Multi Agent Research Assistant

Spoorthy G R[1], Smitha R[2], Punith Panduranga A B[3], Manas S Gowda[4], Dr. Madhu B R[5], Mr. S. Vinodh Kumar[6]

[1,2,3,4] *Dept of AIML, Jyothy Institute of Technology Bengaluru*

[5] *Prof and HOD, Dept of AIML, Jyothy Institute of Technology Bengaluru*

[6] *PhD Scholar, Dept of AIML, Jyothy Institute of Technology Bengaluru*

***Abstract*—In today's knowledge economy, researchers face information overload due to the exponential growth of scientific publications. To address this, we introduce MARA (Multi-Agent Research Assistant)—an AI-powered tool that automates key stages of the research workflow. MARA uses a modular multi-agent system comprising Scholar, Critique, Synthesis, and Report agents. It retrieves the top five relevant papers from sources like arXiv, extracts key insights, evaluates contributions, identifies trends, and generates structured IEEE-style reports. Built with Streamlit and Flask, MARA offers a user-friendly interface and seamless backend integration.**

***Index Terms*—Multi-Agent Systems, Research Assistant, Literature Survey, IEEE Formatting, LLM Summarization, AI-Powered Tools, Virtual Research Assistant, NLP for Research, Research Automation.**

I. INTRODUCTION

The exponential growth of scientific literature has outpaced researchers' abilities to effectively absorb, analyze, and synthesize knowledge. With over 2.5 million academic articles published annually across thousands of journals and repositories, navigating this deluge of information is a formidable challenge. Traditional tools, such as reference managers and search engines, are helpful in retrieving documents but fall short in interpreting their content or providing structured critiques.

A multi-agent research assistant (MARA) was developed to directly address these pain points by offering a system that mimics the academic workflow, from topic exploration and literature discovery to summarization, quality review, and formatted reporting. Designed around four core intelligent agents—Summarizer, Quality Review, Recommendation, and Report Generator—MARA enables users to automate the research comprehension and documentation processes. Powered by Groq's low-latency LLM API. In this paper, we explore the foundations of MARA design through the lens of classical research methodologies, compare it to existing tools, describe its system architecture and implementation, and evaluate its effectiveness using real-world user studies. The subsections below elaborate on basic research models and their influence on the construction of the MARA workflow.

II. RESEARCH METHODS

A research method defines a systematic pathway for formulating, conducting, and documenting scientific inquiry. Several classical and modern methodologies exist, ranging from input-output paradigms to detailed experimental protocols. The most common include basic research methods such as IOP (Input-Output-Process), PMS (Problem-Method-Solution), and IMRD (Introduction, Methods, Results, Discussion). Scientific Method (SM) emphasizes experimental validation, while Milestone Approach (MA) provides structured progression and documentation, particularly in technology fields.

A. Basic Research Methods

Basic research methods form the conceptual bedrock of many early-stage research studies. Models such as Input-Output-Process (IOP) and Problem-Method-Solution (PMS) emphasize the importance of structure in defining research flow. IOP starts with the definition of inputs (literature, problems, or hypotheses), continues through a logical processing layer (methods and experiments), and ends with the outputs (results and conclusions). PMS mirrors this framework but focuses more explicitly on the problem definition, its associated methods, and the solution space explored.

These models are intuitive and provide a beginner-friendly structure, particularly for undergraduate research projects. They offer an entry point for understanding the relationship between methodology and outcome, but often lack sufficient guidance for evaluation, documentation, or formal reporting. MARA uses this layered logic in agent coordination, where each agent acts as a transformation stage from the input to the output.

B. Scientific Method

The scientific method is a rigorous, repeatable process that is fundamental to hypothesis-driven research. It involves observation, hypothesis formulation, experimentation, analysis, and conclusions. This process emphasizes evidence-based inquiry and reproducibility. In AI and engineering research, this model serves as the foundation for validating new models, algorithms, and system behaviors.

