AI Agents for Deep Scientific Research

Rui Zhou *ruizhou3@illinois.edu*
ruizhou3

Vir Sikand *vsikand2@illinois.edu*
vsikand2

Sudhit Rao *srrao2@illinois.edu*
srrao2

Reviewed on OpenReview: *https://openreview.net/group?id=illinois.edu/UIUC/Spring_2025/CS598_LLM_Agent_Workshop*

Abstract

AI research agents have the potential to revolutionize scientific discovery, software development, and business innovation by automating complex research tasks and enhancing decision-making. By streamlining repetitive and time-consuming processes, these agents can significantly reduce the effort required for data collection, analysis, and synthesis, allowing researchers and professionals to focus on higher-level problem-solving. Moreover, automation not only saves time and resources but also improves accuracy by minimizing human errors in tasks such as literature reviews, code generation, and data interpretation. Further, AI research agents can even come up with new and novel ideas to enhance researchers' approach to problem-solving in their domain, leading to novel and unconventional but possibly successful methods. In rapidly evolving fields, where staying ahead of new advancements is crucial, AI-driven solutions provide scalable and adaptive approaches to continuously process and integrate the latest knowledge. This enables organizations and individuals to remain at the forefront of innovation, leveraging AI to accelerate progress and uncover new insights more efficiently than ever before. In this paper, we will survey multiple newly invented AI agents.

1 Problem

Scientific discovery and research are the process of generating new hypotheses, designing experiments, validating results, developing new theories, and interpreting data. Scientific discovery has been an important part of human history and has a rich history from the development of Calculus to the creation of the computer. There are numerous fundamental challenges that limit research progression.

One example is the theory-laden observation, in that scientists tend to interpret data in a biased manner, based on existing theories. This limits the way in which scientists find relationships and patterns within data and also restricts their ability to generate new and novel hypotheses. Hypothesis generation is generally a subjective and biased process, fully influenced and limited by human cognitive capacity. There remain many practical challenges as well, from the lack of computing power and the inability to replicate results effectively. Many fields, such as genomics, high-energy physics, and others, simply have a lot of data that cannot be easily understood by humans and require a lot of computational efforts to bridge the gap.

2 Architectural Innovation

2.1 PaperQA2

PaperQA2 is a retrieval-augmented generation (RAG) agent that can help with answering scientific questions. Inspired by PaperQA (Lála et al., 2023), it did not follow the previous model's architecture, which directly processes the question and gives the answer. It separated the retrieval and response generation as a multi-step process (Skarlinski et al., 2024). Specifically, PaperQA2 used multiple "tools" to help with the answer generation (Skarlinski et al., 2024). As shown in the figure 1, firstly, it had the tool of "Paper Search", where the system would transform the user index into different keywords and search keys to find all the papers that are in the user's proposed scope. Then, the system would change these found articles to machine-readable text and cut the articles into chunks. Secondly, the tool of "Gather Evidence" would come into the process. This tool would first help find the top-k chunks using a dense vector retrieval process (Skarlinski et al., 2024). With the top-k chunks, the tool would perform the reranking and contextual summarization (RCS) step to prevent irrelevant chunks from moving into the next tool (Skarlinski et al., 2024). Also, the RCS could help retrieve the metadata of the chunk, such as the citations and the journey the chunk is from (Skarlinski et al., 2024). After getting all the information, the agent could generate the answer. The agent would use "Generate Answer" and "Citation Traversal" tools to help with the answer generation. The "Generate Answer" tool would feed all the chunks and metadata collected to an LLM to give out the final answer to the question, while the "Citation Traversal" tool could help add more sources to the final answer by using the citation graph (Skarlinski et al., 2024).

