Bootstrapping AI Research with AI: Imitate-and-Evolve for Long-Horizon Planning in Paper Generation

ANONYMOUS AUTHOR(S)

TODO.

CCS Concepts: • **Computing methodologies** → **Multi-agent planning**; **Natural language generation**.

Additional Key Words and Phrases: Imitate-and-Evolve, Long-Horizon Planning, Research Paper Generation

ACM Reference Format:

Anonymous Author(s). 2018. Bootstrapping AI Research with AI: Imitate-and-Evolve for Long-Horizon Planning in Paper Generation. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, **??** pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Artificial Intelligence (AI) increasingly influences the landscape of scientific research [**?** **?**]. Automating general scientific discovery has been a long-standing ambition of the research community, dating back to the 1970s–1980s [**?**] with the advent of computer science, and, in AI, to early visions of automating science with AI itself [**?**]. Yet, for decades, day-to-day research has remained highly manual: researchers must read and synthesize overwhelming amounts of literature to formulate ideas, and then design and execute experiments to validate them [**?**]. Many promising ideas are thus buried under the rising costs of evidence gathering and verification.

Recent large language models (LLMs) [**?**] have made impressive progress in processing, organizing, and generating scientific text, enabling AI systems to assist across the full "idea generation—literature synthesis—experimental validation—manuscript writing" pipeline [**?**]. A growing body of work uses LLMs to propose research ideas [**?** **?** **?**], to assist in running or orchestrating experiments [**?**], or to act as AI scientists that generate open-ended scientific publications [**?** **?**]. Going further, end-to-end frameworks now close the loop by integrating ideation, code authoring, experiment execution, writing, and simulated peer review, making it possible to produce reviewable papers from scratch [**?**].

Despite these advances, fully automated research still faces a structural challenge: writing a research paper is a long-horizon process that requires multi-stage reasoning—from formulating testable hypotheses, to systematically aligning with the literature, to designing and revising feasible experiments. In practice, even expert researchers rarely "plan once and execute to the end." Instead, they proceed non-linearly—tracking frontier results, probing prototypes, and continuously re-aligning ideas with evidence. We refer to the failure mode of static, one-shot plans as *evidence drift*: scientific arguments must keep methods aligned with evidence, but static plans cannot absorb new facts surfaced by literature checks, pilot runs, or metric fluctuations. The result is a misalignment among method, evidence, and conclusion—manifesting as logical jumps between design and claims, mismatches between citations and implementations, or narrative statements that outrun their empirical support. Under this lens, static planning tends to enter a no-win

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.

Manuscript submitted to ACM

corner: if one "reaches forward" for novelty without mid-course calibration, feasibility collapses; if one "steps back" for feasibility with early, conservative pruning, novelty collapses. These are not mere trade-offs by choice, but two facets of the same misalignment: a plan and its evolving evidence remain out of sync for too long, and the deviation becomes irrecoverable downstream.

This observation motivates a different question: can we generate expert-level papers that are both novel and feasible by treating planning as a policy constrained by evidence, rather than a fixed script? Concretely, the goal is not to "revise frequently" per se, but to convert weak, incremental signals into actionable small steps, to front-load calibration and error correction into the process, and to treat each step's inputs and outputs as state variables under control—thereby preserving soundness while pushing the frontier of innovation.

We propose an Imitation-and-Evolve approach that operationalizes this principle in both writing and revision cycles. During writing, given a research theme, we first identify its algorithmic paradigm and task lineage, then retrieve frontier papers via academic graphs and entity-centric knowledge, and construct an initial plan in the "candidate algorithm × target task" space. Instead of "plan once, execute to the end," we employ a dynamic re-planning loop: before each planned step, we retrieve evidence snippets from relevant documents to test feasibility; if the evidence is weak or conflicting, we revise that step or switch techniques; if the evidence supports, we execute the step, then parse execution logs and outputs to decide whether to re-plan subsequent steps. Thus, each step proceeds in a closed loop of "evidence retrieval—feasibility check—result review—re-plan," which suppresses evidence drift and prevents error cascades.