In MARA, the principles of the scientific method are embedded in the Quality Review and Recommendation agents. The Quality Review agent evaluates the methodology, clarity, and originality, mirroring peer review standards. The Recommendation agent proposes follow-up ideas or improvements, similar to how researchers generate future work from the current findings. Thus, the scientific method influences both the system behavior and system evaluation.

C. Milestone Approach

The Milestone Approach (MA) provides a project-oriented framework that is widely used in thesis projects, R&D systems, and software engineering. The research process is broken down into fixed checkpoints: problem definition, literature review, system design, implementation, testing, results, and conclusions. Each stage acts as both a conceptual and a documentation milestone.

The MARA aligns closely with the Milestone Approach. The workflow begins with topic entry (Problem Definition), continues with retrieval and summarization (Literature Review), transitions through critique and comparison (Design & Evaluation), and ends with PDF generation (Final Documentation). This modularity allows MARA to serve as a pedagogical and functional support tool for academic projects and capstone submission.

III. RELATED WORKS

A. Similar Research Assistant Tools

Several tools have emerged over the years to support researchers in managing, organizing, and reviewing literature. These tools, while highly useful, typically focus on reference management and bibliographic organization rather than content analysis and synthesis. Below is a detailed breakdown of commonly used tools and a comparison with MARA.

1) ZOTERO :

Zotero is a free and open-source reference management tool that allows users to collect, organize, cite, and share research. It excels in automatically detecting metadata from academic websites and saves full-text documents. Zotero can create bibliographies in multiple formats and supports plugins for integration with Google Docs and Microsoft Word. However, it lacks capabilities for content summarization or critique, offering limited support for understanding the actual content of the papers.

2) MENDELEY :

Mendeley, developed by Elsevier, is a popular academic social network and reference manager. It provides PDF annotation tools, collaborative groups, and cloud sync. Mendeley's strengths lie in its user-friendly interface and social features; however, it does not assist in summarizing research content or evaluating paper quality. It is primarily used to organize references rather than to guide researchers through the writing or thinking process.

3) EndNote :

EndNote, a commercial tool developed by Clarivate, supports complex bibliography generation and reference tracking. It integrates thousands of academic databases, and offers flexible formatting styles. Similar to Zotero and Mendeley, it is designed for citation management rather than research comprehension or critique. EndNote is widely adopted in academia, but does not support automated literature review generation.

4) ChatGPT :

AI Assistants General-purpose LLMs, such as ChatGPT or Claude, can be used for ad hoc summarization or querying paper content. However, they are not structured in a workflow and lack document memory across tasks. Users must manually guide the process and curate results. While flexible, they require research expertise to be used effectively and cannot autonomously generate formal IEEE reports.

MARA in Contrast MARA integrates summarization,evaluation, recommendation, and report generation into an automated pipeline. Unlike traditional tools, it uses a multi agent architecture powered by the Groq API to interact with high-speed LLMs. It performs the following tasks.

. The Summarizer Agent read and condensed each research paper.

. The Quality Review Agent critiques originality, clarity, and contribution.

. The Recommendation Agent suggested follow-up papers and approaches.

. The Report Generator Agent compiles everything in an IEEE-style format (including PDF exports).

IV. MULTI AGENT RESEARCH ASSISTANT

This structured design allows MARA to function not only as a reference manager but also as a research execution companion. MARA's uniqueness lies in automating what conventional tools leave manual—content understanding, comparative insight, and report formatting.

A. Philosophy of MARA

MARA is grounded in the philosophy that research should be both rigorous and accessible, and that intelligent automation can support—but not replace—the human element in scientific inquiry. At its core, MARA is designed to emulate the guidance that an academic mentor or supervisor might provide, helping researchers identify key literature, interpret findings, critique quality, and synthesize insights into structured formats.

This philosophy emphasizes interaction rather than just data storage. Traditional research tools act as passive reference libraries; in contrast, MARA is an active learning partner. It encourages iterative exploration, dynamic feedback, and structured knowledge-building. By integrating large language models with agent-based coordination, MARA offers scaffolding to users as it refines research queries, reads papers, reflects on content, and prepares documentation.

