PaperScout: An Autonomous Agent for Academic Paper Search with Process-Aware Sequence-Level Policy Optimization

Anonymous ACL submission

Abstract

Academic paper search is a fundamental task in scientific research, yet most existing approaches organize retrieval around predefined workflows or structured interaction protocols that struggle with complex, conditional queries. To address this limitation, we propose PaperScout, an autonomous agent that reformulates paper search as a sequential decision-making process. Unlike static workflows, PaperScout dynamically decides whether, when, and how to invoke `search` and `expand` tools based on accumulated retrieval context. However, training such agents presents a fundamental challenge: standard reinforcement learning methods, typically designed for single-turn tasks, suffer from a granularity mismatch when applied to multi-turn agentic tasks—where token-level optimization diverges from the granularity of sequence-level interactions—leading to noisy credit assignment and unstable training dynamics. We introduce Proximal Sequence Policy Optimization (PSPO), a process-aware, sequence-level policy optimization method that aligns optimization with agent–environment interaction. Comprehensive experiments on both synthetic and real-world benchmarks demonstrate that PaperScout significantly outperforms strong structured retrieval and RL baselines in both recall and relevance, validating the effectiveness of our adaptive agentic framework and optimization strategy. Our code is available[1].

1 Introduction

Academic paper search underpins effective knowledge discovery (Timmins and McCabe, 2005; Marchionini, 2006). Most traditional approaches rely on lexical or semantic matching over largely static corpora, which perform well on well-formed queries (Vine, 2006; Zhu et al., 2023; Shi et al., 2025). However, as scholarly publications originate from increasingly heterogeneous sources, static databases become costly to maintain (Manghi, 2024). Meanwhile, traditional methods exhibit limited capability when faced with fine-grained and conditional queries (Gusenbauer and Haddaway, 2020). A researcher seeking *"Apply reinforcement learning to protein folding while excluding Transformer-based architectures"* often encounters thematically relevant but constraint-violating papers, leading to substantial manual search costs (Gusenbauer and Haddaway, 2021).

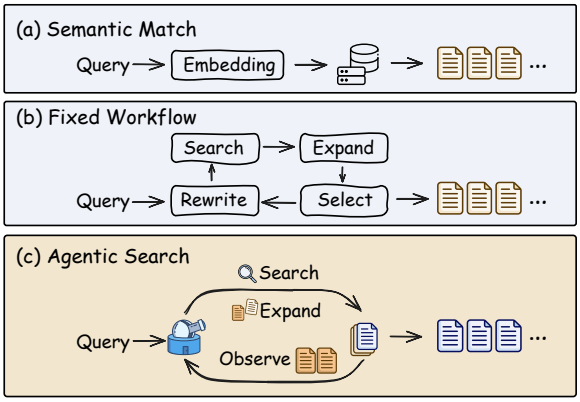


Figure 1: Comparison of different academic search paradigms: Semantic Match, Fixed Workflow, and Agentic Search.

Recent advances in large language models (LLMs) have introduced new opportunities for academic paper search (Zhang et al., 2025; Wang et al., 2025a,b). Going beyond single-round querying and static ranking, LLMs leverage accumulated context to guide subsequent steps and produce structured outputs, such as rewritten queries, candidate inspection criteria, and citation cues (Zhu et al., 2023), thereby enabling fine-grained control over the retrieval process. Unlike traditional semantic matching, this interaction-driven retrieval better reflects practical search processes, effectively handling complex queries and gradually narrowing the search space (Shi et al., 2025).

[1] https://anonymous.4open.science/r/PaperScout-283E

Despite the increased expressive capacity of LLM-based approaches for academic paper search, most systems still rely on predefined workflows or constrained interaction protocols. They decompose retrieval into stages such as query rewriting, search, paper inspection, and reference expansion, and specify how these stages are connected. This constraint also applies to agentic systems such as PaSa, where search, expansion, and stopping actions are still organized by a paper-queue traversal protocol. Consequently, learning mainly improves local decisions within a fixed structure, but leaves higher-level search control—when to broaden search, deepen expansion, or redirect exploration—largely underexplored.

To address these limitations, we propose **PaperScout**, an autonomous paper search agent that reframes retrieval as a sequential decision-making process rather than predefined workflow execution. Instead of being tied to a search-then-traverse protocol, PaperScout decides at each step *whether*, *when*, and *how* to invoke `Search` and `Expand` based on the papers accumulated so far. By giving the agent direct control over retrieval decisions, PaperScout enables flexible, context-dependent exploration, allowing search strategies to evolve as the retrieval process unfolds.

However, training such an autonomous multi-turn retrieval agent remains challenging for existing policy optimization methods. Outcome-oriented approaches (e.g., GRPO) (Shao et al., 2024), while effective for reasoning tasks, primarily optimize against final outcomes and cannot naturally incorporate process rewards from intermediate retrieval steps. This limitation is particularly restrictive for multi-turn retrieval, where step-wise feedback is crucial for guiding sequential decisions. In contrast, standard token-level methods (e.g., PPO) (Schulman et al., 2017) optimize policies at the token level, whereas the agent's decisions are made at the interaction level, with each turn producing a complete response that triggers tool executions. This mismatch leads to misaligned optimization, unstable value estimation, and noisy credit assignment.

To bridge this granularity mismatch, we propose **Proximal Sequence Policy Optimization** (PSPO), a process-aware policy optimization method tailored for multi-turn agents. Unlike outcome-only baselines, PSPO performs advantage estimation at the sequence level (per interaction turn) and explicitly incorporates intermediate process rewards, ensuring that the optimization granularity aligns with agent–environment interactions.

