Imitate-and-Evolve: A Multi-agent frameowrk to Bootstrap Long-Horizon Planning for Scientific Research 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. Imitate-and-Evolve: A Multi-agent frameowrk to Bootstrap Long-Horizon Planning for Scientific Research 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

Scientific research plays a crucial role in drivinginnovation, advancing knowledge, solving problems, expanding our understanding of the world,and ultimately improving the lives of people in tan-gible ways [**?**]. However, thisis a slow, effort-intensive process, which requiresreading and synthesizing overwhelming amountsof knowledge over the vast corpus of rapidly growing scientific literature to formulate research ideas,as well as design and perform experimental valida-tions of those ideas [**?**]. Many promising ideas are thus buried under the intensive effort.

Automating general scientific discovery has been a long-standing ambition of the research community, dating back to the 1970s–1980s [**?**]. Recent large language models (LLMs) [**?**] have made impressive progress in processing, organizing, and generating scientific text [**?**]. Most current efforts have relied on commercial LLMs to build agents that propose research ideas research ideas [**? ? ?**], asan assistant to conduct experiment [**?**], or an AI scientist capable of generating automated open-ended scientific publications [**? ?**].Most recent work now close the loop performing the full cycle of automated researchand review, from literature review and manuscript preparation to peer review andpaper refinement. [**?**].

With the generated the first workshop paper written entirely by AI and accepted through peer review [**?**], there is road for Automating general scientific discovery human expert , technically sound, well-organized and clearly written, especially when the research problems addressed, or methods is novel from exisitng work. In this paper, we focus on one of the most challening bottlneck, error accumulation long-horizon planning. It is known drafting a writing a research paper is a long-horizon process that requires multi-stage reasoning, from literature review and manuscript preparation to peer review andpaper refinement. Previous studies on multi-step models show that the inherent long-horizon workflow of such systems inevitably introduces the risk of compounding errors. In the workflow, small mistakes made at earlier steps can accumulate and propagate through subsequent steps, as errors that occur at a

Manuscript submitted to ACM

certain step will affect the next state [**?**]. Without effective self-correction mechanisms, these deviations will further expand the error in later steps [**? ? ?**].

As a demonstration, we employ the state-of-the-art research assistant, OpenAI' Deep Research [**?**] and Gemini Deep Research to draft a paper on *Diffusion models for data augmentation in EEG (electroencephalogram)-based emotion recognition*. As shown in Fig. **??**, the generated paper contains several critical flaws, including a misalignment between the proposed method and the cited references, as well as inconsistencies between the experimental design and the stated research objectives. , plan match These issues likely stem from error accumulation during the long-horizon planning process

Based on this observation, a question raise: what is the best way to plan for long-horizon research paper generation mitigate the error accumulation? It is known that 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 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.

Acutally, 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 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 Methodology

In this section, we present our Imitation-and-Evolve approach in detail. The overall framework, illustrated in Fig. **??**, consists of two stages: drafting and revision. In the drafting stage, the imitation agent emulates relevant work to plan the manuscript outline, while the evolve agent iteratively drafts and refines the outline. In the revision stage, the imitation agent simulates human review to generate revision plan, and the evolve agent iteratively revises the manuscript and decides whether to adopt each point.

3.1 Imitation-and-Evolve for Drafting

Given a user request U describing a research topic, we first propose an imitation agent to imitate human expert planning. The imitation agent first prompts an LLM to analyze U and decompose it into an algorithm of interest U_A and an application of interest U_T. Formally, we represent this as $\mu(U) = (U_A, U_T; r)$, where $\mu(.)$ is the decomposition function and r denotes any additional requirements or constraints.