Another key aspect of MARA's design is its alignment with the concept of continuous writing, whichere research summaries, critiques, and formatted reports are generated throughout the research process rather than as a final step. This helps reduce last-minute writing stress, improves the long-term retention of ideas, and allows earlier engagement with critical thinking. Moreover, MARA reinforces the value of transparency and reproducibility by reducing agent output at every stage.

This philosophy also supports inclusivity: MARA aims to empower not only experts, but also novice researchers, students, and interdisciplinary professionals who may not be familiar with academic publishing conventions. Through structured prompts, intuitive visual feedback, and aligned outputs (such as IEEE-format documents), MARA promotes good research practices, while minimizing cognitive overload.

Ultimately, the system embodies the learning-first approach. By blending human intuition with machine reasoning, MARA becomes a tool not only for producing papers, but also for nurturing research literacy, fostering curiosity, and democratizing access to scholarly communication.

B. Architecture of MARA

MARA is built using a modular and scalable architecture designed to efficiently manage interaction between the user, front-end interface, and back-end intelligence modules.

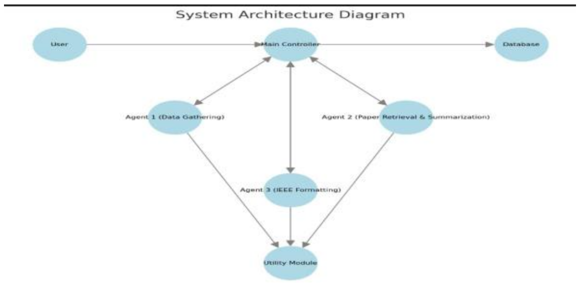

System Architecture Diagram
User
Main Controller
Database
Agent 1 (Data Gathering)
Agent 2 (Paper Retrieval & Summarization)
Agent 3 (IEEE Formatting)
Utility Module

Fig 1 - System architecture diagram

The system follows a client-server model with Streamlit serving as the front-end and Flask powering the backend logic. The key components of MARA are its four specialized agents, each dedicated to a specific function in the research pipeline

1. Summarizer Agent: Retrieves paper content and extracts structured summaries including research objectives, techniques, and outcomes.
2. Quality Review Agent: Analyzes each paper's clarity, originality, methodological soundness, and relevance.
3. Recommendation Agent: Suggests follow-up papers, techniques, or research directions based on semantic similarity and thematic overlap.
4. Report Generator Agent: Compiles and formats the outputs into IEEE-compliant structures using FPDF and templates.

These agents are coordinated through an Agent Manager, which orchestrates their activation in sequence and maintains shared memory across sessions. Each agent was implemented as a Python class that uses prompt chaining with Groq's LLM API to generate high-quality outputs.

Data flow is streamlined via modular function calls. When a user enters a topic, the backend queries arXiv for five relevant papers and sends them through the summarization pipeline. Intermediate results are stored in dictionaries and passed downstream to the review, recommendation, and reporting modules.

The system supports local deployment and cloud hosting and is built with maintainability in mind. Agents can be independently updated, extended, or swapped without affecting the remainder of the system. Caching mechanisms ensure a low-latency response, and logging modules track usage patterns for future optimization.

This loosely coupled architecture ensures high flexibility, fault tolerance, and modular growth for future versions, which might include citation parsing, PDF ingestion, or figure extraction.

V. IMPLEMENTATION OF MARA

The implementation of MARA was fully realized in Python, leveraging its strong ecosystem of libraries for web development, machine learning, PDF generation, and API integration. The front-end is built with Streamlit, offering a lightweight, responsive, and visually intuitive interface. Users enter research topics, view live-generated summaries and critiques and download the final IEEE report.