During revision, we again "imitate and evolve." We retrieve public reviews from papers on similar topics, align their human criteria (clarity, validity, novelty, reproducibility, etc.), and distill them into executable edit instructions. We then run a fine-grained "self-review—LLM reviewer—action" cycle: each review point is decomposed into atomic edits; the system decides to revise or to rebut based on current content; and only if the post-edit score, measured against human-aligned criteria, improves significantly do we accept the change—otherwise we roll back. This gated loop from "review signal → action plan → effect measurement" avoids blind compliance and prevents over-editing, ensuring that every change yields real quality gains.

In summary, our contributions are threefold. First, by matching and recombining "algorithm × task" under explicit evidence constraints, we expand the constructive search space and enable combinational innovation without sacrificing feasibility—resolving the aforementioned no-win corner. Second, by triggering dynamic re-planning through targeted evidence retrieval and result parsing, we decompose the long-horizon paper-generation problem into verifiable short-horizon decisions, continuously re-aligning plan, evidence, and conclusion to mitigate evidence drift. Third, by anchoring revision to human-aligned review criteria and enforcing a "revise-then-re-evaluate" gate, we convert external feedback into executable edits and guarantee measurable improvements rather than late-stage, costly overhauls. Compared to pipeline-style, fixed plans, our policy-like process better mirrors real scientific practice: it advances through uncertainty and feedback, and it evolves through alignment with evidence and standards—thereby increasing the likelihood of producing research outputs that are both genuinely novel and methodologically sound.

2 Related Work

2.1 LLMs for Research

In recent years, several studies have explored using language models for creative tasks in research, such as multi-agent collaborative writing [**?**] and multi-module retrieval [**?**] to improve research idea generation. These works aim to boost the novelty and diversity of AI in creative tasks. Si et al. [**?**] conducted a comprehensive human evaluation of the task

of idea generation by language models. Wang et al. [**?**] proposed using LLMs to automatically write survey papers. Additionally, LLMs have been used to automate the research process: Huang et al. [**?**] introduced a benchmark for evaluating LLMs in coding solutions for machine learning problems; Wang et al. [**?**] proposed a method leveraging LLMs for scientific literature retrieval. The AI Scientist project [**?**] introduced a fully automated, promptdriven research pipeline. The following work, AI Scientist-v2 project [**?**] further introduced a workshop-level automated scientific discovery via agentic tree search. More recently, Weng et al. [**?**] developed an iterative self-rewarding framework that enables the LLM to refine its ideas continuously, enhancing both diversity and practicality in research proposal generation. Guo et al. [**?**] proposed a benchmark for evaluating LLMs in research idea generation. Pu et al. [**?**] proposed a method for iterative research idea development through evolving and composing idea facets with literature-grounded feedback. Garikaparthi et al. [**?**] proposed a method for interactive research ideation system for accelerating scientific discovery.

Existing work have made progress in research paper generation. However, they rely on static planning, challenges of.

2.2 Planning of LLM Agents

LLMs have achievedremarkable success across various domains, showcasing sig-nificant intelligence in reasoning, tool usage, planning, andinstruction-following. The surprising intelligence of LLMssheds light on employing LLMs as the cognitive core ofagents, thereby offering the potential to improve planning ability [**?**]. For example, Plan-and-Solve [**?**] propose a two-step prompt instruction: “Let’s first de-vise a plan” and “Let’s carry out the plan”. This zero-shot approach has achieved improvements in mathematical rea-soning, common-sense reasoning, and symbolic reasoning. ProgPrompt [**?**] translates natural languagedescriptions of tasks into coding problems. It symbolizes theagent’s action space and objects in the environment throughcode, with each action formalized as a function and each ob-ject represented as a variable. Plan-Act [**?**] for HTML manipulation . LLMCompiler [**?**] break down QA tasks into parallel execution graphs.

Despite promising contributions in each direction,it is seen that LLMs still fall short in more complexscenarios, i.e., long-horizon planning [**?**]. There are several hierarchical planning frameworks. AgentOccam [**?**] incorporates planning into the action space with tree-like planning, WebPilot [**?**] uses six different agents, AdaPlanner [**?**] employs InPlan and Out-of-Plan Refiners for replanning, and ADaPT [**?**] uses recursive decomposition. PLAN-AND-ACT [**?**] provides a simpler two-agent framework with a systematic approach to generating high-quality training data for open-source LLMs.