In summary, our main contributions are:

- We propose PaperScout, the first autonomous paper search agent that adaptively controls search and reference expansion based on accumulated retrieval context.
- We introduce Proximal Sequence Policy Optimization (PSPO), a process-aware, sequence-level policy optimization method that aligns optimization granularity with multi-turn agent interactions.
- Extensive experiments demonstrate that PaperScout enables more effective and flexible multi-turn retrieval than baselines, while analysis shows that PSPO delivers superior sample efficiency and stability over token-level and outcome-oriented methods.

2 Related Work

2.1 Query-Centric Academic Search

Traditional academic paper search methods predominantly follow a query-centric retrieval setting, in which retrieval is driven by a single user query and focuses on single-round query–document relevance modeling (Schütze et al., 2008; Guo et al., 2020). Early approaches retrieve papers by measuring term-level similarity between the query and documents (Qaiser and Ali, 2018). To mitigate the limitations of semantic mismatch, later work moves toward dense retrieval, which improves query–document matching by encoding queries and documents into semantic vector representations (Jiang et al., 2019; Reimers and Gurevych, 2019). More recently, LLMs have been employed to rewrite or expand queries prior to retrieval in order to better capture user intent (Ma et al., 2023; Gusenbauer and Haddaway, 2020). Despite these advances, such methods still treat paper search as a single-round query–document matching task, which limits their ability to handle complex search queries (Gusenbauer and Haddaway, 2020).

2.2 Multi-Turn Academic Paper Search

To move beyond the query-centric, single-round retrieval paradigm, recent studies have explored multi-turn approaches for academic paper search (Cheng and Suo, 2025; Aytar et al., 2025; Park et al., 2025). These methods typically formulate

retrieval as a sequence of operations, enabling systems to iteratively search, inspect candidate papers, and expand along citation links. Representative work such as PaSa (He et al., 2025) trains a crawler to invoke search, expansion, and stop actions within a paper-queue interaction framework, while SPAR (Shi et al., 2025) integrates query rewriting, reference exploration, and re-ranking in a modular workflow. Despite their effectiveness, these approaches still rely on predefined interaction structures, offering limited flexibility in adapting high-level retrieval strategies to different queries or evolving search contexts.

2.3 Reinforcement Learning for LLM Agents

Reinforcement learning has been widely adopted to enhance LLM capabilities (Luo et al., 2025b; Jiang et al., 2025; Liu et al., 2025). Traditional approaches like PPO (Schulman et al., 2017) operate at the token level, where the difficulty of accurate credit assignment often leads to optimization instability under sparse rewards (Ouyang et al., 2025). To mitigate this, outcome-oriented methods such as GRPO (Shao et al., 2024) and GSPO (Zheng et al., 2025) have gained traction by leveraging group-wise outcome comparisons. However, extending these paradigms to multi-turn agents presents significant challenges (Wang et al., 2025c; Cheng et al., 2025). Most existing approaches generalize outcome-based logic to the trajectory level, treating the entire trajectory as a monolithic unit (Luo et al., 2025a). This granularity is coarse for multi-turn retrieval, as it overlooks the dense process signals available at intermediate steps.

3 Methodology

In this section, we present PaperScout, an autonomous LLM-based agent for academic paper search. An overview of the PaperScout framework is shown in Figure 2. We first formalize paper search as a Partially Observable Markov Decision Process (POMDP) in Section 3.1. Building on this formulation, Section 3.2 instantiates each POMDP component in a concrete multi-tool agent system. To optimize the agent policy under sequence-level feedback, Section 3.3 introduces Proximal Sequence Policy Optimization (PSPO).

3.1 Problem Definition

We formalize academic paper search as a Partially Observable Markov Decision Process (POMDP) $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \Omega, \mathcal{O}, \gamma \rangle$. The state $s_t \in \mathcal{S}$ represents a latent *paper pool* that contains all papers accumulated up to step t. Due to limited context, the agent cannot access s_t directly and instead receives an observation $o_t = \mathcal{O}(s_t) \in \Omega$, which provides a partial view of the pool (e.g., a small subset of highly relevant papers). Based on o_t, the agent chooses an action $a_t \in \mathcal{A}$ to acquire new information. In our setting, an action is instantiated by invoking one or more external retrieval tools (such as web search or reference expansion), whose execution returns additional candidate papers. The transition $\mathcal{P}(s_{t+1} \mid s_t, a_t)$ updates the paper pool by incorporating newly retrieved papers, and the reward $r_t = \mathcal{R}(s_t, a_t)$ measures the marginal utility of this acquisition with respect to the user query. This formulation casts paper search as an iterative decision-making problem rather than a one-shot ranking task.

Tool	Description
`Search(query)`	Calls scholarly search APIs to retrieve new papers.
`Expand(paper)`	Retrieves reference papers cited by the input paper.

Table 1: Tools available to the agent.

3.2 PaperScout

PaperScout instantiates the POMDP in a modular agentic system (Figure 2). While Section 3.1 provides an abstract formulation, we now describe the concrete design for state tracking, observation construction, and policy execution.

State and Observation Space. We implement the latent state s_t as a *paper pool* $\mathcal{B}_t$. Each paper in $\mathcal{B}_t$ stores (i) its content and metadata, (ii) a relevance score $\rho(\cdot)$ with respect to the user query, and (iii) an expansion flag indicating whether its references have been explored. Together, $\mathcal{B}_t$ maintains the agent's current search frontier.

