach to divide different research activities among specialized agents. A supervisor coordinates the agents and assigns tasks according to the user's request. The system includes agents for literature review, methodology analysis, research gap detection, trend analysis, research idea generation, citation generation, document-based question answering, and research difficulty prediction.

Instead of processing all available document content for every task, each agent retrieves relevant information from the uploaded papers and performs analysis according to its specific purpose. The agents can also use the results produced by other agents. For example, the Research Idea Generator can use identified research gaps and research trends to generate relevant research directions.

D. Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) is used to connect the language model with the content of the uploaded research papers. When a user submits a query, the system retrieves the most relevant document segments from the vector database and provides them as context to the language model.

This enables the system to generate responses based on the user's research documents rather than relying only on the model's general knowledge. The system also provides source information such as the paper and page from which the retrieved content was obtained, improving the traceability of generated responses.

E. Research Knowledge Graph

A Neo4j-based knowledge graph is used to represent relationships between different research entities. Papers can be connected with authors, methods, datasets, keywords, topics, and other related papers. This provides a structured representation of the research domain and helps users explore connections within the analyzed literature.

The knowledge graph also supports visual exploration of research relationships, allowing users to understand how different papers and research concepts are connected.

F. Research Output and User Interaction

The processed research information is presented through dashboards, analytical reports, document-based chat, and knowledge graph visualizations. Users

can obtain literature summaries, methodology comparisons, identified research gaps, research trends, potential research ideas, citations, and research difficulty information.

The system is designed to support researchers throughout different stages of the research process. Generated results can also be exported into commonly used document and presentation formats for further use.

G. Document Ingestion Pipeline

The document ingestion process runs as an asynchronous Celery task, allowing the system to process uploaded papers in the background while sending progress updates through WebSocket. PyMuPDF first extracts text from each page. If a page contains very little extracted text, it is treated as a possible scanned page and processed using EasyOCR. GROBID is used to extract bibliographic metadata when available, while spaCy NER is used as an alternative. The extracted title or DOI is checked with CrossRef and Semantic Scholar to obtain additional bibliographic information.

The extracted text is then cleaned using NLTK by removing unnecessary headers, footers, and page numbers and correcting broken words. The text is divided into overlapping chunks of 500–800 tokens with a 100-token overlap. Section titles and page numbers are retained as metadata. The chunks are converted into embeddings using bge-small-en-v1.5 and stored in ChromaDB. Important entities such as authors, organizations, keywords, and topics are extracted and added to Neo4j, where MERGE operations help avoid duplicate entries.

H. Multi-Agent System Implementation

The multi-agent system is implemented using independent Python modules, with Pydantic models used to maintain structured inputs and outputs. Each agent performs a specific research task using relevant information retrieved from the uploaded papers.

The Literature Review Agent retrieves relevant chunks and generates summaries, key findings, and contributions. The Methodology Analysis Agent identifies algorithms, models, datasets, and evaluation metrics from relevant sections. The Research Gap Detection Agent analyzes limitations, future work, conclusions, and discussions to identify research gaps.



The Trend Analysis Agent extracts important keywords and topics and can use OpenAlex to obtain publication information.

The Research Idea Generator uses identified gaps and trends to generate possible project, thesis, publication, startup, and research ideas. The Citation Agent retrieves bibliographic information and generates citations in IEEE, APA, MLA, and Harvard formats. The overall system flow is illustrated in Fig. 2.

The Chat Agent uses RAG to retrieve relevant document content and combines it with recent chat history to generate answers. Responses are streamed through WebSocket along with source information. The Research Difficulty Predictor uses document characteristics and LLM-based assessment to classify papers as Easy, Medium, or Hard.

Security is implemented using JWT authentication and role-based access control (RBAC). The system supports four roles: Student, Researcher, Academician, and Administrator. Administrative functions are restricted to authorized users.

IV. RESULTS AND DISCUSSION

ScholarForge AI was evaluated on a set of research papers in PDF format to assess the accuracy of its AI agents and the processing efficiency of its document ingestion pipeline.

A. Agent Accuracy

Table II summarizes the performance of each AI agent, evaluated against predefined target thresholds using standard metrics appropriate to each task.