Then, the imitation agent retrieves related work in terms of the algorithm of interest and the application of interest. The retrieval results are represented as $\phi(U_A) = \{L_A^{(1)}, \ldots, L_A^{(K)}\}$ and $\phi(U_T) = \{L_T^{(1)}, \ldots, L_T^{(K)}\}$, where $L_A^{(m)}$ and $L_T^{(n)}$ denote the m-th and n-th retrieved papers, respectively. The imitation agent is equipped with two tools for retrieval, i.e., a search tool and a rerank tool. The search tool receives a query and invokes HuggingFace's paper API[1] to search for candidate papers. The rerank tool uses the bge-reranker-v2-m3 [**?**] model to encode both the query and the titles and abstracts of the searched papers, and returns the top K most similar results. To find an optimal combination of retrieved algorithm and application, we prompt the LLM to evaluate all pairs of retrieved algorithms and applications, and identify the best-matched pair. This process derives $M_A \in L_A^{(m)}$ and $M_T \in L_T^{(n)}$, representing the matched algorithm and application papers, respectively.

The imitation agent refers to the section and subsection outlines of these selected papers as human expert plans for organizing the manuscript. We prompt the LLM to extract the sections and subsections along with their summaries from M_A and M_T, resulting in $P_A = \sigma(M_A)$ and $P_T = \sigma(M_T)$. Next, the imitation agent imitates these outlines to generate a new outline for the target manuscript, with the algorithm of interest U_A, application of interest U_T, and any constraints r as additional inputs. The derived outline P serves as the initial plan for drafting the manuscript.

Then, P is handed off to the evolve agent, which is responsible for executing P and updating it whenever discrepancies between the plan and observed results arise. The evolve agent follows an iterative paradigm, starting from step $n = 1$ and adjustment index $t = 1$, which corresponds to the initial execution before any adjustments. The plan at the t-th adjustment is denoted as $P^{(t)} = \{P_1^{(t)}, \ldots, P_N^{(t)}\}$, where N is the number of planned steps. For the plan $P_n^{(t)}$ at the n-th step after $t-1$ adjustments, the evolve agent first fetches observations relevant to $P_n^{(t)}$. The evolve agent is equipped with two tools, i.e., an external tool for fetching segments from external resources, and a contextual tool for fetching content within the existing draft. When using these tools, the evolve agent is provided with external resources, such as M_A and M_T, or the current draft as context. The LLM is prompted to identify the appropriate line ranges, and the content within these ranges is extracted and concatenated to facilitate efficient evidence gathering. The content retrieved from context for step n is denoted as C_n, while the content retrieved from external resources is denoted as E_n. Using these observations, the evolve agent updates the plan according to

$$P_i^{(t+1)} = \pi(P_i^{(t)} \mid C^{(t)}, E^{(t)}, P_i^{(t)}) \tag{1}$$

[1] https://huggingface.co/docs/hub/api

where $\pi(.)$ is a policy function that revises each step i of the current plan based on the available context $C^{(t)}$, external evidence $E^{(t)}$, and the i-th step of the current plan $P_i^{(t)}$. The policy function is implemented by prompting the LLM to update the plan according to the observations. Then, the evolve agent

The revised step $P_i^{(t+1)}$ is then executed to produce the draft D_i for the i-th section, using the contextual segments C_i and external segments E_i as context, and $P_i^{(t+1)}$ as the instruction. After deriving the draft D_i, the imitation agent further updates the remaining steps in the plan. The subsequent plan steps are modified as needed to be appropriate given the new results:

$$P_j^{(t+2)} = \begin{cases} P_j^{(t)}, & \text{if } j < i \\ P_j^{(t+1)}, & \text{if } j = i \\ \pi\left(P_j^{(t)} \mid \{D_p^{(t)}\}_{p \le i}, \{P_q^{(t)}\}_{q<j}\right), & \text{if } j > i \end{cases} \tag{2}$$

where $P_j^{(t+2)}$ denotes the j-th step in the updated plan after considering the collection of drafts $\{D_p^{(t)}\}_{p \le i}$ already produced and the collection of plan steps $\{P_q^{(t)}\}_{q<j}$ as context. In this paper, the LLM is prompted to revise the subsequent plan steps and return a revised version if the current steps are not appropriate; otherwise, the original steps are retained.