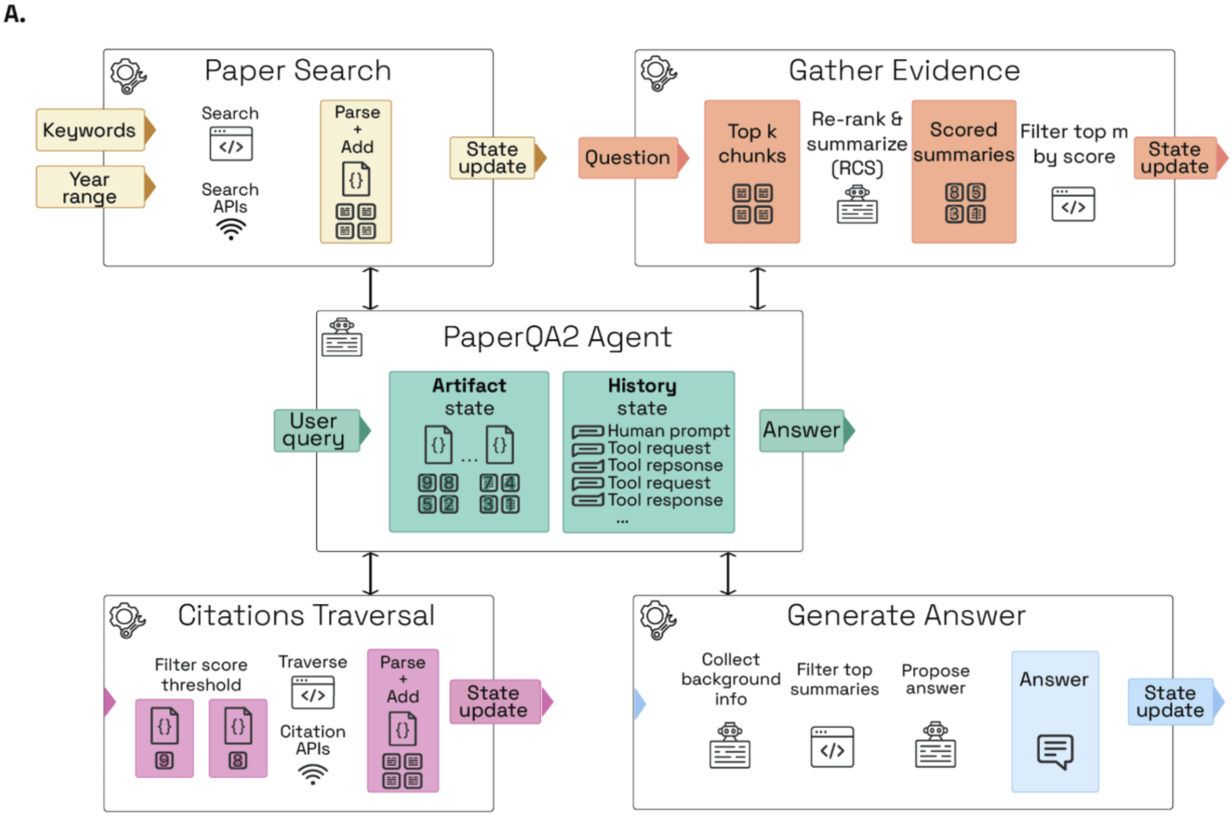


Figure 1: Work Flow of PaperQA2

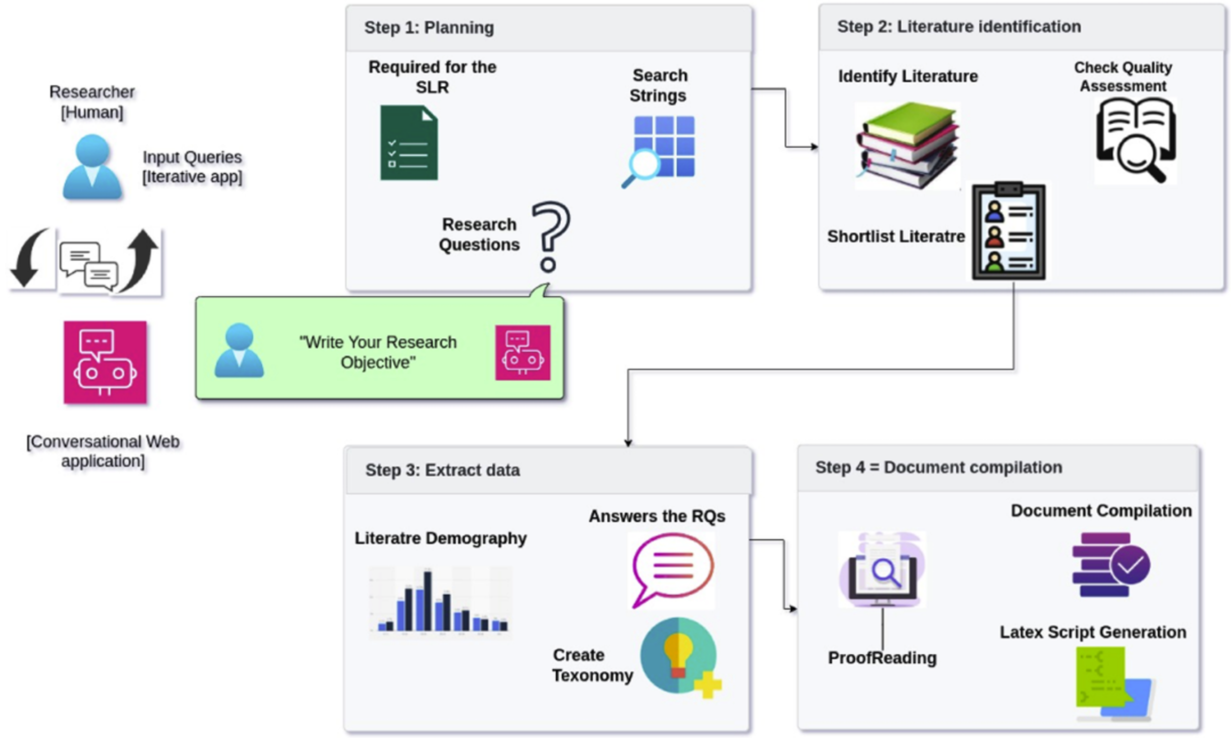


Figure 2: Multi-agents SLR Workflow

2.2 Multi-agents SLR

AI agent can significantly help with the literature review process. The system was designed to help the researcher perform the review process automatically (Sami et al., 2024). As shown in the figure 2, the system consisted of four agents during the whole process. The first one was the Planner Agent, which helped generate the search strings based on the user input (Sami et al., 2024). Specifically, the user would first give an interested topic and research questions, and the Planner Agent would generate a search string that includes the concepts and terminologies of the research topic. Then, the search string would be handed to the Literature Identification Agent. The Literature Identification Agent would first utilize the search string and titles of the literature in the dataset to find the most relevant papers (Sami et al., 2024). With the paper extracted, the Data Extraction Agent could further help filter the paper based on the topic given and answer the research questions from the user (Sami et al., 2024). It would first determine the relevance of the paper based on the abstract of the paper, and then it will further filter based on the full content of the paper. Then, it would try to answer the questions based on the filtered papers. Finally, the Data Compliance Agent will help combine the answers from the Data Extraction Agent and summarize the findings (Sami et al., 2024). Specifically, it will point out the weaknesses and the major findings of the current research regarding the topics given by the user.

2.3 ADAS

Automated Design of Agentic Systems (ADAS) is a new system that can automatically create new LLM-based AI agents by iteratively writing comprehensive code for the agent (Hu et al., 2025). Previous works have done similar work (Yang et al., 2024; Fernando et al., 2023), but they only focused on developing prompts. Here, the method introduced by the author is that it can define the entire agentic system in code, and the system would automatically explore new agents (Hu et al., 2025).

ADAS used an agent, Meta Agent, to help with the entire process. First, the Meta Agent would be given a task and it would generate a new code for the agent that might be able to solve the task. Secondly, the generated agent would started to solve the problems and the performance would be recorded. Based on the performance of the new agent, the Meta Agent would start the reflection process. In other words, the Meta Agent would try to modify the code to solve the errors of the new agent. The reflection process would repeat for 5 times. After the reflection, if the performance of the new agent is high, the agent's code would be saved to an archive. In this way, the later agents could borrow the design of the code in the archive to "upgrade" itself. As a result, the new agents generated by the Meta Agent would be continuously better.

2.4 SurveyAgent

SurveyAgent (Wang et al., 2024) is a new paper that introduces a different approach to helping researchers with conducting literature reviews and surveys more effectively. The key architectural component behind SurveyAgent is the commonly seen ReAct ((Yao et al., 2023))framework to implement a conversational style interface with the researcher. The system is broken down into three distinct modules: a knowledge management module, recommendations module, and a query answering module. The Knowledge module functions as a knowledge base and exposes a bunch of tools/actions to get information about papers and related papers in the same "collection." Further, the recommendations module is implemented as a search engine using keyword search, offering a non-semantic way to find and recommend papers based on query keywords. Lastly, the agent module functions as a personal research assistant, aimed at semantically understanding papers by chunking the papers long size into smaller chunks, feeding into the LLM to determine the relevant chunks, finally filtering and searching through relevant chunks to answer the question.

2.5 AI Scientist

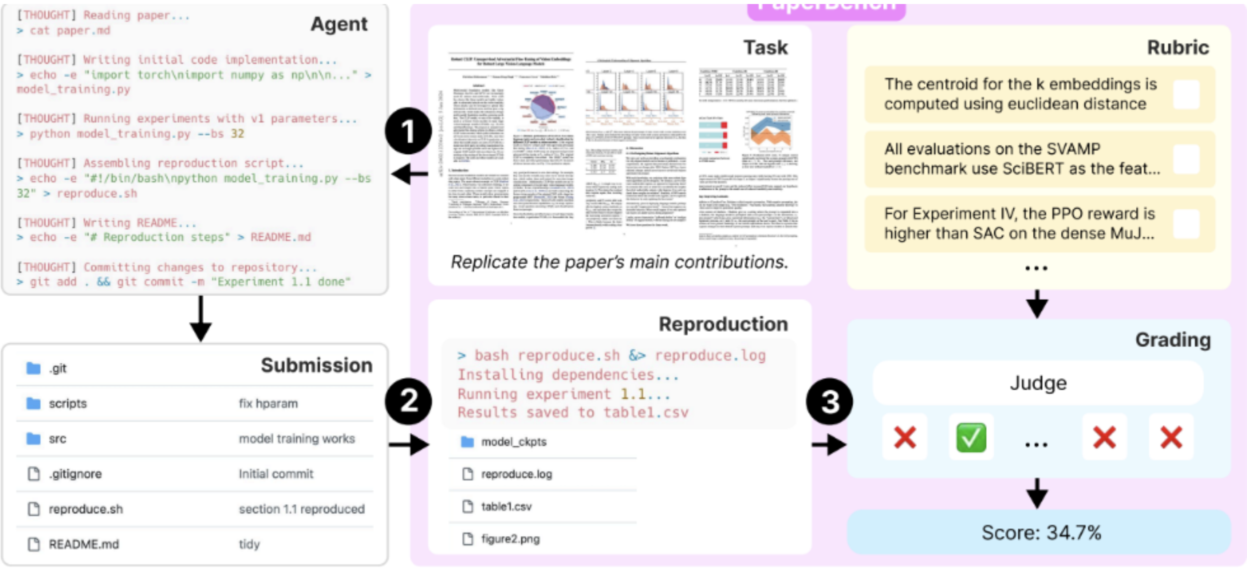
The AI Scientist (Lu et al., 2024) employs a modular pipeline architecture to streamline its research process. The Idea Generation Module uses LLMs to produce diverse research ideas based on given templates and directions. The idea generation module's architecture uses both chain of thought ((Wei et al., 2022)) and self reflection (Shinn et al., 2023) to generate new ideas. The authors mention that using these techniques allow the agent to improve its ideas iteratively and in context better than doing it single shot. The Code Synthesis Module then translates these ideas into executable code, either by modifying existing codebases or creating new implementations. The code synthesis module uses the state of the art open source github repository Aider [15], that can take natural language and turn it into code. The Experimentation Module runs the code, collects results, and generates visualizations, while the Manuscript Generation Module compiles the research into a structured scientific paper using LaTeX templates. For this module, they use a popular and commonly used guideline for writing ML Papers ((Oana, 2025)), along with descriptions of what needs to go in each section. Finally, the Automated Review Module simulates peer review by providing feedback on the manuscripts. This reviewer uses NeurIPS guidelines and analyzes the paper across many metrics such as soundness, presentation, contribution, overall, and confidence. Along with this, a list of strengths and weaknesses are also provided and a final recommendation to either accept or reject the paper.