The observation o_t summarizes $\mathcal{B}_t$ under the agent's limited context budget. We use a dual-list view that partitions top-ranked papers into (a) expanded papers, which provide stable context from previously explored nodes, and (b) unexpanded papers, which are candidates for further exploration. We further augment o_t with the interaction history $\mathcal{H}_{t-1}$ to reduce redundant exploration, including past queries and previously expanded papers.

Policy and Execution Mechanism. The environment provides a structured observation o_t, which we serialize into the LLM input sequence

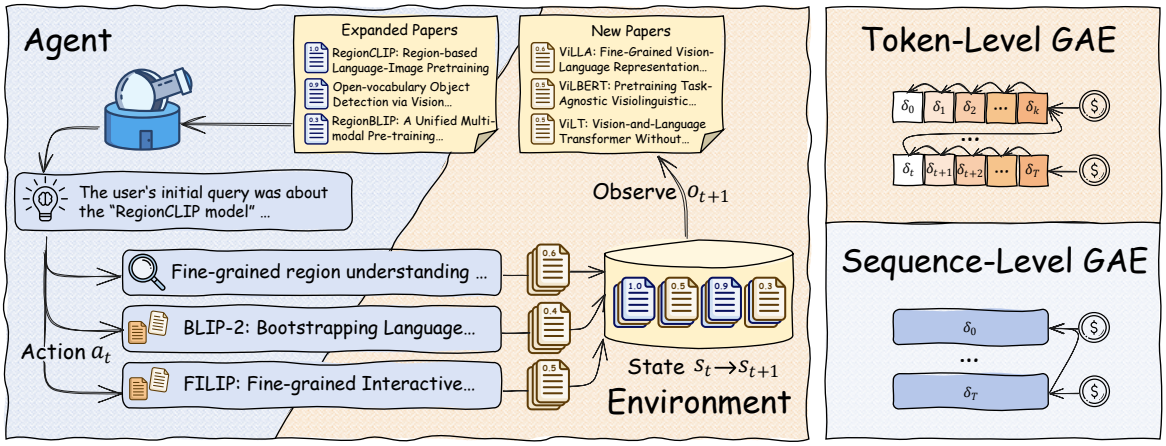


Figure 2: **Overview of PaperScout and the motivation for PSPO. Left:** PaperScout models multi-turn paper search as a POMDP by maintaining a paper pool as the latent state; at each step, the agent observes a summarized view of the pool and issues `Search` or `Expand` tool calls to retrieve new papers, which are then merged to update the pool. **Right:** *Granularity mismatch* in credit assignment: PPO assigns single step reward to many tokens, diluting supervision, whereas PSPO treats each complete response as an atomic action and optimizes at the sequence level.

x_t using a fixed prompt template together with the user query and tool description. The LLM then samples an output sequence $y_t \sim \pi_\theta(\cdot \mid x_t)$. We parse y_t into $y_t = (z_t, \mathcal{C}_t)$, where z_t is a brief reasoning trace and $\mathcal{C}_t$ is a set of tool invocations defined in Table 1. The environment executes $\mathcal{C}_t$ and returns raw results $\mathcal{D}(\mathcal{C}_t)$. We then filter these results to obtain the accepted candidate set $\mathcal{V}_t$:

$$\mathcal{V}_t = \{\, p \in \mathcal{D}(\mathcal{C}_t) \;\mid\; \rho(p) \geq \tau \;\wedge\; p \notin \mathcal{B}_t \}, \quad (1)$$

where τ is a minimum relevance threshold. Only papers in $\mathcal{V}_t$ are merged into the pool, yielding the state transition $s_t \rightarrow s_{t+1}$.

3.3 Proximal Sequence Policy Optimization

To effectively train the agent within the POMDP framework, we introduce Proximal Sequence Policy Optimization (PSPO). This section details the reward formulation and the core optimization algorithm designed to bridge the gap between token-level generation and sequence-level utility.

Reward Formulation. PaperScout aims to maximize the recall of relevant papers. We design the reward so that the expected return matches the expected number of relevant papers acquired over the search process. Let $\mathcal{V}_t$ denote the set of novel papers accepted into the pool at step t. To reduce variance, we compute the gain on the top-k papers in $\mathcal{V}_t$ ranked by relevance (using all papers if $|\mathcal{V}_t| < k$). The step reward is:

$$r_t = \underbrace{\sum_{p \in \text{top-}k(\mathcal{V}_t)} \rho(p)}_{\text{relevance gain}} - \underbrace{\eta \sum_{c \in \mathcal{C}_t} \mathbb{I}(c \in \mathcal{H}_{t-1})}_{\text{repetition penalty}}, \quad (2)$$

where $\rho(p) \in [0, 1]$ is the relevance score of paper p, interpreted as the probability that p is relevant to the user query. With this interpretation, the cumulative relevance gain corresponds to the expected number of relevant papers retrieved, while the repetition term discourages redundant tool usage.

Algorithm Implementation. In our setting, the agent interacts with the environment at the *sequence level*: each step corresponds to generating a complete response that triggers tool executions and yields a single step reward. This induces a granularity mismatch for token-level optimization, as illustrated in Figure 3: sparse sequence-level feedback must be attributed to tokens, leading to noisy credit assignment and unstable value learning.