Next, the evolve agent moves to the next step in the plan and starts a new round of iteration. This iterative process continues until all N steps are completed. Finally, the evolve agent collects the section titles according to the last version of the plan and combines them with all the drafts $\{D_1, D_2, \ldots, D_N\}$ to form the main body of the manuscript. The evolve agent then prompts the LLM to generate the title and abstract based on the completed body as context. Supplemented with references, all the content is rendered into a LaTeX project according to the conference template, and the final PDF of the paper is generated.

3.2 Imitation-and-Evolve for Revision

After deriving draft sections $\{D_1, D_2, \ldots, D_{N_D}\}$, we design the revision stage, which aims to further improve the manuscript. To plan systematic revisions, we introduce an imitation agent that mimics the peer-review and revision process commonly practiced by human experts in scientific research.

We first design the imitation agent to generate review feedback and decompose it into actionable points, which then serve as the revision plan. Following the standard peer-review protocols adopted by leading conferences that make their review comments public on OpenReview[2] (e.g., ICLR), our imitation agent prompts the LLM to generate comprehensive review feedback across multiple dimensions. Formally, given the complete manuscript $\{D_1, D_2, \ldots, D_{N_D}\}$, the imitation agent applies a feedback generation function δ such that $F = \delta(\{D_1, D_2, \ldots, D_{N_D}\})$. The generated feedback F covers aspects such as summary, soundness, presentation, contribution, strengths, weaknesses, questions, and overall rating. Next, we decompose F into a set of review points $\{G_1, G_2, \ldots, G_L\} = \beta(F)$, where β is a decomposition function that prompts the LLM to enumerate feedback items from F. Each G_l represents a comment that can be addressed independently, forming the initial revision plan.

These review points $\{G_1, G_2, \ldots, G_L\}$ are then handed off to the evolve agent, which is responsible for executing and updating the revision plan in response to observed outcomes. The evolve agent operates in an iterative paradigm, starting from review point $l = 1$ and adjustment index $t = 1$, corresponding to the initial execution before any adjustments. The revision plan at the t-th adjustment is denoted as $\{G_1^{(t)}, \ldots, G_L^{(t)}\}$, where L is the total number of review points. For a

[2]https://openreview.net/

given review point $G_l^{(t)}$ at step l after $t-1$ adjustments, the evolve agent first gathers relevant observations. Similar to imitation-and-evolve for drafting, it is equipped with two tools: an external tool for retrieving evidence from external resources, and a contextual tool for extracting content from the current manuscript draft. The content extracted from the current draft for step l is denoted as C_l, while content from external resources is denoted as E_l.

Not all review feedback should be followed unconditionally, as some points may arise from misunderstandings. Therefore, for each plan G_l, the evolve agent determines whether to revise or rebut using a decision function $C(G_l, C_l, E_l) \in \{0, 1\}$, where 0 indicates rebuttal and 1 indicates revision. The update to the plan before execution can be formalized as:

$$G_l^{(t+1)} = \begin{cases} \varnothing, & \text{if } C(G_l, C_l, E_l) = 0 \\ G_l^{(t)}, & \text{if } C(G_l, C_l, E_l) = 1 \end{cases} \tag{3}$$

where $\varnothing$ denotes an empty plan that does not need to be executed, indicating that the review point is skipped if the decision is to rebut. If the decision is to rebut, the agent skips to the next point. Otherwise, the agent proceeds to modify the relevant section identified by C_l. The revised section is denoted as $\hat{D}_i = \eta(D_i, G_l)$, where η is a modification function that incorporates the feedback G_l into the original section draft D_i.