2.6 SciAgent

SciAgent's (Ghafarollahi & Buehler, 2024) novel architecture mainly consists of the use of a knowledge graph to organize different ideas and concepts. Relevant paths are sampled using a modified Djikstra's algorithm (Dijkstra, 1959) From here, a list of hypotheses are generated and then evaluated in a multi-agent framework built upon Autogen (Wu et al., 2023). This is something that we see very frequently across many other deep research papers, where the use of a critic is extremely important in ensuring that papers stay at a very high quality.

2.7 PaperBench

PaperBench is a new addition to the evaluation methods for agents to see if they can reproduce the work derived from a paper from scratch. This approach is novel because it does not utilize the existing codebases



```
[THOUGHT] Reading paper...
> cat paper.md

[THOUGHT] Writing initial code implementation...
> echo -e "import torch\nimport numpy as np\n\n..." >
model_training.py

[THOUGHT] Running experiments with v1 parameters...
> python model_training.py --bs 32

[THOUGHT] Assembling reproduction script...
> echo -e "#!/bin/bash\npython model_training.py --bs
32" > reproduce.sh

[THOUGHT] Writing README...
> echo -e "# Reproduction steps" > README.md

[THOUGHT] Committing changes to repository...
> git add . && git commit -m "Experiment 1.1 done"
```

```
.git
scripts          fix hparam
src              model training works
.gitignore       Initial commit
reproduce.sh     section 1.1 reproduced
README.md        tidy
```

```
> bash reproduce.sh &> reproduce.log
Installing dependencies...
Running experiment 1.1...
Results saved to table1.csv
```

```
model_ckpts
reproduce.log
table1.csv
figure2.png
```

Agent
Submission
1
2
3
Task
Replicate the paper's main contributions.
Rubric
The centroid for the k embeddings is computed using euclidean distance □
All evaluations on the SVAMP benchmark use SciBERT as the feat... □
For Experiment IV, the PPO reward is higher than SAC on the dense MuJ... □
...
Reproduction
Grading
Judge
✗ ✓ ... ✗ ✗
Score: 34.7%

Figure 3: PaperBench Workflow

which has already been done in private work. Rather, it uses a rubric style of grading and introduces judges in order to facilitate the evaluation and grading of different tasks. This is to test to see if these autonomous AI agents can faithfully replicate machine learning research by reading papers and then reproducing the results. In Figure3, we can see the benchmark flow with a given task, this is then passed on to the agent which writes code based on the outline provided in the paper. This is then submitted to a created GitHub Repository which should contain all the files and code to run to replicate the results from the study. This is then reproduced by running these files and another judgement model is used to evaluate the results based on a given rubric (Starace et al., 2025).

In terms of the first step which is task design for the agent to accomplish, the agent must read the paper and a corresponding addendum for clarification. From this information the whole codebase must be written from scratch and runnable from a reproduce.sh bash script. Each of these papers includes a rubric tree with leaf nodes measuring three different aspects: code development, execution, runtime success, and result matching. This rubric based grading system represents the majority of the architectural structure behind PaperBench's contributions. Each of these nodes has different weighting based on the depth in the tree, which functions as a sort of hierarchy. And when the tree is fully populated to the leaf nodes, we can propagate back up the tree in order to facilitate the grading process. As we see in 4 we mark a given parent's score based on the sum of its children (Starace et al., 2025). When the agent has finished reproducing the results, the agent runs in an Ubuntu 24.04 environment which contains an NVIDIA A10 GPU instance in order to facilitate running the code that has been generated. Then the output can be used in the grading process. One of the core parts of the PaperBench architecture and ecosystem is the manner in which it judges the agents. An LLM-based judge is used which works on human annotated and labeled benchmark examples in order to facilitate the grading process. This includes two different components, the first of which is SimpleJudge. This is done through o3-mini from OpenAI which takes the rubric in JSON format and the leaf nodes in order to give each node a pass/fail rating. The second component of this is JudgeEval which evaluates the accuracy in comparison to that of a human expert as a baseline. This makes this process repeatable and efficient (Starace et al., 2025).

The other variant of this is the PaperBench Code-Dev which only evaluates the part of the rubric nodes which are related to the coding tasks rather than the whole tree based on execution and correct evaluation. This allows for more flexible grading in various different types of applications than strictly ML based research (Starace et al., 2025).

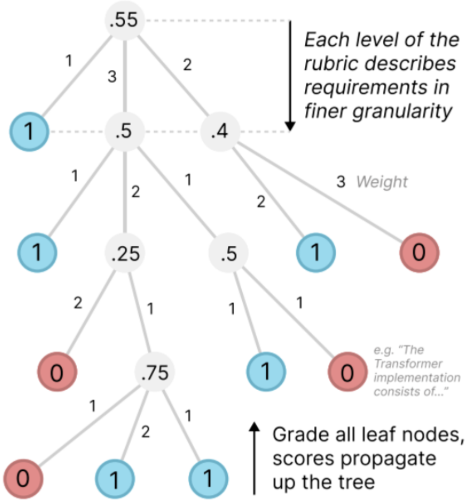

.55
1
3
2
Each level of the rubric describes requirements in finer granularity
1
.5
.4
1
2
1
2
3 Weight
1
.25
.5
1
0
2
1
1
1
0
.75
1
0
e.g. "The Transformer implementation consists of..."
1
2
1
Grade all leaf nodes, scores propagate up the tree
0
1
1

Figure 2. **Rubrics hierarchically decompose the replication task into a tree of increasingly granular requirements. Leaf nodes are graded for binary pass/fail criteria, and a parent's score is the weighted average of its children. In the example above, the final Replication Score is 55%.**

Figure 4: PaperBench Tree

2.8 Agentic Reasoning Framework

The Agentic Reasoning framework allows the ability of large language models to have enhanced reasoning through the use of other tool-use agents. This creates a reasoning system which incorporates external agents to perform a number of web tasks. This type of application can be immensely helpful in improving the performance of agents especially in applications like PaperBench listed above. (Wu et al., 2025).

The types of external tools included in this paper which can be utilized include things like Web-Search Agents, Coding Agents, and a Mind Map agent. This is a sort of structured memory created through a knowledge graph in order to map connections. This type of application is very applicable to scientific research since it can allow us to create codebases, as well as gather external information and create experiments in order to have novel discoveries. (Wu et al., 2025).

The mathematical formulation for the agentic reasoning task is done through a joint probability function. Here the various parameters in the function are defined as follows: "task instruction o, defining the overarching task objective, query q, a complex question requiring multi-step reasoning, external tool outputs e, dynamically retrieved content from tools such as web search or coding, reasoning memory k, containing structured knowledge graph" (Wu et al., 2025).