However, lack of study on long-horizon planning for LLMs for research.

3 Method

In this section, we introduce our Imitation-and-Evolve approach in detail. The overall framework is shown in Figure **??**.

3.1 Imitation-and-Evolve for Drafting

Given a user request U describing a research theme, our drafting process begins with intent detection to identify the desired algorithmic paradigm U_A and target task U_T. This decomposition enables targeted retrieval and matching in the "algorithm × task" space. We then retrieve relevant academic papers through a two-stage process: semantic similarity search followed by reranking. This yields algorithm-related papers $L_A = \{L_A^1, L_A^2, \cdots, L_A^n\}$ and task-related papers $L_T = \{L_T^1, L_T^2, \cdots, L_T^m\}$. To identify the most promising combination, we employ an LLM to evaluate all pairwise matches and select the optimal pair (M_A, M_T) where $M_A \in L_A$ and $M_T \in L_T$ based on their complementarity and innovation potential.

From the selected papers, we extract structural outlines O_A from M_A and O_T from M_T. By imitating and recombining these outlines, we construct an initial paper outline O that inherits the methodological rigor from O_A and the task-specific structure from O_T. The core of our approach lies in dynamic task execution with evidence-driven replanning. We decompose the outline O into an initial task list $P^{(0)} = \{P_1^{(0)}, P_2^{(0)}, \cdots, P_k^{(0)}\}$, where each $P_j^{(0)}$ represents a writing task for a specific section or subsection, and k is the total number of tasks.

For each step $i \in \{1, 2, \cdots, k\}$, the execution follows a closed-loop process. Before executing task $P_i^{(i-1)}$, we deploy an evidence retrieval agent that searches across three sources: the current draft D_{i-1}, downloaded paper content from $L_A \cup L_T$, and publicly available academic databases. The agent analyzes $P_i^{(i-1)}$ to determine relevant sources and extracts evidence snippets $E_i = \{e_1, e_2, \cdots, e_r\}$. Based on E_i, we assess the feasibility of $P_i^{(i-1)}$. If the evidence suggests infeasibility or conflicts, we revise the current task with minimal modifications to obtain $P_i^{(i)} = \mathcal{R}(P_i^{(i-1)}, E_i)$, where $\mathcal{R}$ is the revision function guided by evidence. Otherwise, we proceed with $P_i^{(i)} = P_i^{(i-1)}$.

A writing agent then generates the section content based on the refined task: $W_i = \mathcal{W}(P_i^{(i)}, E_i, D_{i-1})$, where $\mathcal{W}$ is the writing function that incorporates evidence and maintains consistency with the existing draft. The updated draft becomes $D_i = D_{i-1} \cup W_i$. After executing task i, we parse the results and assess whether downstream tasks need adjustment. The remaining tasks are updated as:

$$\{P_{i+1}^{(i)}, P_{i+2}^{(i)}, \cdots, P_k^{(i)}\} = \mathcal{U}(\{P_{i+1}^{(i-1)}, P_{i+2}^{(i-1)}, \cdots, P_k^{(i-1)}\}, W_i, E_i) \tag{1}$$

where $\mathcal{U}$ is the update function that propagates changes based on execution results and new evidence. Note that for $j \leq i$, task P_j has already been executed and remains unchanged in subsequent iterations.

This iterative process continues until all k tasks are completed, yielding the final draft D_k. At iteration i, we maintain the task list state $P^{(i)} = \{P_1^{(1)}, P_2^{(2)}, \cdots, P_i^{(i)}, P_{i+1}^{(i)}, \cdots, P_k^{(i)}\}$, where tasks 1 through i have been executed with their final versions, and tasks $i+1$ through k represent the most recent planning based on accumulated evidence and execution results. By treating each step as a state transition conditioned on evidence, our approach maintains alignment between plan and execution throughout the long-horizon writing process. The dynamic replanning mechanism ensures that evidence drift is continuously corrected, preventing the accumulation of errors that would otherwise cascade through static plans. This policy-like approach enables the system to navigate the trade-off between novelty and feasibility by making incremental, evidence-grounded decisions at each step.