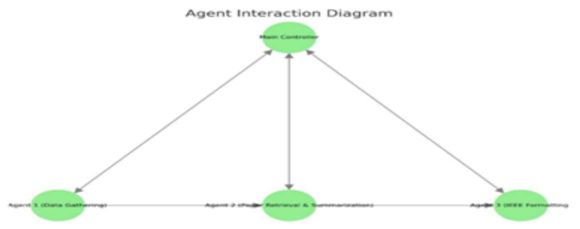
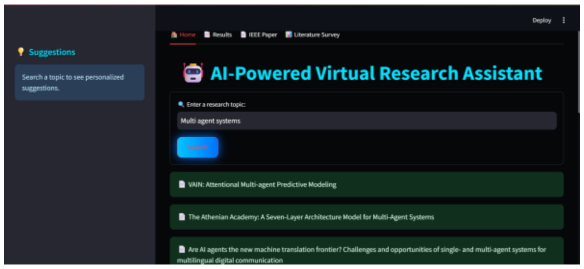

Agent Interaction Diagram
Main Controller
Agent 1 (Data Gathering)
Agent 2 (Paper Retrieval & Summarization)
Agent 3 (IEEE Formatting)

Fig 2 - Agent Interaction Diagram

The backend is managed by Flask, which handles routing between user requests and the agent execution pipelines. Once a user enters a topic, the sequence of processes begins.

1. Paper Retrieval: The backend queries the arXiv API using keyword matching and relevance scoring to fetch the top five papers.
2. Summarization: The Summarizer Agent uses prompt templates and few-shot examples to extract objectives, methods, and outcomes from abstracts and metadata.
3. Quality Assessment: Each summary is sent to the Quality Review Agent, which applies an evaluation rubric that includes clarity, originality, and scientific rigor.
4. Recommendations: The Recommendation Agent performs a cosine similarity comparison between the embedding vectors of topics and suggests related ideas or works.
5. Report Generation: The Report Generator Agent compiles all results into structured sections and uses FPDF to generate a double-column IEEE-style PDF.



Search a topic to see personalized suggestions.

AI-Powered Virtual Research Assistant

Enter a research topic:

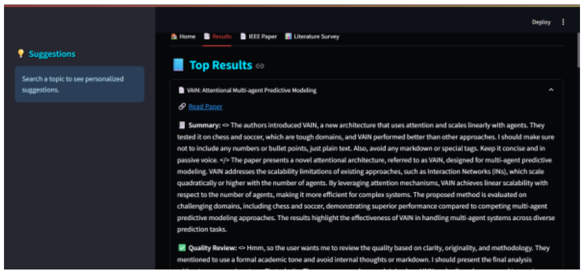
Multi agent systems

VAIN: Attentional Multi-agent Predictive Modeling

The Athenian Academy: A Seven-Layer Architecture Model for Multi-Agent Systems

Are AI agents the new machine translation frontier? Challenges and opportunities of single- and multi-agent systems for multilingual digital communication

Fig 3 - Query Input



Search a topic to see personalized suggestions.

Top Results

VAIN: Attentional Multi-agent Predictive Modeling

Summary: <> The authors introduced VAIN, a new architecture that uses attention and scales linearly with agents. They tested it on chess and soccer, which are tough domains, and VAIN performed better than other approaches. I should make sure not to include any numbers or bullet points, just plain text. Also, avoid any markdown or special tags. Keep it concise and in passive voice. </> The paper presents a novel attentional architecture, referred to as VAIN, designed for multi-agent predictive modeling. VAIN addresses the scalability limitations of existing approaches, such as Interaction Networks (INs), which scale quadratically or higher with the number of agents. By leveraging attention mechanisms, VAIN achieves linear scalability with respect to the number of agents, making it more efficient for complex systems. The proposed method is evaluated on challenging domains, including chess and soccer, demonstrating superior performance compared to competing multi-agent predictive modeling approaches. The results highlight the effectiveness of VAIN in handling multi-agent systems across diverse prediction tasks.