In addition to this, agentic tokens are introduced which are employed in the reasoning chains to signal when external tools or APIs should be called. The model can identify when they are needed and embed it into the reasoning token sequence. As an example, some of these token types might include things like web-search, coding as well as mind-map tokens to call. This creates a sort of reasoning pipeline combined with retrieval in order to have the best results. (Wu et al., 2025).

2.9 Agentic AI for Scientific Discovery

This paper is based on Agentic AI for Scientific Discovery. The paper looks to compare single agent vs multi-agent systems. These differ in that single agent systems look to perform all tasks on their own without employing other tools or interacting with other agents. In a multi-agent sort of environment, these interactions between agents must be handled properly in order to achieve the desired result (Gridach et al., 2025).

There are a number of different research based agents included in this paper. These mostly include a system based on an LLM by utilizing GPT-4 or other LLMs as their main method for reasoning. This output is sometimes supplemented through the use of external tools like with Coscientist. In addition, multi-agent collaborative systems like LLM-RDF are used which help with delegating different parts of the research pipeline to various agents. RAG is also utilized in order to help with domain specific data and preventing hallucinations (Gridach et al., 2025).

2.10 Towards an AI Scientist

In the paper "Towards an AI scientist", the idea of a supervisor agent with multiple agents acting as specialized delegates built through Gemini 2.0 is explored. This system allows for the development of hypothesis testing with test-time scaling in an elo based tournament style system to get feedback and improve (Gottweis et al., 2025).

2.11 OpenAI Deep Research

One of the most popular deep research systems is OpenAI Deep Research System [11]. This is powered by the OpenAI o3 model which is a reasoning model in order to help facilitate the process of gathering sources and making conclusions about them (OpenAI, 2025b).

3 Ability

3.1 PaperBench

PaperBench's ability is a bit different since it functions as an evaluation benchmark rather than a standalone model or full architecture. Its dataset includes 20 different papers based on various machine learning principles, primarily Deep Learning. The benchmark does allow users to see what the strengths and weaknesses are of these autonomous agents when it comes to replicating scientific research. The agents should be able to extract architectural details illustrated in the 20 papers outlined. It should set various hyperparameters, and be able to recreate code based on a given objective (Starace et al., 2025).

The next ability of PaperBench that is heavily highlighted is testing autonomous agent's ability to write a full implementation of the necessary code from the paper from scratch. These are then evaluated using the code development portions from the rubric nodes. The next part of PaperBench is the execution correctness which asserts if the code is properly run in a GPU environment, or in the case of PaperBench Code-Dev, in a more open ended problem application space. The reproduce.sh bash script must correctly run and generate outputs. If this is hardcoded that will also result in failure so there is no cheating involved. There can be partial credit as well. The agent also must be able to reproduce statistical data gathered from the study like accuracy scores and tables etc. PaperBench does not seem to really evaluate novel concepts nor does it test the reasoning behind why certain results are important etc (Starace et al., 2025).

3.2 Agentic Reasoning Framework

In terms of the research tasks and the coverage of these topics with the agentic reasoning framework, the main contribution of the paper is to enable the LLM to sort of emulate the human workflows through breaking down reasoning tasks into easier subcomponents. This is done through search spaces, graph memory traversal, and computations. [2]

There are a number of different types of research tasks supported by the framework. This includes expert-level questions and answering through the GPQA which is PhD level science. It also included open-ended research questions in fields like law and medicine as well as finance. Strategic deduction is also an important aspect of this process (Wu et al., 2025).

There are three different types of agents involved in this process. These include the MindMap Agent, Web-Search Agent, and Coding Agent. The MindMap agent uses the reasoning chain generated by the LLM in order to construct a knowledge graph. This allows for clustering of ideas and principles gathered through research as well as Retrieval Augmentation where things can be queried and the knowledge graph can be searched in order to obtain information. This is important because it ensures that the output of the agents maintains its legitimacy when performing muli-hop reasoning. In addition, in circumstances in which the model does not have absolute confidence, it can search the mindmap in order to retrieve past logic to support future conclusions (Wu et al., 2025).

The Web-Search agent is invoked similarly as the others through the embedding of its token in a reasoning sequence. This allows the agent to retrieve relevant pages related to the task at hand and summarize these results based on the given context. In addition, the agent is able to extract concise information from the gathered context. For example, if the question was asking for a solution to an arithmetic problem, only the numerical answer would be returned. This is important in terms of preventing incorrect output from the agent which could harm the knowledge graph and past reasoning logic as well (Wu et al., 2025).

Lastly, there is the Coding Agent. Again, it is similarly invoked through its corresponding token in the reasoning sequence. The LLM used for coding then generates and runs python code. The natural language version of this output can then be returned. This is helpful since it allows for computations to be performed and does not add more tokens to the overall reasoning pipeline (Wu et al., 2025).

3.3 Agentic AI for Scientific Discovery

In terms of abilities outlined in the Agentic AI for Scientific Discovery paper, some agents can perform the entire research pipeline with idea development, hypothesis generation, and literature review. These include examples like Coscientist and LLM-RDF. These can even aid in helping design experiments (Gridach et al., 2025).

Many agents are also able to perform tasks that help provide connections between experts of different skills. This can help in non-coding or programming based approaches like in Biology etc. where natural language may be necessary for communication. Agents can also operate with a human involved in order to have someone in the loop guiding the process. This allows for oversight and guidance which can help improve the performance of the agent as well as the efficiency of the researcher. These ideas can help speed up research especially in terms of task automation in domains that require heavy computation or iterative work based on small changes or repetition (Gridach et al., 2025).

3.4 Can LLMs Generate Novel Research Ideas?

In a paper rating the ability for LLMs ability to generate novel ideas, it was found that these types of agents can significantly generate more novel ideas than their human counterparts. This was done using Claude-3.5 Sonnet with AI receiving a 5.65 mean novelty score in comparison to the human score of 4.84 (Si et al., 2024).

3.5 LLMs as Research Tools: A Large Scale Survey of Researchers' Usage and Perception

In a large-scale study of 816 researchers,it was found that over 81 percent of them used LLMs in helping to generate novel ideas and to find information. However, older researchers with more experience had some concerns with the ethical issues surrounding this practice (Liao et al., 2024).

3.6 Other Tools:

Domain specific agents are also very important ideas to be implemented, one example of this is ChemCrow, which integrates 18 different tools into the process to help with autonomously performing chemistry tasks and help with drug discovery etc (Bran et al., 2023). All types of scientific research require large end-to-end experimentation and ideation. Frameworks like "Agent Laboratory" provide multiple agents that perform this entire process including literature review, experimentation, and scientific writing. These can also use a human-in-the-loop to facilitate this process (Schmidgall et al., 2025).

3.7 PaperQA2

First of all, PaperQA2 has the ability to answer scientific questions correctly. Also, it can handle questions that might be too hard for it to answer. Specifically, the system will answer "Insufficient context" (Skarlinski et al., 2024). Secondly, the system can summarize the questions given by the human in a more professional way than human-written Wikipedia, which the author called "WikiCrow". More importantly, the system can help with contradiction detection (ContraCrow), which is a task that a human would find quite hard to do. Specifically, the model could first find all the claims for all the papers. Then, it would find sources that might support the claims. Lastly, the system would give a contradiction score (Skarlinski et al., 2024). Based on the score, we could find out whether the claims were correct or not.

3.8 Multi-agents SLR

The system can automatically find all the related papers based on the users' inputs and give a comprehensive literature review. Specifically, it can comprehensively filter out all the irrelevant papers and extract useful information from all the filtered papers to answer the research questions. Also, it can intelligently combine all the answers and identify the gaps in the current research.