3.2 Imitation-and-Evolve for Revision

After obtaining all section drafts $D_1, D_2, \ldots, D_k$, we employ an evaluation agent to assess the quality of the complete manuscript. We imitate human expert evaluation by conducting peer review on the draft. Following standard peer review protocols, we prompt the LLM to generate review feedback across multiple dimensions: summary, soundness, presentation, contribution, strengths, weaknesses, questions, rating, and meta-review results. This yields the first round of review feedback $R^{(1)} = \{r_{\text{summary}}, r_{\text{soundness}}, \ldots, r_{\text{meta}}\}$ along with a corresponding quality score $Q^{(1)}$. We then decompose $R^{(1)}$ into atomic, actionable review points $\{R_1^{(1)}, R_2^{(1)}, \ldots, R_N^{(1)}\}$, where each $R_j^{(1)}$ represents a specific critique or suggestion that can be addressed independently.

The revision process proceeds iteratively through each review point. For each review point $R_j^{(1)}$, we first determine whether to revise or rebut. Since not all review feedback necessarily improves paper quality, we employ a decision function $C(R_j^{(1)}, D) \in \{0, 1\}$ that analyzes the review point against the current draft $D = \{D_1, D_2, \ldots, D_k\}$, where 0 indicates rebuttal and 1 indicates revision. This gating mechanism prevents blind compliance with potentially misguided feedback.

When $C(R_j^{(1)}, D) = 1$, we proceed with revision through a targeted modification process. First, a localization agent identifies the specific sections $\mathcal{S}_j \subseteq \{D_1, D_2, \ldots, D_k\}$ that require modification to address $R_j^{(1)}$. For each identified section $D_i \in \mathcal{S}_j$, we deploy an evidence retrieval agent that searches across three sources: the current complete draft, downloaded paper content from $L_A \cup L_T$, and publicly available academic databases. The agent extracts relevant evidence snippets $E_j^{(i)} = \{e_1, e_2, \ldots, e_r\}$ that inform the revision. Based on $R_j^{(1)}$ and $E_j^{(i)}$, we generate a revision plan $P_j^{(i)}$ and execute it to produce the modified section $\hat{D}_i = \mathcal{M}(D_i, R_j^{(1)}, E_j^{(i)})$, where $\mathcal{M}$ is the modification function that incorporates the review feedback while maintaining consistency with supporting evidence.

After completing the revision for review point $R_j^{(1)}$, we obtain an updated draft $\hat{D} = \{D_1, \ldots, \hat{D}_i, \ldots, D_k\}$ where modified sections replace their original versions. We then re-evaluate the updated draft to obtain a new quality score $\hat{Q}_j$. The revision is accepted only if $\hat{Q}_j > Q^{(1)}$, indicating genuine improvement. If $\hat{Q}_j \leq Q^{(1)}$, we roll back to the previous version, maintaining D unchanged. This quality gate ensures that each accepted revision contributes positively to the overall manuscript quality, preventing the introduction of new errors or inconsistencies during the revision process.

This process continues iteratively through all review points $\{R_1^{(1)}, R_2^{(1)}, \ldots, R_N^{(1)}\}$. After processing all points, we obtain the revised draft $D^{(2)}$ with quality score $Q^{(2)} \geq Q^{(1)}$. The entire revision cycle can be repeated with new rounds of review if needed, generating $R^{(2)}, R^{(3)}, \ldots$ until convergence or a predefined quality threshold is reached. By treating revision as a sequence of evidence-grounded, quality-gated decisions rather than blind compliance with feedback, our approach ensures that the manuscript evolves toward higher quality while maintaining scientific rigor and internal consistency. This mirrors the iterative refinement process that human researchers undergo when incorporating peer feedback, where each change is carefully evaluated for its net contribution to the work's clarity, validity, and impact.