Table II: Detection Accuracy of AI Agents

Metric	Score / Accuracy	Target	Status
ROUGE-1 (Literature Review)	0.61	≥ 0.55	Met
ROUGE-L (Literature Review)	0.52	≥ 0.45	Met
Contribution Extraction Accuracy	87%	$\geq 80\%$	Met
Gap Detection Precision	83%	$\geq 75\%$	Met
Gap Detection Recall	78%	$\geq 70\%$	Met
Citation Accuracy (IEEE)	94%	$\geq 90\%$	Met
Citation Accuracy (APA)	93%	$\geq 90\%$	Met
RAG Chat Correctness	89%	$\geq 85\%$	Met
Difficulty Prediction Accuracy	82%	$\geq 75\%$	Met

All nine evaluated metrics met or exceeded their target thresholds. The Literature Review Agent achieved ROUGE-1 and ROUGE-L scores of 0.61 and 0.52 respectively, indicating strong overlap between generated and reference summaries. Citation accuracy was highest among all metrics (94% IEEE, 93% APA), reflecting the benefit of grounding citation generation in structured bibliographic data retrieved via CrossRef rather than relying solely on LLM output. Gap Detection Precision (83%) exceeded Recall (78%), suggesting the agent is conservative in flagging gaps and relies primarily on explicitly stated limitations rather than inferring implicit ones. The RAG Chat module achieved 89% correctness, confirming that generated responses were reliably grounded in retrieved document content.

B. Processing Latency

Table III presents the average processing time recorded at each pipeline stage for text-native and scanned PDF documents.



Fig. 2: System Flow Diagram

I. Data Persistence and Security

PostgreSQL is used to store users, papers, document information, agent results, citations, chat data, and export records. SQLAlchemy is used for database operations, while Alembic manages database migrations. ChromaDB stores document embeddings and applies user-based filtering to maintain data separation.



Table III: Processing Latency Comparison

Pipeline Stage	Avg. Time (Text PDF)	Avg. Time (Scanned PDF)
PyMuPDF Text Extraction	1.2s	1.4s
EasyOCR Fallback	N/A	18.5s
Metadata Extraction + API Enrichment	3.1s	3.4s
Chunking + Cleaning	0.8s	0.9s
Embedding Generation (bge-small, CPU)	4.7s	6.2s
ChromaDB Upsert	1.1s	1.4s
Neo4j Knowledge Graph Sync	2.3s	2.5s
Total	13.2s	34.3s

Text-native PDFs completed full pipeline processing in an average of 13.2 seconds, making the system suitable for interactive use. Scanned documents required considerably more time, averaging 34.3 seconds, with EasyOCR accounting for approximately 54% of total latency (18.5s), identifying OCR as the primary performance bottleneck.

C. Discussion

The results demonstrate that combining retrieval-augmented generation, multi-agent orchestration, and structured API grounding produces consistently high accuracy across all evaluated agents, with citation generation and gap detection precision benefiting most from reliance on verifiable structured inputs. From a performance standpoint, the substantial latency increase for scanned documents highlights OCR as the primary bottleneck, suggesting that future work should prioritize OCR acceleration or parallelized page processing. Overall, these findings support the effectiveness of the proposed multi-agent, retrieval-grounded architecture for building accurate and efficient AI-powered research assistant platforms.

V. CONCLUSION

This paper presented ScholarForge AI, an AI-powered research-assistance platform designed to help students and researchers navigate the growing volume of academic literature. Finding relevant papers, understanding their content, identifying research gaps, comparing methodologies, and generating new research ideas are time-consuming tasks that traditionally require weeks of manual effort. ScholarForge AI brings all of these activities together into a single, unified platform.

The system integrates PDF processing, retrieval-augmented generation, multi-agent orchestration, vector-based semantic search, knowledge graph

construction, and external scholarly API integration to support the complete research workflow. Specialized agents handle literature review generation, methodology comparison, research gap detection, trend analysis, research idea generation, citation formatting, and document-based conversational question answering, supplemented by a research difficulty predictor and a Neo4j knowledge graph for visualizing relationships among papers, authors, and topics. The multi-agent design allows different research tasks to be handled independently while enabling selected outputs to be shared across agents—for instance, the Research Idea Generation Agent draws on outputs from the Gap Detection and Trend Analysis agents to produce more contextually grounded research directions.