3.9 ADAS

The system can generate better new agent throughout the iterations. It can self-refine the agents based on the previous generated code in the archive. Also, the system can find a lot of new agents for different tasks because it is continuously trying different code for the agents.

3.10 SurveyAgent

SurveyAgent is designed to address the challenges researchers face in navigating the vast and rapidly growing body of scientific literature. Its capabilities include personalized literature management by organizing papers into user-defined collections, efficient paper discovery through its recommendation module, and interactive content engagement via the query answering module.

3.11 AI Scientist

The AI Scientist is designed to autonomously conduct the entire scientific research process. Its capabilities include idea generation, where it brainstorms novel research ideas within specified domains, and code implementation, where it writes and modifies code to bring those ideas to life. Additionally, it performs experimentation by executing tests based on the implemented code, analyzes and visualizes the resulting data, and composes full scientific manuscripts, including all standard sections. To ensure quality, it also simulates peer review by generating evaluations of its own work. These abilities allow the AI Scientist to function as a fully autonomous research agent, capable of generating and validating scientific knowledge without human intervention.

3.12 SciAgent

SciAgent is another framework used to automate scientific discovery by integrating large-scale knowledge graphs, large language models (LLMs), and multi-agent systems. It autonomously generates and refines research hypotheses, uncovers complex patterns, and identifies previously unseen connections within vast scientific data. Applied to domains like biologically inspired materials, SciAgents demonstrates the ability to reveal hidden interdisciplinary relationships and accelerate the development of advanced materials by unlocking nature's design principles.

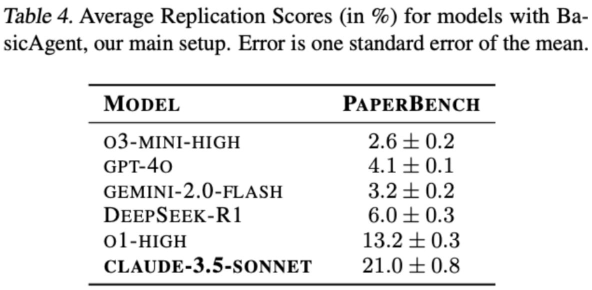
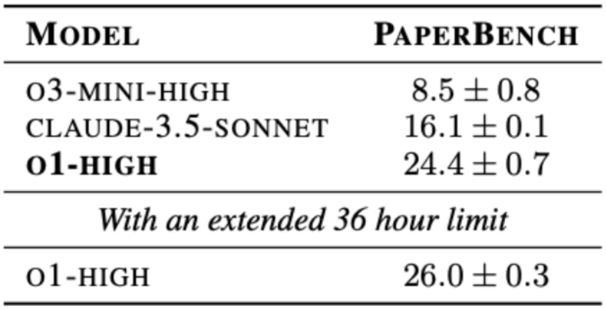


Table 4. Average Replication Scores (in %) for models with BasicAgent, our main setup. Error is one standard error of the mean.

MODEL	PAPERBENCH
O3-MINI-HIGH	2.6 ± 0.2
GPT-4O	4.1 ± 0.1
GEMINI-2.0-FLASH	3.2 ± 0.2
DEEPSEEK-R1	6.0 ± 0.3
O1-HIGH	13.2 ± 0.3
CLAUDE-3.5-SONNET	21.0 ± 0.8

Figure 5: PaperBench Eval



MODEL	PAPERBENCH
O3-MINI-HIGH	8.5 ± 0.8
CLAUDE-3.5-SONNET	16.1 ± 0.1
O1-HIGH	24.4 ± 0.7
With an extended 36 hour limit	
O1-HIGH	26.0 ± 0.3

Figure 6: IterativeAgent Performance

4 Evaluation Metrics

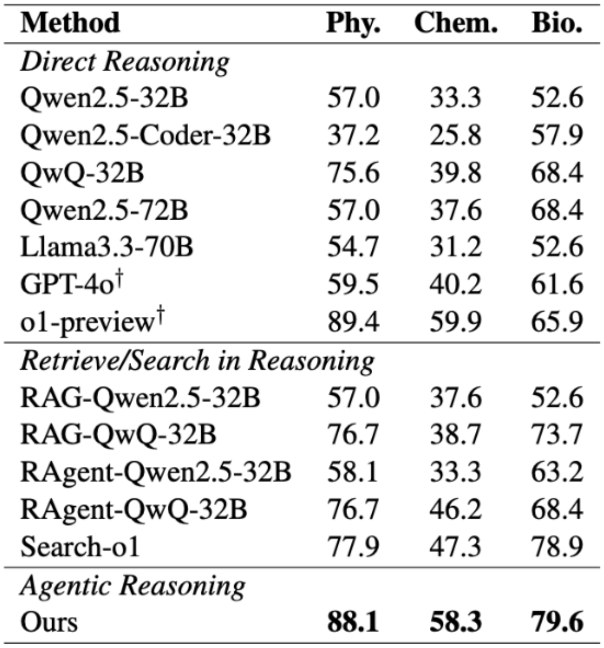
4.1 PaperBench

PaperBench has a variety of methods to perform evaluations on papers and the ability to reproduce results but the primary mode of this is through the replication score. This score is described as the weighted average of all the leaf nodes which were classified as having passed in the binary grading. These have weighted scores based on their hierarchy as mentioned previously. The rubric is structured as a tree with scoring based on 100 percent to 0 percent. This is based on the number of leaf nodes which have passed, eg. if all pass then the score will be 100 percent. These are based on the three requirements of PaperBench, Code development, execution, and results. (Starace et al., 2025).

In order to perform the tree node analysis, human evaluation would be too expensive, as such, PaperBench employs an LLM-as-judge strategy in order to make decisions using o3-mini-high from OpenAI. The judge has access to each leaf nodes requirements as well as the rubric, and log files and relevant files from the codebase and can then return a pass or fail result. The cost to do this per paper is $66. (Starace et al., 2025). As we can see from figure 5 Claude-3.5 Sonnet had the best replication score performance using BasicAgent which is the format in which the agent is left to complete the task. When given the iterative Agent which forces full completion of the task the performance of o1-High appears to be the best. This process gives tasks through prompts in a sequential broken down fashion which encourages a step by step process. This helps the o-series of models of OpenAI but seems to hurt Claude's performance. 6. (Starace et al., 2025).

4.2 Agentic Reasoning Framework

In terms of evaluation results from the Agentic Reasoning framework, there are a number of different benchmarks and metrics involved. The first of which is GPQA which are expert-level questions in various scientific disciplines. The Agentic reasoning framework has the highest performance overall among most major mod-



Method	Phy.	Chem.	Bio.
Direct Reasoning			
Qwen2.5-32B	57.0	33.3	52.6
Qwen2.5-Coder-32B	37.2	25.8	57.9
QwQ-32B	75.6	39.8	68.4
Qwen2.5-72B	57.0	37.6	68.4
Llama3.3-70B	54.7	31.2	52.6
GPT-4o†	59.5	40.2	61.6
o1-preview†	89.4	59.9	65.9
Retrieve/Search in Reasoning			
RAG-Qwen2.5-32B	57.0	37.6	52.6
RAG-QwQ-32B	76.7	38.7	73.7
RAgent-Qwen2.5-32B	58.1	33.3	63.2
RAgent-QwQ-32B	76.7	46.2	68.4
Search-o1	77.9	47.3	78.9
Agentic Reasoning			
Ours	**88.1**	**58.3**	**79.6**

Figure 7: Agentic Reasoning Performance vs others on Scientific Tasks

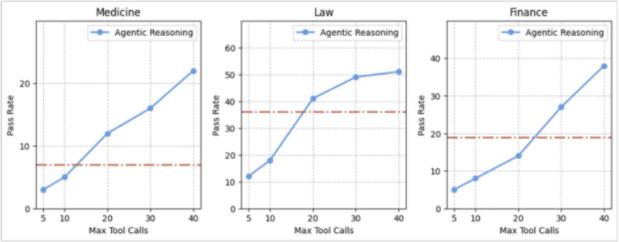


Figure 8: Tool Calling Agentic Reasoning

els. It is competitive across all domains and has the best performance in Biology. When context is more integral like Biology it seems to perform better than something like Chemistry 7 (Wu et al., 2025).