PSPO resolves this mismatch by optimizing policies at the sequence level. Let x_t be the prompt obtained by serializing the observation o_t, and y_t be the complete response at step t. For a trajectory $\mathcal{T} = \langle x_0, y_0, \ldots, x_{T-1}, y_{T-1} \rangle$ of length T, we define $r_t = r(x_t, y_t)$ and learn a value function $V_\phi(x_t)$. We estimate advantages with GAE:

$$\hat{A}_t = \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \, \delta_{t+l}, \quad (3)$$

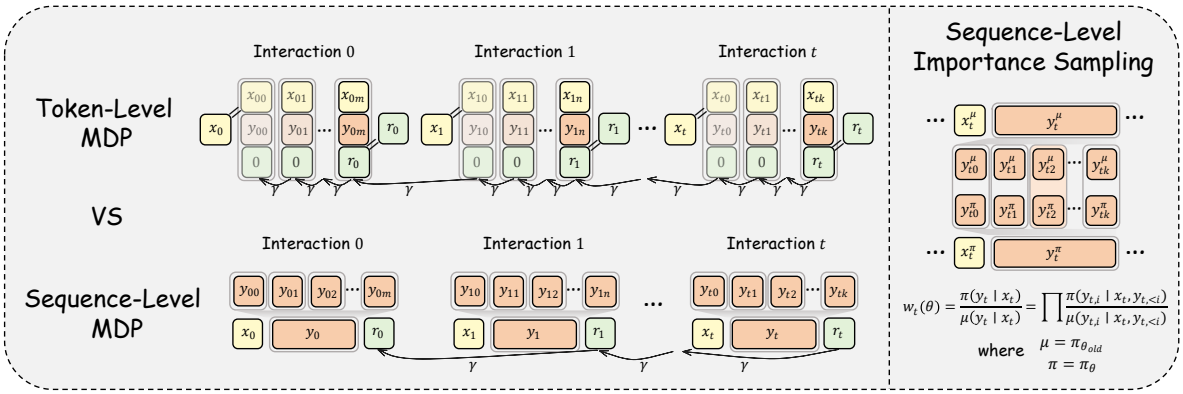


Figure 3: **PSPO optimizes policies at the sequence (interaction) level.** PPO assigns a single turn-level reward to many tokens in the generated response, which dilutes learning signals under token-level optimization. PSPO instead treats each complete response y_t as the atomic action in an interaction-level MDP, and performs advantage estimation and policy updates at the sequence level. This interaction-level formulation naturally leads to sequence-level importance sampling for off-policy correction.

where $\delta_t = r(x_t, y_t) + \gamma V_\phi(x_{t+1}) - V_\phi(x_t)$ denotes the temporal-difference (TD) error.

The actor is trained with a clipped surrogate objective function:

$$\mathcal{L}_{\text{actor}}(\theta) = \mathbb{E}_{\mathcal{T}\sim\pi_{\theta_{\text{old}}}}\Big[\min\big(w_t(\theta)\hat{A}_t, \text{clip}(w_t(\theta), 1-\varepsilon_{low}, 1+\varepsilon_{high})\hat{A}_t\big)\Big], \tag{4}$$

where the sequence-level importance ratio is

$$w_t(\theta) = \frac{\pi_\theta(y_t \mid x_t)}{\pi_{\theta_{\text{old}}}(y_t \mid x_t)} = \prod_{i=1}^{L_t} \frac{\pi_\theta(y_{t,i} \mid x_t, y_{t,<i})}{\pi_{\theta_{\text{old}}}(y_{t,i} \mid x_t, y_{t,<i})}. \tag{5}$$

The critic minimizes $\mathbb{E}\big[(V_\phi(x_t) - R_t)^2\big]$ with

$$R_t = \sum_{k=0}^{T-t-1} \gamma^k r_{t+k}. \tag{6}$$

4 Experiments

4.1 Experiment Setup

We provide a clear basis for our experiments, including the datasets, baselines, key experimental environments, and evaluation metrics.

Dataset Description. We conduct experiments on two benchmarks: AutoScholarQuery and RealScholarQuery (He et al., 2025), which represent synthetic and expert-curated scholarly search scenarios. AutoScholarQuery is a large-scale synthetic benchmark constructed from recent top-tier conference papers, including ICLR, ICML, NeurIPS, ACL, and CVPR, while RealScholarQuery is a human-annotated benchmark designed to evaluate realistic and challenging scholarly search settings. Detailed statistics of the datasets are summarized in Table 3.

For reinforcement learning, the agent is trained on the AutoScholarQuery training split and validated on the development split. We evaluate on two test sets: (i) a filtered subset of the AutoScholarQuery test set with 112 queries having at least five ground-truth relevant papers, and (ii) RealScholarQuery, which contains 50 real-world scholarly queries with reference papers.

Retrieval Backends. To enable stable RL training under high concurrency, we build a fully local retrieval environment for training. Specifically, we deploy Milvus[2] over millions of paper metadata to emulate a scholarly search service, and precache millions of papers from ar5iv[3] to support reference-based expansion. If a requested paper is missing from the local cache, it would be downloaded on demand from ar5iv or arXiv[4]. During evaluation, we standardize the search backend by issuing search calls through Google Search for fair comparison across methods.

Baselines. We compare against a diverse set of baselines spanning traditional academic search engines, LLM-enhanced retrieval workflows, and recent multi-turn paper search systems:

[2] https://milvus.io
[3] https://ar5iv.labs.arxiv.org
[4] https://arxiv.org

Table 2: Performance comparison on the RealScholarQuery and AutoScholarQuery benchmarks. Best results are highlighted in **bold** and second-best results are underlined. Note that PaperScout develops based on Qwen3-4B.