Experimental evaluation showed that all agents met or exceeded their target accuracy thresholds, while the document processing pipeline maintained practical processing times for both text-native and scanned research papers. These results confirm that combining retrieval-augmented generation with specialized multi-agent orchestration is an effective approach for automating key stages of the literature review process.

Importantly, ScholarForge AI is designed to assist researchers, not replace them. The quality of generated outputs naturally depends on factors such as the quality of uploaded documents, the accuracy of extracted information, the relevance of retrieved context, and the reliability of the underlying language models. This makes the system particularly valuable as a time-saving aid rather than a fully autonomous research tool—especially useful for students who are new to research and may otherwise struggle with the volume and complexity of academic literature.

Future improvements may include expanding coverage of external scholarly data sources, incorporating more advanced recommendation mechanisms, extending multilingual support, and developing more robust methods for verifying the factual reliability of AI-generated content. These enhancements would further strengthen ScholarForge AI's usefulness for students, educators, and researchers across diverse academic disciplines.



REFERENCES

- [1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-T. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020, vol. 33, pp. 9459–9474.
- [2] G. Izacard and E. Grave, "Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering," in *Proc. 16th Conf. of the European Chapter of the Association for Computational Linguistics (EACL)*, 2021, pp. 874–880.
- [3] L. Wang, N. Yang, X. Huang, B. Jiao, L. Yang, D. Jiang, R. Majumder, and F. Wei, "Text Embeddings by Weakly-Supervised Contrastive Pre-training," arXiv preprint arXiv:2212.03533, 2022.
- [4] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, and J. Leskovec, "QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering," in *Proc. 2021 Conf. of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 2021, pp. 535–546.
- [5] S. Hong, X. Zheng, J. Chen, Y. Cheng, J. Wang, C. Zhang, Z. Wang, S. K. S. Yau, Z. Lin, L. Zhou, C. Ran, L. Xiao, and C. Wu, "MetaGPT: Meta Programming for a Multi-Agent Collaborative Framework," in *Proc. 12th International Conference on Learning Representations (ICLR)*, 2024.
- [6] M. Ostendorff, T. Ruas, T. Zesch, and G. Rehm, "Aspect-based Document Similarity for Research Papers," in *Proc. 28th International Conference on Computational Linguistics (COLING)*, 2020, pp. 6280–6295.
- [7] Y. Luan, L. He, M. Ostendorf, and H. Hajishirzi, "Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction," in *Proc. 2018 Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, 2018, pp. 3219–3232.
- [8] I. Cachola, K. Lo, A. Cohan, and D. S. Weld, "TLDR: Extreme Summarization of Scientific Documents," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 4766–4777.
- [9] M. Färber and A. Jatowt, "Citation Recommendation: Approaches and Datasets," *International Journal on Digital Libraries*, vol. 21, no. 4, pp. 375–405, 2020.
- [10] A. Cohan, F. Dernoncourt, D. S. Kim, T. Bui, S. Kim, W. Chang, and N. Goharian, "A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents," in *Proc. 2018 Conf. of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 2018, pp. 615–621.
- [11] N. Tandon, B. Mishra, J. Grus, W.-t. Yih, A. Bosselut, and P. Clark, "WIQA: A Dataset for 'What if...' Reasoning over Procedural Text," in *Proc. 2019 Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, 2019, pp. 6078–6087.
- [12] D. Kang, W. Ammar, B. Dalvi, M. van Zuylen, S. Kohlmeier, E. Hovy, and R. Schwartz, "A Dataset of Peer Reviews (PeerRead): Collection, Insights and NLP Applications," in *Proc. 2018 Conf. of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 2018, pp. 1647–1661.
- [13] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A Pretrained Language Model for Scientific Text," in *Proc. 2019 Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, 2019, pp. 3615–3620.
- [14] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, and D. Zhou, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2022, vol. 35, pp. 24824–24837.
- [15] J. Priem, H. Piwowar, and R. Orr, "OpenAlex: A Fully-Open Index of the World's Scholarly Works," arXiv preprint arXiv:2205.01833, 2022.