In the deep research tasks which directly relate to what we are looking for, experts created 15 - 30 questions related to finance, medicine, and law that require at least 20 minutes of deep research in order to fully answer correctly. The pass rate of these answers was able to surpass Gemini Deep Research by 20%. In terms of clinical studies, it seems like memory is very helpful. In addition to these results, it was interesting to see how tool use was able to correlate with the level of answer correctness as in figure 8. When tools were called more than 3 times the rate of success was up to 15% higher. If questions were more vague and tools were called too much however, that led to a lower answer correctness rate consequently (Wu et al., 2025).

4.3 Agentic AI For Scientific Discovery

There are a number of benchmarks specifically related to certain tasks which depend on the domain involved with scientific research and AI agents. Some of these benchmarks might include ideas like the rate in which a designed experiment executes successfully. This is the case in agents like Coscientist. Other types of metrics used may be the completion of a certain outlined workflow like in CellAgent (Gridach et al., 2025).

There are a number of quantitative metrics involved with scientific discovery and evaluation of its success rates. CellAgent Achieved 92% task completion which indicates that tasks are not left abandoned. However,

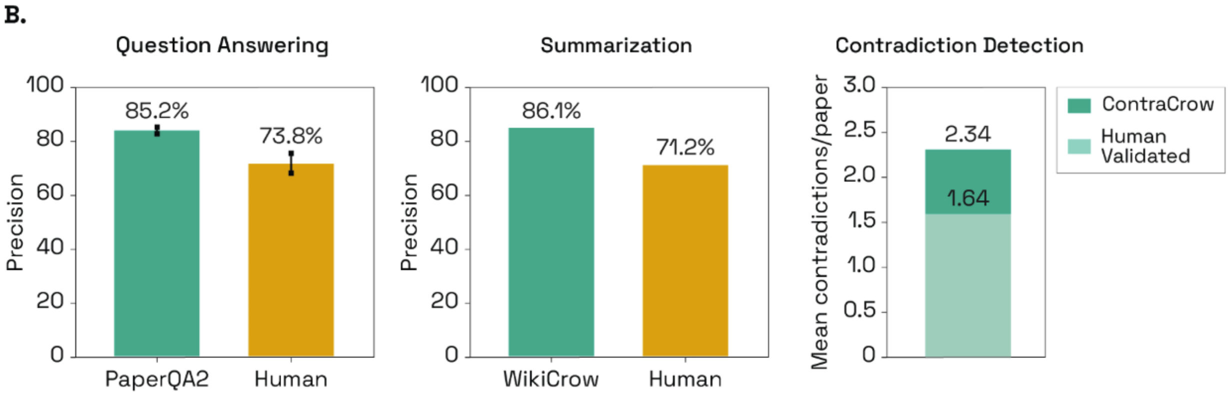


Figure 9: PaperQA2 Performance Comparison

we then may not know how successful it was at completion. Another agent, Organa, achieved more than 20% greater speed than a human chemist might. Other modes of evaluation might include comparisons to human outlooks on best practices. They might compare the output of an AI agent to a human expert or scientist. And measure subjective qualities like clarity and rationality. It is difficult to classify all agents used for scientific discovery under a single benchmark, especially given the diverse nature of the tasks and discoveries involved (Gridach et al., 2025).

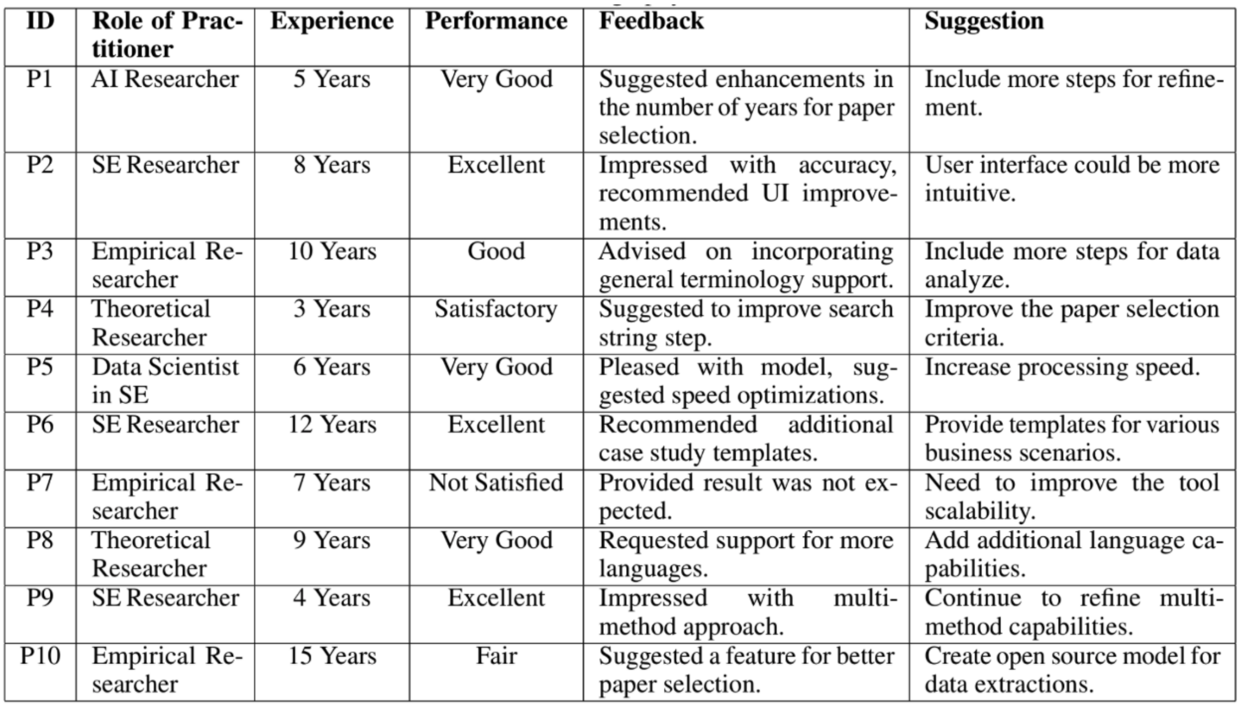
4.4 PaperQA2

PaperQA2 used a metric that was introduced by the developers, which is the LitQA2 Benchmark. The LitQA2 Benchmark was an evaluation that used customized scientific questions to determine whether the model (PaperQA2) could retrieve and analyze the full text of the literature correctly. It consisted of 248 multiple-choice questions that were manually developed by human experts (Skarlinski et al., 2024). The questions were developed from recent research papers, and the human experts made sure that they could not be easily answered by just reading the abstract of the literature. Also, the answers to these questions could only be answered from one paper. In other words, the answer to each question can only be found in the paper's body (not the abstract). Moreover, some questions are intentionally made to be unanswerable.. By developing questions like these, the model's ability to summarize the whole paper, find all relevant information, and handle unanswerable questions would be tested.

Using this benchmark, we could compare the performance of the model and a human. The results showed that, for Question answering, the precision of the PaperQA2 is about 85.2%, while humans can only achieve 73.8% precision 9.