Model	Post-threshold			Recall@k				LLM-score
	Precision	F1	Recall	Recall@all	Recall@100	Recall@50	Recall@25	
RealScholarQuery								
Google Search	0.059	0.074	0.304	0.304	0.304	0.267	0.221	1.116
Google Search + LLM	0.067	0.077	0.254	0.254	0.254	0.216	0.202	1.237
Google Scholar	0.045	0.064	0.247	0.247	0.247	0.205	0.158	1.251
PaSa	0.415	0.417	0.541	0.598	0.591	0.572	0.517	2.111
SPAR	0.412	0.408	0.496	0.545	0.545	0.537	0.504	2.415
PaperScout-Qwen3-4B	0.404	0.411	0.497	0.545	0.543	0.535	0.498	2.261
PaperScout-Qwen3-Max	0.435	0.427	0.562	0.664	0.648	0.606	0.532	2.483
PaperScout	**0.442**	**0.441**	**0.574**	**0.691**	**0.683**	**0.646**	**0.577**	**2.576**
AutoScholarQuery								
Google Search	0.009	0.017	0.127	0.127	0.127	0.101	0.058	1.263
Google Search + LLM	0.010	0.019	0.138	0.138	0.138	0.115	0.081	1.322
Google Scholar	0.006	0.011	0.081	0.081	0.081	0.070	0.057	1.212
PaSa	0.095	0.129	0.442	0.699	0.579	0.463	0.344	2.186
SPAR	0.091	0.131	0.386	0.561	0.535	0.466	0.341	2.295
PaperScout-Qwen3-4B	0.092	0.128	0.382	0.564	0.540	0.448	0.324	2.202
PaperScout-Qwen3-Max	0.102	0.115	0.427	0.770	0.584	0.469	0.336	2.368
PaperScout	**0.115**	**0.134**	**0.459**	**0.811**	**0.636**	**0.496**	**0.357**	**2.467**

Dataset	#Train	#Dev	#Test
AutoScholarQuery	33551	1000	1000
RealScholarQuery	–	–	50

Table 3: Details of the two datasets.

- **Google Search** [5]**.** Standard Google Search using the original query.
- **Google Search + LLM.** Google Search with a refined query generated by LLM.
- **Google Scholar.** Direct retrieval from Google Scholar without LLM intervention.
- **PaSa (He et al., 2025).** A reinforcement learning–based paper search system that optimizes query rewriting and reference expansion under a predefined workflow.
- **SPAR (Shi et al., 2025).** A multi-turn academic search framework that integrates multiple retrieval components within a predefined structured workflow.

Our Variants. We evaluate three variants of our approach to disentangle the effects of RL fine-tuning and model scale: (i) PaperScout (Qwen3-4B fine-tuned with PSPO), (ii) PaperScout-Qwen3-4B (the same backbone without RL fine-tuning), and (iii) PaperScout-Qwen3-Max (a larger backbone, Qwen3-Max, without RL fine-tuning).

Evaluation Protocol. For Google Search, Google Search + LLM, and Google Scholar, we evaluate the top 100 retrieved papers. For other multi-turn methods, we use each system's reranked and filtered outputs. For PaperScout, candidate papers are ranked by relevance score $\rho(\cdot)$, and those with score $\rho(p) \geq 0.5$ are retained.

Metrics. We report Precision, Recall, F1-score, Recall@k, and an LLM-based relevance score. The post-threshold Precision, Recall, and F1-score are computed based on exact matches between the ground-truth relevant papers and the set of papers retained after applying a relevance threshold $\rho(p) \geq 0.5$. Recall@k measures the fraction of ground-truth relevant papers appearing within the top-k ranked results.

To mitigate evaluation bias caused by incomplete annotations, we additionally report an LLM-based relevance score. Three LLMs (DeepSeek-V3.2 (Liu et al., 2024), Qwen3-Max (Yang et al., 2025), and GPT-5.1) independently rate each retrieved paper–query pair on a four-level scale (0–3), where higher scores indicate greater relevance. The final score is the average of the three scores. Details of the scoring protocol and reliability analysis are provided in Appendix B.

[5] https://serper.dev

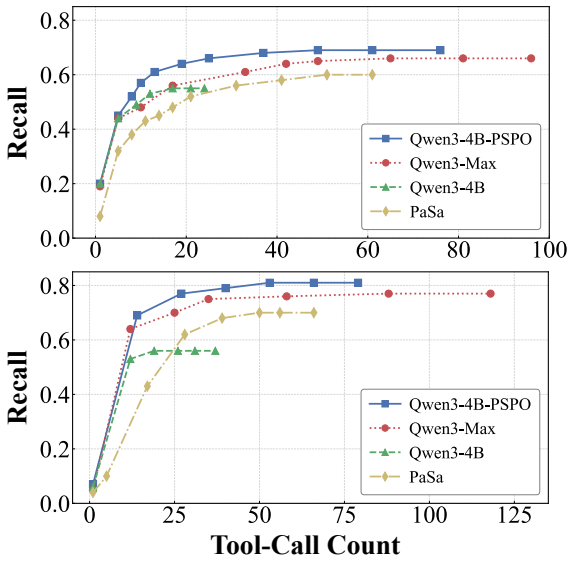


Figure 4: Average recall as tool calls accumulate on RealScholarQuery (top) and AutoScholarQuery (bottom). PSPO consistently yields higher recall with the same number of tool calls.