After executing the plan, we evaluate the revised draft to assess its quality. Here, we prompt the LLM to generate feedback on both versions, putting the revised section $\hat{D}_i$ and the original D_i back into the manuscript, respectively. Then we use $\delta()$ to generate feedback on the two versions of the manuscript and extract a quality score, with the derived score of the corresponding version denoted as $\gamma(\cdot)$. The acceptance or rollback of the update can be formalized as:

$$D_i^{(t+1)} = \begin{cases} \hat{D}_i, & \text{if } \gamma(\hat{D}_i) \geq \gamma(D_i) \\ D_i, & \text{if } \gamma(\hat{D}_i) < \gamma(D_i) \end{cases} \tag{4}$$

where the revision is accepted only if $\gamma(\hat{D}_i) \geq \gamma(D_i)$, indicating a genuine improvement. If the quality does not improve, we roll back to the previous version, keeping D_i unchanged.

This iterative process continues for all review points $\{G_1, G_2, \ldots, G_L\}$, resulting in a revised draft. The entire revision cycle can be repeated with new rounds of review feedback and revisions.

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).

5 Experiments

5.1 Experimental Settings

We conduct experiments on two benchmark datasets, AutoSurvey [**?**] and SurveyEval [**?**], as well as a self-constructed benchmark dataset, TopSurvey. For hyperparameters, we set the maximum number of iterations for exploration, exploitation, and experience taskforces to 4. The hyperparameter θ is set to 500. We use GPT-4.1 as the LLM for both generation and evaluation. The evaluation comprises two categories of metrics. For citation quality, we adopt citation recall and citation precision as proposed by **?**]: recall measures whether cited passages fully support all statements, while precision measures the proportion of relevant citations that support their corresponding statements. For content quality, following **?**], we use coverage, structure, and relevance, each rated by LLMs on a 5-point scale. Coverage assesses the extent to which the survey addresses all relevant aspects of the topic; structure evaluates logical organization and coherence; and relevance measures alignment with the specified research topic. We do not filter out fractional scores, such as 4.5.

5.2 Comparision Experiments

We first evaluate our framework on the AutoSurvey [**?**] benchmark, following the protocols established by **? ?**]. We use the same 20 topics from diverse subfields of LLM research to generate survey articles for comparison. We compare our approach with three baselines: Naive RAG-based LLMs using retrieval-augmented generation, AutoSurvey [**?**], and SurveyX [**?**]. We use the official implementations of AutoSurvey[3] and SurveyX[4] to conduct experiments. Due to the lack of an online version of SurveyX, we conduct offline generation.

As shown in Table **??**, most algorithms exhibit reduced performance as survey length increases, particularly in recall, precision, structure, and relevance. This decline is most pronounced in the Naive RAG baseline, indicating that longer workflows lead to greater error accumulation and propagation. In contrast, coverage remains stable or improves with longer surveys, likely due to more comprehensive topic inclusion. State-of-the-art methods such as AutoSurvey and SurveyX continue to face challenges with citation precision, frequently generating statements unsupported by references. For content quality, structure consistently receives the lowest scores, reflecting disorganized article organization. Our proposed method remains robust to these issues and consistently achieves superior results across all metrics.

We further evaluate our framework on the SurveyEval benchmark [**?**], using the same protocols. SurveyEval is the first benchmark in computer science that pairs surveys with complete reference papers, comprising 384 arXiv cs.CL surveys citing over 26,000 references. Twenty topics were selected for testing based on reference completeness and reference list diversity. Experimental results are shown in Fig. **??**, with Coverage, Structure, and Relevance scores normalized to a 100-point scale.

In Fig. **??**, all methods achieve high coverage and relevance, indicating that LLMs like GPT-4.1 can generate comprehensive and relevant content. However, recall and precision remain low, reflecting poor reference retrieval and insufficient support for generated statements. As a result, state-of-the-art methods struggle with logical organization, leading to lower structure scores. Our method overcomes these issues, achieving the best performance across all metrics.