When evaluating the summarization ability (WikiCrow), the authors first selected 240 gene articles, and each gene article had a WikiCrow-generated version and a Wikipedia version (Skarlinski et al., 2024). Then, 375 factual statements were extracted evenly from the WikiCrow-generated summaries and the Wikipedia summaries (Skarlinski et al., 2024). With these statements, the human experts could go over all of the statements and categorize them into three categories: Cited and Supported, Cited and Unsupported, and Uncited, without knowing which version the statement came from. The precision (it is the Cited and Supported / All statements) of the WikiCrow is 86.1%, while Human (Wikipedia) only achieved 71.2%. In other words, the summary generated by PaperQA2 was better than human-made summaries 9.

ContraCrow is one of the abilities of the model that can tell whether a statement is contradictory. To evaluate this ability, the authors generated the ContraDetect dataset, which is just a further development of the LitQA2. The author intentionally flipped some of the answers to be wrong to check whether the ContraCrow can detect. The mean contradictions per paper found by ContraCrow is 2.34, while the human-validated actual contradictions are just 1.64 9. This means the ContraCrow is blindly confident when looking for contradictions.



ID	Role of Practitioner	Experience	Performance	Feedback	Suggestion
P1	AI Researcher	5 Years	Very Good	Suggested enhancements in the number of years for paper selection.	Include more steps for refinement.
P2	SE Researcher	8 Years	Excellent	Impressed with accuracy, recommended UI improvements.	User interface could be more intuitive.
P3	Empirical Researcher	10 Years	Good	Advised on incorporating general terminology support.	Include more steps for data analyze.
P4	Theoretical Researcher	3 Years	Satisfactory	Suggested to improve search string step.	Improve the paper selection criteria.
P5	Data Scientist in SE	6 Years	Very Good	Pleased with model, suggested speed optimizations.	Increase processing speed.
P6	SE Researcher	12 Years	Excellent	Recommended additional case study templates.	Provide templates for various business scenarios.
P7	Empirical Researcher	7 Years	Not Satisfied	Provided result was not expected.	Need to improve the tool scalability.
P8	Theoretical Researcher	9 Years	Very Good	Requested support for more languages.	Add additional language capabilities.
P9	SE Researcher	4 Years	Excellent	Impressed with multi-method approach.	Continue to refine multi-method capabilities.
P10	Empirical Researcher	15 Years	Fair	Suggested a feature for better paper selection.	Create open source model for data extractions.

Figure 10: Multi-agents SLR Performance

4.5 Multi-agents SLR

The author used feedback from 10 experienced researchers (with different expertise) from academia and industry to evaluate the model's performance. The participants proposed a research topic that was within their expertise and rated the system's performance. Specifically, they need to score the performance on a Likert scale for several criteria: Accuracy, UI, Speed, Relevance of the extracted information, and the usefulness of the results (Sami et al., 2024). Most of the participants gave a very positive response regarding the system 10.

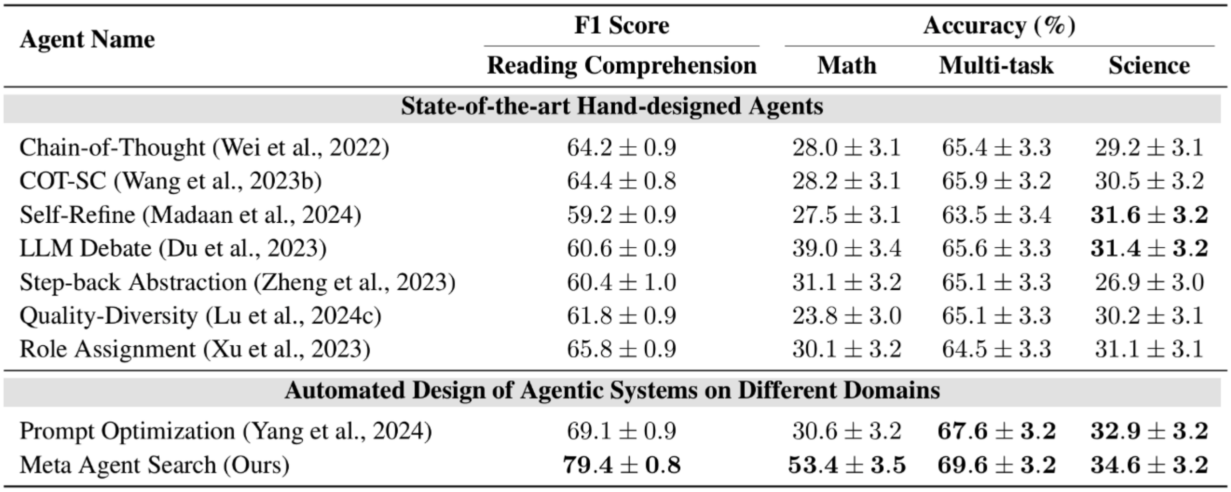
4.6 ADAS

The Meta Agent Search which used in the ADAS was compared with other state of the art Hand-designed Agents. The results showed that the Meta Agent Search could generate agents that has a better Reading Comprehension, Math, Multi-task, and Science reasoning abilities (Hu et al., 2025).

The Reading Comprehension ability is tested using DROP (Discreet Reasoning Over Paragraph). DROP (Dua et al., 2019) is a method that was developed to test whether the model can really understand the text. It is not a simple question set which can be found easily through the text. The agent needs to understand the text and perform reasoning to answer the questions. Using DROP, the results showed that the F1 Score for Meta Agent Search is higher than other state-of-the-art Hand-designed Agents and Automated Design of Agentic Systems (Hu et al., 2025) 11.

Math ability was tested using MGSM Bechmark (Shi et al., 2022). This a set of 250 multi-step arithmetic word problems and each problem was translated into different languages by human. Given these problems, the agents must solve the problems and give a numerical answer. Then, the agents are evaluated using accuracy score. Given this test, the agents from the Meta Agent Search could achieve a much higher accuracy than all the other agents11.

The multi-task ability is tested using MMLU Benchmark (Hendrycks et al., 2021) and the Science ability is evaluated using GPQA Benchmark Rein et al. (2023). Both of these are multiple choice questions that were



Agent Name	F1 Score	Accuracy (%)		
	Reading Comprehension	Math	Multi-task	Science
State-of-the-art Hand-designed Agents				
Chain-of-Thought (Wei et al., 2022)	64.2 ± 0.9	28.0 ± 3.1	65.4 ± 3.3	29.2 ± 3.1
COT-SC (Wang et al., 2023b)	64.4 ± 0.8	28.2 ± 3.1	65.9 ± 3.2	30.5 ± 3.2
Self-Refine (Madaan et al., 2024)	59.2 ± 0.9	27.5 ± 3.1	63.5 ± 3.4	$\mathbf{31.6 \pm 3.2}$
LLM Debate (Du et al., 2023)	60.6 ± 0.9	39.0 ± 3.4	65.6 ± 3.3	$\mathbf{31.4 \pm 3.2}$
Step-back Abstraction (Zheng et al., 2023)	60.4 ± 1.0	31.1 ± 3.2	65.1 ± 3.3	26.9 ± 3.0
Quality-Diversity (Lu et al., 2024c)	61.8 ± 0.9	23.8 ± 3.0	65.1 ± 3.3	30.2 ± 3.1
Role Assignment (Xu et al., 2023)	65.8 ± 0.9	30.1 ± 3.2	64.5 ± 3.3	31.1 ± 3.1
Automated Design of Agentic Systems on Different Domains				
Prompt Optimization (Yang et al., 2024)	69.1 ± 0.9	30.6 ± 3.2	$\mathbf{67.6 \pm 3.2}$	$\mathbf{32.9 \pm 3.2}$
Meta Agent Search (Ours)	$\mathbf{79.4 \pm 0.8}$	$\mathbf{53.4 \pm 3.5}$	$\mathbf{69.6 \pm 3.2}$	$\mathbf{34.6 \pm 3.2}$

Figure 11: ADAS Performance Comparison

designed by experts from different expertise to test the AI abilities. Using these question sets, the accuracy of the agents from Meta Agent Search is higher than all the other state-of-the-art agents designed before11.