4.2 Main Results

Table 2 summarizes the performance on RealScholarQuery and AutoScholarQuery. Across both benchmarks, multi-turn retrieval methods substantially outperform single-shot baselines, highlighting the importance of iterative tool use for academic paper search under complex query settings.

Among all multi-turn systems, PaperScout consistently achieves the strongest performance. On RealScholarQuery, PaperScout attains the highest Recall of 0.574 (vs. 0.541 for the best baseline) together with the best LLM-score of 2.576, and improves the best baseline F1-score from 0.417 to 0.441 (+5.8% relative). PaperScout also achieves the best Recall@k across all cutoffs, indicating stronger ranking quality of relevant papers. On AutoScholarQuery, PaperScout again ranks first, yielding the highest Recall of 0.459 and highest LLM-score of 2.467, while also achieving the best F1-score (0.134). Notably, the RL-trained 4B PaperScout matches or surpasses a larger untrained backbone (PaperScout-Qwen3-Max), motivating further analysis of its tool-call efficiency and optimization behavior in the following sections.

4.3 Analysis of PaperScout

To analyze the strong retrieval performance of PaperScout, we examine its behavior from the perspectives of tool-call efficiency and flexibility in multi-turn retrieval.

Table 4: Performance comparison of PaperScout optimized by PPO, GSPO and PSPO on RealScholarQuery.

Model	Precision	F1	Recall	LLM-score
PPO	0.405	0.408	0.537	2.417
GSPO	0.433	0.439	0.557	2.510
PSPO	**0.442**	**0.441**	**0.574**	**2.576**

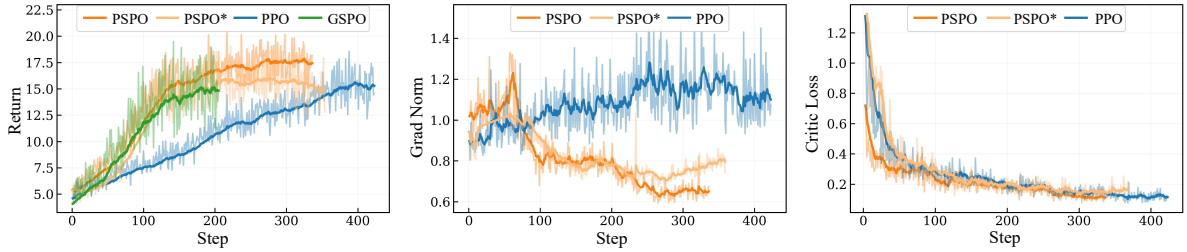
Tool-Call Efficiency. Figure 4 shows how recall evolves as tool calls accumulate on both benchmarks. PaperScout consistently achieves higher recall than all baselines under the same number of tool calls, indicating greater retrieval gain per invocation. In the low-call regime, the trained PaperScout agent exhibits a steeper recall increase than PaSa and the untrained 4B variant, suggesting more effective early-stage retrieval. As tool calls increase, the untrained 4B agent quickly saturates, whereas PaperScout continues to accumulate relevant papers and maintains a clear advantage. Notably, PaperScout matches or surpasses the much larger Qwen3-Max backbone across a wide range of tool-call counts on both datasets, indicating that efficient multi-turn retrieval can compensate for model scale through more informative tool usage.

Analysis of Tool Distribution. Figure 6 analyzes how PaperScout allocates `Search` and `Expand` tool calls across queries. The untrained Qwen3-4B agent concentrates near the origin, indicating few tool calls and limited multi-turn exploration. In contrast, Qwen3-Max shows a strong skew toward `Expand`, with many trajectories dominated by expansion-heavy behavior and relatively few `Search` calls, restricting new retrieval directions. PaperScout instead occupies a broader region of the `Search`–`Expand` space and distributes calls more evenly between the two actions, reflecting a better balance between retrieval breadth and depth. Overall, these patterns suggest that PaperScout enables more flexible multi-turn retrieval rather than fixed or biased tool-use behaviors.

4.4 Effectiveness of PSPO

To evaluate PSPO for multi-turn retrieval optimization, we analyze its impact on retrieval quality and optimization stability, compared with PPO, GSPO, and PSPO* (without process rewards).

Superior Retrieval Performance. Table 4 reports the retrieval performance on RealScholarQuery. PSPO achieves the best results across all metrics, improving Recall to 0.574 (vs. 0.557 with



(a) Return of Trajectory (b) Actor Gradient Norm (c) Critic Loss

Figure 5: Training dynamics and optimization statistics for different policy optimization methods. (a) Trajectory returns. PSPO converges faster and reaches a higher plateau than PPO, GSPO, and PSPO* (PSPO without process rewards). (b) Actor gradient norm during policy updates. PSPO shows smaller gradients with a clearer downward trend than PPO. (c) Critic loss. PSPO is consistently lower than PPO and PSPO*, particularly in early training.

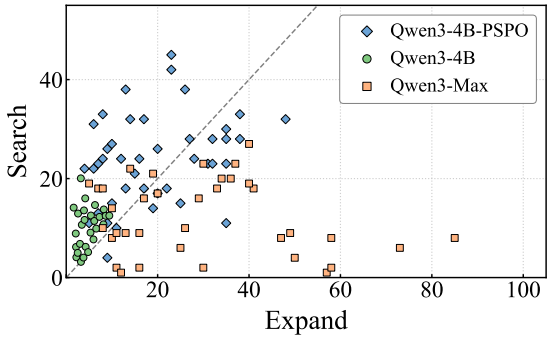


Figure 6: Tool call distribution across queries for different models. The trained PaperScout (Qwen3-4B-PSPO) shows a broader, more balanced allocation of `Search` and `Expand` calls than the untrained Qwen3-4B and Qwen3-Max, indicating flexible multi-turn retrieval.