[3]https://github.com/AutoSurveys/AutoSurvey
[4]https://github.com/IAAR-Shanghai/SurveyX

width=

2.5*Methods	Citation Quality		Content Quality			
	Rec. ↑	Pre. ↑	Cov. ↑	Str. ↑	Rel. ↑	Avg. ↑
w/o exploration	94.38	88.56	4.83	4.83	5.00	4.89
w/o exploitation	97.86	79.02	4.97	4.93	4.93	4.94
w/o experience	97.78	80.82	4.79	4.89	4.97	4.88
gray!15Proposed	gray!15**98.17**	gray!15**89.28**	gray!15**4.97**	gray!15**4.95**	gray!15**5.00**	gray!15**4.97**

Table 1. Ablation study of the proposed modules on the benchmark dataset.

width=0.75

2.5*Methods	Citation Quality		Content Quality			
	Recall ↑	Precision ↑	Coverage ↑	Structure ↑	Relevance ↑	Avg. ↑
Human	93.07	87.76	5.00	4.97	5.00	4.99
Naive RAG (2024)	64.57	61.89	4.29	3.58	4.67	4.18
AutoSurvey (2024)	70.03	71.66	4.68	4.67	4.87	4.74
SurveyX (2025)	75.51	77.90	4.71	4.84	4.93	4.83
gray!15Proposed	gray!15**86.63**	gray!15**81.98**	gray!15**4.85**	gray!15**4.90**	gray!15**5.00**	gray!15**4.92**

Table 2. Comparison of automatic survey generation methods at a survey length of 64k tokens on the new large-scale benchmark dataset. Higher scores indicate better performance.

width=

2.5*Method	2.5*Iteration	Citation Quality		Content Quality			
		Rec. ↑	Pre. ↑	Cov. ↑	Str. ↑	Rel. ↑	Avg. ↑
5*Exper.	1	83.64	74.37	4.61	4.79	4.94	4.78
	2	84.99	78.19	4.77	4.83	4.98	4.86
	3	85.11	80.13	4.82	4.86	4.98	4.89
	gray!154	gray!1586.63	gray!15**81.98**	gray!15**4.85**	gray!15**4.90**	gray!15**5.00**	gray!15**4.92**
	5	**86.71**	81.86	**4.85**	4.88	4.98	4.90
5*Explor.	1	82.84	81.26	4.71	4.78	**5.00**	4.83
	2	84.79	81.67	4.83	4.82	**5.00**	4.88
	3	85.69	81.97	**4.85**	4.84	**5.00**	4.90
	gray!154	gray!1586.63	gray!15**81.98**	gray!15**4.85**	gray!15**4.90**	gray!15**5.00**	gray!15**4.92**
	5	**86.65**	81.92	**4.85**	4.87	**5.00**	4.91
5*Exploi.	1	86.31	72.68	4.81	4.87	4.91	4.86
	2	86.00	77.34	4.83	4.90	4.94	4.89
	3	**86.75**	79.10	4.84	4.89	4.98	4.90
	gray!154	gray!1586.63	gray!15**81.98**	gray!15**4.85**	gray!15**4.90**	gray!15**5.00**	gray!15**4.92**
	5	86.47	80.27	4.84	4.89	5.00	4.91

Table 3. Sensitivity experiments across different iterations for the experience, exploration, and exploitation taskforces on the new large-scale benchmark.

5.3 Ablation Study

We further conduct an ablation study to evaluate the effectiveness of each component in our proposed framework. Experiments are performed on the same benchmark dataset as before, with all experimental settings unchanged and the survey length set to 8k tokens. We compare the full model with three variants. We use a single round of retrieval and organization for the variant without the exploration taskforce to generate the overall outline. We draft the manuscript using only one round of extraction and writing for the variant without the exploitation taskforce. For the variant without the experience taskforce, each agent completes its assigned task, without revision.