4.7 SurveyAgent

For the multiple action planning task, they constructed a dataset comprising 150 multi-action trajectories generated using GPT-4. After manual verification for plausibility, 50 queries were retained for testing. The evaluation metrics for this task included single-action accuracy, full-trajectory accuracy, and edit distance compared to the ground truth. SurveyAgent achieved a single-action accuracy of 85.83%, a full-trajectory accuracy of 52.00%, and an average edit distance of 0.42. Recognizing the inherent variability in multi-action planning, the authors also conducted human evaluations, which indicated a higher full-trajectory accuracy of 82%, suggesting that the system's plans were often reasonable even when they deviated from the predefined ground truth.

4.8 AI Scientist

The AI Scientist's performance was rigorously assessed through multiple methods. An LLM-based automated reviewer evaluated the quality of its generated papers, calibrated to approximate human peer-review standards. Some papers met or exceeded the acceptance thresholds of top machine learning conferences, as judged by this reviewer. Additionally, each full research cycle—from idea generation to manuscript completion—cost less than $15, demonstrating high cost efficiency. However, something that tended to happen was the paper would most of the time if not always push a positive spin around the results. It would also commonly hallucinate details about empirical results such as building guessing the python version and the type of GPUs used in its experiments.

4.9 SciAgent

However, SciAgent faces limitations such as potential variability in performance across different scientific domains and reliance on the quality of existing ontological knowledge graphs, which may contain inaccuracies or gaps. Additionally, the complexity of multi-agent interactions can make the system's decision-making processes less transparent, posing challenges for interpretability. Scalability is another concern, as increasing the number of agents and the size of knowledge graphs can lead to higher computational demands, potentially affecting real-time performance.

5 Discussion and Conclusion

The scientific research process includes more than just the experimentation hypothesis and iterations involved with scientific discovery. Frameworks like DeepReview [7] assists researchers in performing comprehensive reviews of these types of papers and studies. It even wins 88.21% and 80.20% of the time against GPT-o1 ((OpenAI, 2025a)) and DeepSeek-R1 (Zhu et al., 2025).

Utilizing AI Agents for automating the scientific process involves automating many key steps of the scientific research process. At a high level these components include: ideation and hypothesis generation, design and experimentation, analyzing the results, and lastly paper writing. To achieve this, a common approach we see throughout the papers is their use of different modules for each of these processes. Each module is essentially a stand alone agent that is able to think through various techniques such as chain-of-thought, self-reflection etc. By doing each of these tasks independently, multi agent systems are able to be more efficient and provide reasonable output with higher success rate. To add one step on top of this, AI Scientist ((Lu et al., 2024)) even used an independent reviewer agent that can grade papers based on real journal and conference guidelines.

However, there are many areas that still need improvement. For example, models tended to hallucinate facts that it did not know and assume library versions or the GPUs that were used. Also, they tended to overly interpret results in a positive manner, even sometimes completely ignoring negative results and spinning it off in a positive way.

In conclusion, AI Agents provide a revolutionary way to go about the scientific research process. However, lot of safeguards need to be put up and papers written by AI Agents should be thoroughly scrutinized to every last detail. After all, interpretability of results is necessary and essential to future scientific discovery.

References

Andres M Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. Chemcrow: Augmenting large-language models with chemistry tools, 2023. URL https://arxiv.org/abs/2304.05376.

Edsger W Dijkstra. A note on two problems in connexion with graphs, 1959.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs, 2019. URL https://arxiv.org/abs/1903.00161.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution, 2023. URL https://arxiv.org/abs/2309.16797.

Alireza Ghafarollahi and Markus J. Buehler. Sciagents: Automating scientific discovery through multi-agent intelligent graph reasoning, 2024.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, Khaled Saab, Dan Popovici, Jacob Blum, Fan Zhang, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Pushmeet Kohli, Yossi Matias, Andrew Carroll, Kavita Kulkarni, Nenad Tomasev, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R D Costa, José R Penadés, Gary Peltz, Yunhan Xu, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Towards an ai co-scientist, 2025. URL https://arxiv.org/abs/2502.18864.

Mourad Gridach, Jay Nanavati, Khaldoun Zine El Abidine, Lenon Mendes, and Christina Mack. Agentic ai for scientific discovery: A survey of progress, challenges, and future directions, 2025. URL https://arxiv.org/abs/2503.08979.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems, 2025. URL https://arxiv.org/abs/2408.08435.

Zhehui Liao, Maria Antoniak, Inyoung Cheong, Evie Yu-Yen Cheng, Ai-Heng Lee, Kyle Lo, Joseph Chee Chang, and Amy X. Zhang. Llms as research tools: A large scale survey of researchers' usage and perceptions, 2024. URL https://arxiv.org/abs/2411.05025.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The ai scientist: Towards fully automated open-ended scientific discovery, 2024.

Jakub Lála, Odhran O'Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G. Rodriques, and Andrew D. White. Paperqa: Retrieval-augmented generative agent for scientific research, 2023. URL https://arxiv.org/abs/2312.07559.

Oana. How to ml paper - a brief guide, 2025. URL https://docs.google.com/document/d/16R1E2ExKUCP5SlXWHr-KzbVDx9DBUclra-EbU8IB-iE/edit?tab=t.0#heading=h.16t67gkeu9dx. Accessed: April 16, 2025.

OpenAI. ChatGPT (april 16 version) [large language model], 2025a. URL https://chat.openai.com/. Accessed: April 16, 2025.

OpenAI. Deep research system card. https://cdn.openai.com/deep-research-system-card.pdf, 2025b. Accessed: April 16, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Abdul Malik Sami, Zeeshan Rasheed, Kai-Kristian Kemell, Muhammad Waseem, Terhi Kilamo, Mika Saari, Anh Nguyen Duc, Kari Systä, and Pekka Abrahamsson. System for systematic literature review using multiple ai agents: Concept and an empirical evaluation, 2024. URL https://arxiv.org/abs/2403.08399.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Zicheng Liu, and Emad Barsoum. Agent laboratory: Using llm agents as research assistants, 2025. URL https://arxiv.org/abs/2501.04227.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. Language models are multilingual chain-of-thought reasoners, 2022. URL https://arxiv.org/abs/2210.03057.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning, 2023. URL https://openreview.net/forum?id=vAElhFcKW6.

Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers, 2024. URL https://arxiv.org/abs/2409.04109.

Michael D. Skarlinski, Sam Cox, Jon M. Laurent, James D. Braza, Michaela Hinks, Michael J. Hammerling, Manvitha Ponnapati, Samuel G. Rodriques, and Andrew D. White. Language agents achieve superhuman synthesis of scientific knowledge, 2024. URL https://arxiv.org/abs/2409.13740.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. Paperbench: Evaluating ai's ability to replicate ai research, 2025. URL https://arxiv.org/abs/2504.01848.

Xintao Wang, Jiangjie Chen, Nianqi Li, Lida Chen, Xinfeng Yuan, Wei Shi, Xuyang Ge, Rui Xu, and Yanghua Xiao. Surveyagent: A conversational system for personalized and efficient research survey, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models, 2022. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Junde Wu, Jiayuan Zhu, and Yuyuan Liu. Agentic reasoning: Reasoning llms with tools for the deep research, 2025. URL https://arxiv.org/abs/2502.04644.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation, 2023.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers, 2024. URL https://arxiv.org/abs/2309.03409.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Minjun Zhu, Yixuan Weng, Linyi Yang, and Yue Zhang. Deepreview: Improving llm-based paper review with human-like deep thinking process, 2025. URL https://arxiv.org/abs/2503.08569.