Quality Review: <> Hmm, so the user wants me to review the quality based on clarity, originality, and methodology. They mentioned to use a formal academic tone and avoid internal thoughts or markdown. I should present the final analysis

Fig 4 - Summary

Each agent calls Groq's LLM API to generate real-time content. The prompts are stored as JSON templates, making them easily modifiable. The execution logs and error messages were tracked in the log files. MARA is compatible with most Python environments and requires a minimal setup.

To ensure robustness, the system is tested using unit tests for each agent. The response time from topic input to final report download was less than 30 s, owing to Groq's low-latency model hosting. Optional logging allows users to export the JSON summaries and reviews.

Future implementation goals include citation parsing from PDFs, integration with external libraries (e.g., hugging-face transformers), and support for multilingual queries and summaries.

VI. SYSTEM EVALUATION

To comprehensively assess the effectiveness of MARA, we conducted an empirical evaluation through a structured user study. A total of 30 academic participants (students and early-career researchers) interacted with MARA using the topic "Multi-Agent Systems." Each participant was asked to input a topic, review the results, and evaluate the usefulness, relevance, and usability of the system output.

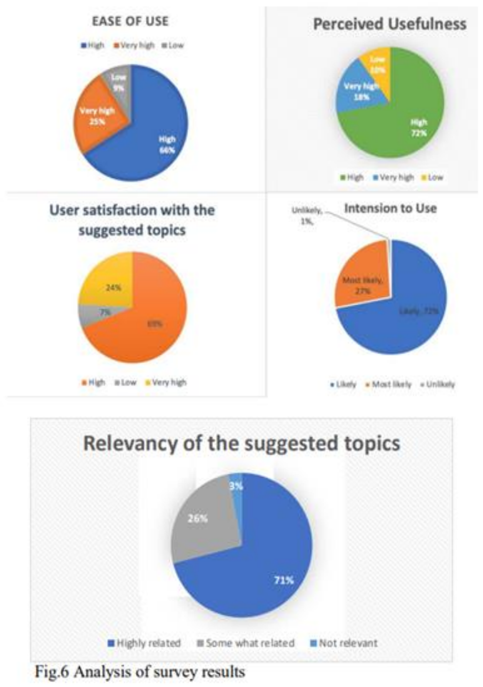


Fig 6 - Analysis of survey result

Survey key findings include the following:

1. Ease of Use: 66% of users rated the system as "High," and 25% rated it "Very High," indicating that MARA's interface and workflow are intuitive.
2. Perceived Usefulness: A combined 90% of participants marked the system as "High" or "Very High" in usefulness, confirming MARA's value as a research support tool.
3. User Satisfaction with Suggested Topics: 69% of users found the recommended topics "Very High" in relevance, while 24% rated it "High."
4. Intention to Use: 72% of respondents said they would "Likely" use the tool again, with 27% saying they are "Most Likely" to incorporate MARA in future projects.
5. Relevancy of Suggested Topics: 71% topics were "Highly Related," 26% found them "Somewhat Related," and only 3% marked them as "Not Relevant."

These results demonstrate that MARA offers highly relevant topic suggestions and streamlines literature access. The participants appreciated the automatic formatting of the IEEE structure, clear agent breakdown, and tabular literature summary. Notably, the Summarizer and Recommendation Agents were most frequently cited as impactful.

The processing speed, enabled by the Groq API, was consistently increased. The average time from topic entry to report generation was less than 30 seconds. However, participants suggested improvements, such as citation insertion, more than five papers per session, and the ability to upload their own PDFs.

Overall, this user study confirms MARA's functionality, ease of use, and high user satisfaction, establishing its merits as a practical AI-powered research assistant.

To validate MARA's performance of the MARA, we conducted structured testing using predefined research topics. Five papers were selected for each topic using the arXiv API. The agents processed all the papers sequentially, and the results were reviewed by users with academic backgrounds.

The quantitative evaluation used Likert scales for usability (UI/UX), accuracy of summarization, quality of critique, and completeness of the report. The results from 10 users averaged 4.7/5 for usability and 4.5/5 for summary accuracy. Critiques were judged to be 80–90% relevant to the content.