GSPO and 0.537 with PPO) and yielding the highest LLM-score of 2.576. These gains are consistent with the training dynamics shown in Figure 5(a), where PSPO converges faster and reaches a higher return plateau than PPO, GSPO, and PSPO*.

The slower improvement of PPO stems from the granularity mismatch in multi-turn retrieval: token-level optimization under sparse sequence-level feedback introduces noisy credit assignment and difficult value fitting, reducing sample efficiency despite smooth learning curves. GSPO performs sequence-level optimization and achieves rapid early gains but plateaus earlier, largely due to its inability to effectively leverage process rewards, relying instead on coarse trajectory-level signals that hinder credit assignment across intermediate decisions. PSPO bridges these behaviors by aligning optimization with sequence-level interaction while leveraging process rewards through the critic, resulting in faster convergence and more stable improvements. In contrast, PSPO* aggregates process rewards into the final trajectory reward, preserving the total reward but yielding a lower return plateau, suggesting that step-wise rewards provide useful intermediate learning signals.

Stable Model Optimization. Figure 5(b) presents the actor gradient norm during training. Compared with PPO, PSPO maintains a smaller gradient norm with a clear downward trend, indicating more controlled and stable policy updates under sequence-level optimization. Figure 5(c) further shows that PSPO consistently achieves a lower critic loss than both PPO and PSPO*, particularly in the early training stage. This suggests that step-wise process rewards provide more informative learning targets for value regression, whereas removing intermediate rewards weakens credit assignment and makes value estimation more difficult under sparse sequence-level feedback. Overall, PSPO improves optimization stability for both the actor and critic, leading to stronger and more reliable retrieval performance.

5 Conclusion

In this paper, we present PaperScout, an autonomous agent that formulates academic paper search as a sequential decision-making process. By explicitly deciding when and how to invoke `search` and `expand` tools based on accumulated retrieval context, PaperScout moves beyond static workflows and enables flexible multi-turn paper exploration. We further introduce Proximal Sequence Policy Optimization (PSPO), a process-aware, sequence-level policy optimization method that aligns optimization with agent–environment interaction and enables stable training of multi-turn retrieval agents. Experiments on both synthetic and real-world benchmarks demonstrate that PaperScout trained by PSPO achieves superior retrieval performance compared to existing methods.

Limitations

Despite the effectiveness of our approach in multi-turn paper search, several limitations remain. First, our current evaluation primarily focuses on the computer science domain, and extending the study to broader research areas would further validate the generality of the proposed framework. Second, PaperScout relies on papers that are publicly accessible through online search engines and open repositories; retrieving papers from restricted or paywalled sources remains challenging and may limit coverage in certain domains. Third, our current implementation adopts a relatively limited search backend, and incorporating multi-source retrieval from heterogeneous scholarly databases could further improve robustness and recall. Finally, citation expansion currently considers only outgoing references, while leveraging incoming citations and richer citation graph signals may provide additional retrieval cues. We leave these directions for future work.

References

Ahmet Yasin Aytar, Kamer Kaya, and Kemal Kılıç. 2025. A synergistic multi-stage rag architecture for boosting context relevance in data science literature. *Natural Language Processing Journal*, page 100179.

Mingyue Cheng, Jie Ouyang, Shuo Yu, Ruiran Yan, Yucong Luo, Zirui Liu, Daoyu Wang, Qi Liu, and Enhong Chen. 2025. Agent-r1: Training powerful llm agents with end-to-end reinforcement learning. *arXiv preprint arXiv:2511.14460*.

Yuxiao Cheng and Jinli Suo. 2025. Openlens ai: Fully autonomous research agent for health infomatics. *arXiv preprint arXiv:2509.14778*.

Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W Bruce Croft, and Xueqi Cheng. 2020. A deep look into neural ranking models for information retrieval. *Information Processing & Management*, 57(6):102067.

Michael Gusenbauer and Neal R Haddaway. 2020. Which academic search systems are suitable for systematic reviews or meta-analyses? evaluating retrieval qualities of google scholar, pubmed, and 26 other resources. *Research synthesis methods*, 11(2):181–217.

Michael Gusenbauer and Neal R Haddaway. 2021. What every researcher should know about searching–clarified concepts, search advice, and an agenda to improve finding in academia. *Research synthesis methods*, 12(2):136–147.

Yichen He, Guanhua Huang, Peiyuan Feng, Yuan Lin, Yuchen Zhang, Hang Li, and 1 others. 2025. Pasa: An llm agent for comprehensive academic paper search. *arXiv preprint arXiv:2501.10120*.

Chuang Jiang, Mingyue Cheng, Xiaoyu Tao, Qingyang Mao, Jie Ouyang, and Qi Liu. 2025. Tablemind: An autonomous programmatic agent for tool-augmented table reasoning. *arXiv preprint arXiv:2509.06278*.