Table **??** presents the results. Removing the exploration taskforce causes the largest drops in recall and structure, demonstrating its importance for citation recall and logical organization. Excluding the exploitation taskforce sharply reduces precision and relevance, confirming its role in improving citation precision and content relevance. Without the experience taskforce, all metrics decrease, especially coverage, highlighting its key role in ensuring comprehensive coverage through collaborative revision.

5.4 A New Large-Scale Benchmark

To further validate the effectiveness of our proposed framework, we construct a new large-scale benchmark dataset. This dataset consists of 195 topics from various subfields of computer science, nearly 10 times larger than previous benchmarks [**? ?**]. To ensure topic quality, we collect peer-reviewed survey topics from top computer science conferences, rather than from preprint sources such as arXiv. Survey papers accepted only as abstracts are excluded. To prevent data leakage from LLM pretraining data, we include only survey papers published in 2023, 2024, and 2025. We employ PhD students in computer science to verify whether a paper qualifies as a survey and to collect the final set of 195 survey papers: 9 from AAAI, 34 from ACL, 46 from EMNLP, 3 from ICLR, 2 from ICML, 77 from IJCAI, and 24 from NAACL. Of these, 68 were published in 2023, 105 in 2024, and 22 in 2025.

We conduct experiments using the same settings as above, generating 64k-token literature reviews for evaluation. As shown in Table **??**, compared with results on the existing benchmark in Table **??**, performance on the new large-scale benchmark drops significantly, particularly in recall and coverage. This may be due to the greater number and broader range of topics, which leads to some relevant literature not being retrieved, resulting in lower citation recall and coverage. Precision also drops slightly, likely because the models lack knowledge of the most recent two years. Therefore, compared to existing benchmarks, the new large-scale benchmark is more challenging and leads to more errors in generation. Despite this, our proposed method achieves over 80% in citation scores and an average content quality score of 4.92.

5.5 Sensitivity Analysis

To gain deeper insight into our framework, we conduct a sensitivity analysis to observe how self-improving iterations affect performance. We evaluate the experience, exploration, and exploitation taskforces across different numbers of iterations, employing a controlled variable approach: when varying one hyperparameter, all others are held constant as previously described.

The results are presented in Table **??**. We find that the second and third iterations yield the most significant improvements. This indicates that errors can accumulate at various steps in the workflow, and iterative refinement effectively reduces intermediate errors, thereby enhancing the quality of the final results. Both the Experience and exploitation taskforces contribute most to faithfulness; from the first to the fourth iteration, the number of references

width= 2.5*Cluster	Citation Quality		Content Quality		
	Recall ↑	Precision ↑	Coverage ↑	Structure ↑	Relevance ↑
1	83.61	**82.57**	4.67	4.78	4.93
2	84.54	75.12	4.75	4.85	4.85
3	72.29	76.11	4.59	4.88	4.87
4	**86.63**	81.98	**4.85**	**4.90**	**5.00**
5	81.37	69.77	4.66	4.74	4.73

Table 4. Real-world case study across different topics on 400 generated results. Higher scores indicate better performance.

supporting claims in the final output increases by a large margin. Additionally, the Experience and exploration taskforces contribute notably to coverage and structural quality, suggesting that a single revision is often insufficient for comprehensive improvement. However, the effect of additional iterations gradually diminishes. By the fifth iteration, we observe a slight decline, due to unnecessary modifications overwriting correct results and introducing new errors.

5.6 Disscussions

We evaluated our framework in a real-world setting by deploying an online automatic literature review generation system[5], which has produced over 20,000 reviews. We randomly selected 400 reviews, embedded them with all-MiniLM-L6-v2, and applied K-means clustering to form five groups. Results show differences in citation and content quality among clusters. The best performance appears in computer science (cluster 4), while education and social sciences (cluster 3) perform worse, especially in citation quality. It may result from the search engine not including part of the relevant literature in the social sciences. We also measured system efficiency, and generating an 8,000-token review takes an average of 8.45 minutes.

6 Conclusion

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

[5]Anonymous for review; to be revealed upon acceptance.

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.