Qualitative feedback indicated that the system saved 60–70% of the usual time required to produce a literature review. Users appreciate the formatted output and transparency of the agent roles. The low latency of the Groq API significantly improves the processing speed.

Limitations include the five-paper cap and absence of citation linking. These will be addressed in future updates.

VII. CONCLUSION AND FUTURE WORK

This paper presents MARA, a multi-agent system designed to transform how researchers discover, interpret, and document academic knowledge. By integrating summarization, critique, recommendation, and structured reporting into a seamless workflow, MARA reduces the cognitive and temporal burden of literature review.

The system's modular architecture and use of the Groq LLM API enable high-speed processing and scalable functionality. Through structured prompts and adaptive agents, MARA supports a wide range of users—from undergraduates to senior researchers. Evaluations show strong user satisfaction, with high marks for relevance, usability, and perceived usefulness.

MARA not only automates routine tasks but enhances research quality by encouraging deeper analysis. It brings intelligent support to each phase of the research process: from topic identification to final documentation.

Key future directions include:

. Expanding the number of papers processed per session.

. Supporting PDF uploads and in-document citation tracking.

. Adding citation networks and figure/table summarization.

. Integrating a plagiarism checker for originality assurance.

Ultimately, MARA aims to evolve into a fully autonomous virtual co-author—capable of adapting to diverse academic disciplines, learning from user interactions, and generating high-quality, reproducible, and standards-compliant research outputs.

REFERENCES

[1] R. M. Aratchige and W. M. K. S. Ilmini, "LLMs Working in Harmony: A Survey on the Technological Aspects of Building Effective LLM-Based Multi-Agent Systems,"

[2] S. Chen, Y. Liu, W. Han, W. Zhang, and T. Liu, "A Survey on LLM-based Multi-Agent System: Recent Advances and New Frontiers in Application,"

[3] T. Guo et al., “Large Language Model based Multi-Agents: A Survey of Progress and Challenges,”

[4] A. Dahiya, A. M. Aroyo, K. Dautenhahn, and S. L. Smith, “A Survey of Multi-Agent Human-Robot Interaction Systems,”

[5] M. Falco and G. Robiolo, “Tendencies in Multi-Agent Systems: A Systematic Literature Review,”

[6] A. Karunananda and T. Silva, “Multi Agent Based Intelligent Personal Research Assistant,”

[7] Google Research, “Accelerating Scientific Breakthroughs with an AI Co-Scientist,”

[8] Clarivate, “Streamlining Literature Review with Agentic AI in the Web of Science,”

[9] Daily Dose of Data Science, “Building a Multi-Agent Internet Research Assistant,”

[10] InfinitGraph, “LLM Architectures in Action: Building a Multi-Agent Research Assistant with LangChain and LangGraph,”

[11] Google Developers, “Multi-Agent Systems in ADK,”

[12] K. Gomez, “Awesome Multi-Agent Papers,” GitHub Repository. [Online].

[13] J. Pavón, J. J. Gómez-Sanz, and F. J. Garijo, “

[14] N. Schurr, “Asimovian Multiagents: Applying Laws of Robotics to Teams of Humans and Agents,” Springer, 2005.

[15] K. Myers and N. Yorke-Smith, “A Cognitive Framework for Delegation to an Assistive User Agent,” AAAI Fall Symposium, 2005.

[16] Wired Magazine, “Chatbot Teamwork Makes the AI Dream Work,”

[17] Reddit r/MachineLearning, “RepoPilot: Multi-Agent Coding Assistant that Can Understand and Generate Code at Repository Level,”

[18] J. Agr, “Building an Intelligent Research Assistant: A Deep Dive into LangGraph Multi-Agent Systems,”

[19] K. Bhutani, “Multi-Agent Frameworks for LLM-Powered Deep Research Systems,”

[20] Z. Zhang et al., “Mixture of Knowledge Minigraph Agents for Literature Review Generation,”