4 Experiments

4.1 Experimental Settings

Following the recenttrends in using LLMs to judge the quality of out-put texts (especially in the setting of reference-freeevaluations) [**?** **?**], we use GPT-4 to judge the qual-ity of research ideas. Note that each of the problem,method, and experiment design is evaluated withfive different criteria. We ask the LLM-based evaluation model toeither rate the generated idea on a 5-point Likertscale for each criterion or perform pairwise compar-isons between two ideas from different models.

We compare AutoSurvey with surveys authored by human experts (collected from Arxiv) and naive RAG-based LLMs across 20 different computer science topics across 20 different topics in the field of LLMs (see Table 6). For the naive RAG-based LLMs, we begin with a title and a survey length requirement, then iteratively prompt the model to write the content until completion. Note that we also provide the model with the same number of reference papers with AutoSurvey.

We mainly use the GPT-4 [**?**] releasefrom Nov 06, 2023, as the basis for all models,which is, notably, reported to be trained with dataup to Apr 2023 (meanwhile, the papers used foridea generation appear after May 2023).

4.2 Comparative Experiments

We extensively evaluate The AI Scientist on three templates (as described in Section 3) acrossdifferent publicly available LLMs: Claude Sonnet 3.5 (Anthropic, 2024), GPT-4o (OpenAI, 2023), DeepSeek Coder (Zhu et al., 2024), and Llama-3.1 405b (Llama Team, 2024). The first two modelsare only available by a public API, whilst the second two models are open-weight.

5 Conclusion

We have proposed a new method to solve the challenges in AI research paper generation.

References

[] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *ArXiv Preprint* (2023).

[] Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. 2025. ResearchAgent: Iterative Research Idea Generation over Scientific Literature with Large Language Models. In *Nations of the Americas Chapter of the Association for Computational Linguistics*. 6709–6738.

[] Yanan Chen, Ali Pesaranghader, Tanmana Sadhu, and Dong Hoon Yi. 2024. Can We Rely on LLM Agents to Draft Long-Horizon Plans? Let's Take TravelPlanner as an Example. *ArXiv Preprint* (2024).

[] Jiangshu Du, Yibo Wang, Wenting Zhao, Zhongfen Deng, Shuaiqi Liu, Renze Lou, Henry Zou, Pranav Narayanan Venkit, Nan Zhang, Mukund Srinath, et al. 2024. LLMs Assist NLP Researchers: Critique Paper (Meta-) Reviewing. In *Empirical Methods in Natural Language Processing*. 5081–5099.

[] Lutfi Eren Erdogan, Hiroki Furuta, Sehoon Kim, Nicholas Lee, Suhong Moon, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. Plan-and-Act: Improving Planning of Agents for Long-Horizon Tasks. In *International Conference on Machine Learning*.

[] Lutfi Eren Erdogan, Hiroki Furuta, Sehoon Kim, Nicholas Lee, Suhong Moon, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. Plan-and-Act: Improving Planning of Agents for Long-Horizon Tasks. In *International Conference on Machine Learning*.

[] Aniketh Garikaparthi, Manasi Patwardhan, Lovekesh Vig, and Arman Cohan. 2025. Iris: Interactive research ideation system for accelerating scientific discovery. In *Annual Meeting of the Association for Computational Linguistics*.

[] Yolanda Gil, Mark Greaves, James Hendler, and Haym Hirsh. 2014. Amplify scientific discovery with artificial intelligence. *Science* 346, 6206 (2014), 171–172.

[] Sikun Guo, Amir Hassan Shariatmadari, Guangzhi Xiong, Albert Huang, Myles Kim, Corey M Williams, Stefan Bekiranov, and Aidong Zhang. 2025. Ideabench: Benchmarking large language models for research idea generation. In *Knowledge Discovery and Data Mining*. 5888–5899.

[] Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. 2024. MLAgentBench: evaluating language agents on machine learning experimentation. In *International Conference on Machine Learning*. 20271–20309.

[] Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. 2024. Understanding the planning of LLM agents: A survey. *ArXiv Preprint* (2024).

[] Marcus Hutter. 2001. Towards a universal theory of artificial intelligence based on algorithmic probability and sequential decisions. In *European conference on machine learning*. 226–238.

[] Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W Mahoney, Kurt Keutzer, and Amir Gholami. 2024. An llm compiler for parallel function calling. In *International Conference on Machine Learning*.

[] Pat Langley. 1987. *Scientific discovery: Computational explorations of the creative processes*.

[] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-Eval: NLG Evaluation using Gpt-4 with Better Human Alignment. In *Empirical Methods in Natural Language Processing*. 2511–2522.

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

[] Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. 2024. ADaPT: As-Needed Decomposition and Planning with Language Models. In *Findings of the Association for Computational Linguistics*. 4226–4252.

[] Kevin Pu, KJ Kevin Feng, Tovi Grossman, Tom Hope, Bhavana Dalvi Mishra, Matt Latzke, Jonathan Bragg, Joseph Chee Chang, and Pao Siangliulue. 2025. Ideasynth: Iterative research idea development through evolving and composing idea facets with literature-grounded feedback. In *Conference on Human Factors in Computing Systems*. 1–31.

[] Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. 2024. Can LLMs Generate Novel Research Ideas? A Large-Scale Human Study with 100+ NLP Researchers. In *International Conference on Learning Representations*.

[] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. 2023. ProgPrompt: Generating Situated Robot Task Plans using Large Language Models. In *International Conference on Robotics and Automation*. 11523–11530.

[] Haotian Sun, Yuchen Zhuang, Lingkai Kong, Bo Dai, and Chao Zhang. 2023. Adaplanner: Adaptive planning from feedback with language models. *Advances in Neural Information Processing Systems* 36 (2023), 58202–58245.

[] Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. 2023. Scientific discovery in the age of artificial intelligence. *Nature* 620, 7972 (2023), 47–60.

[] Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models. In *Annual Meeting of the Association for Computational Linguistics*. 2609–2634.

[] Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. 2023. Scimon: Scientific inspiration machines optimized for novelty. In *Annual Meeting of the Association for Computational Linguistics*. 279–299.

[] Yidong Wang, Qi Guo, Wenjin Yao, Hongbo Zhang, Xin Zhang, Zhen Wu, Meishan Zhang, Xinyu Dai, Qingsong Wen, Wei Ye, et al. 2024. Autosurvey: Large language models can automatically write surveys. *Advances in Neural Information Processing Systems* 37 (2024), 115119–115145.

[] Yixuan Weng, Minjun Zhu, Guangsheng Bao, Hongbo Zhang, Jindong Wang, Yue Zhang, and Linyi Yang. 2025. CycleResearcher: Improving Automated Research via Automated Review. In *International Conference on Learning Representations*.

[] Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob Foerster, Jeff Clune, and David Ha. 2025. The ai scientist-v2: Workshop-level automated scientific discovery via agentic tree search. *ArXiv Preprint* (2025).

[] Ke Yang, Yao Liu, Sapana Chaudhary, Rasool Fakoor, Pratik Chaudhari, George Karypis, and Huzefa Rangwala. 2024. AgentOccam: A Simple Yet Strong Baseline for LLM-Based Web Agents. In *International Conference on Learning Representations*.

[] Zonglin Yang, Xinya Du, Junxian Li, Jie Zheng, Soujanya Poria, and Erik Cambria. 2024. Large Language Models for Automated Open-domain Scientific Hypotheses Discovery. In *Findings of the Association for Computational Linguistics*. 13545–13565.

[] Yao Zhang, Zijian Ma, Yunpu Ma, Zhen Han, Yu Wu, and Volker Tresp. 2025. Webpilot: A versatile and autonomous multi-agent system for web task execution with strategic exploration. In *AAAI Conference on Artificial Intelligence*, Vol. 39. 23378–23386.

[] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems* 36 (2023), 46595–46623.

Temporary page!

LaTeX was unable to guess the total number of pages correctly. As there was some unprocessed data that should have been added to the final page this extra page has been added to receive it.

If you rerun the document (without altering it) this surplus page will go away, because LaTeX now knows how many pages to expect for this document.