Jyun-Yu Jiang, Mingyang Zhang, Cheng Li, Michael Bendersky, Nadav Golbandi, and Marc Najork. 2019. Semantic text matching for long-form documents. In *The world wide web conference*, pages 795–806.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. 2025. Webthinker: Empowering large reasoning models with deep research capability. *arXiv preprint arXiv:2504.21776*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Zhiding Liu, Ben Chen, Mingyue Cheng, Enhong Chen, Li Li, Chenyi Lei, Wenwu Ou, Han Li, and Kun Gai. 2025. Towards context-aware reasoning-enhanced generative searching in e-commerce. *arXiv preprint arXiv:2510.16925*.

Xufang Luo, Yuge Zhang, Zhiyuan He, Zilong Wang, Siyun Zhao, Dongsheng Li, Luna K Qiu, and Yuqing Yang. 2025a. Agent lightning: Train any ai agents with reinforcement learning. *arXiv preprint arXiv:2508.03680*.

Yucong Luo, Yitong Zhou, Mingyue Cheng, Jiahao Wang, Daoyu Wang, Tingyue Pan, and Jintao Zhang. 2025b. Time series forecasting as reasoning: A slow-thinking approach with reinforced llms. *arXiv preprint arXiv:2506.10630*.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315.

Paolo Manghi. 2024. Challenges in building scholarly knowledge graphs for research assessment in open science. *Quantitative Science Studies*, 5(4):991–1021.

Gary Marchionini. 2006. Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46.

Yichen Ouyang, Lu Wang, Fangkai Yang, Pu Zhao, Chenghua Huang, Jianfeng Liu, Bochen Pang, Yaming Yang, Yuefeng Zhan, Hao Sun, and 1 others. 2025. Token-level proximal policy optimization for query generation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 31184–31198.

Sangwoo Park, Jinheon Baek, Soyeong Jeong, and Sung Ju Hwang. 2025. Chain of retrieval: Multi-aspect iterative search expansion and post-order search aggregation for full paper retrieval. *arXiv preprint arXiv:2507.10057*.

Shahzad Qaiser and Ramsha Ali. 2018. Text mining: use of tf-idf to examine the relevance of words to documents. *International journal of computer applications*, 181(1):25–29.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Xiaofeng Shi, Yuduo Li, Qian Kou, Longbin Yu, Jinxin Xie, and Hua Zhou. 2025. Spar: Scholar paper retrieval with llm-based agents for enhanced academic search. *arXiv preprint arXiv:2507.15245*.

Fiona Timmins and Catherine McCabe. 2005. How to conduct an effective literature search. *Nursing standard*, 20(11):41–47.

Rita Vine. 2006. Google scholar. *Journal of the Medical Library Association*, 94(1):97.

Daoyu Wang, Mingyue Cheng, Shuo Yu, Zirui Liu, Ze Guo, and Qi Liu. 2025a. Paperarena: An evaluation benchmark for tool-augmented agentic reasoning on scientific literature. *arXiv preprint arXiv:2510.10909*.

Jiahao Wang, Mingyue Cheng, and Qi Liu. 2025b. Can slow-thinking llms reason over time? empirical studies in time series forecasting. *arXiv preprint arXiv:2505.24511*.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. 2025c. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *Preprint*, arXiv:2504.20073.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yunyi Zhang, Ruozhen Yang, Siqi Jiao, SeongKu Kang, and Jiawei Han. 2025. Scientific paper retrieval with llm-guided semantic-based ranking. *arXiv preprint arXiv:2505.21815*.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, and 1 others. 2025. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*.

A Experimental Details

A.1 Implementation of PaperScout

We conduct reinforcement learning for PaperScout on four NVIDIA H800 GPUs, using Qwen3-4B-Instruct-2507[6] as the backbone model. We set the actor and critic learning rates to 1×10^{-6} and 1×10^{-5}, respectively, with a per-gpu batch size of 4. For reward computation, we set the top-k to 3 to reduce reward variance across queries with varying difficulty. We further set the score threshold $\tau = 0.01$ and repeated penalty $\eta = 0.5$. To improve training stability, we first freeze the actor and pre-train the critic for 100 steps, followed by joint optimization of both components. The agent receives up to 10 unexpanded and 10 expanded papers at each turn. If the paper pool remains unchanged for three turns, the retrieval terminates. More training details are shown in Table 5.

Scorer. Given a user query together with the title and abstract of a paper, the scorer evaluates their semantic relevance and produces a continuous score that guides the retrieval decision of PaperScout. Specifically, the scorer prompts a language model to perform a relevance judgment, and defines the relevance score as the probability of generating the

[6] https://www.modelscope.cn/models/Qwen/Qwen3-4B-Instruct-2507

Category	Hyperparameter	Value
Optimization	Actor learning rate	1×10^{-6}
	Critic learning rate	1×10^{-5}
	Policy freezing steps	100
	Total training steps	300
RL Hyperparameters	Advantage estimator	GAE
	Discount factor γ	0.99
	GAE parameter λ	0.95
	Clip ratio (ε_{low})	3×10^{-4}
	Clip ratio (ε_{high})	4×10^{-4}
	KL coefficient	0.001
Batching	Train batch size	128
	Actor mini-batch size	320
	Critic mini-batch size	128
	Actor micro-batch size	4
	Critic micro-batch size	8
Sequence Length	Max prompt length	10,240
	Max response length	4,096

Table 5: Hyperparameters used for PSPO training.

"True" token in the model output, yielding a value in $[0, 1]$. For this component, we utilize the off-the-shelf pasa-7b-selector[7], a Qwen2.5-7B based model optimized for relevance assessment. The detailed prompt is shown in Appendix F.

B Details about LLM-score

Evaluation Protocol. We employ an LLM-as-a-Judge to assess the relevance between a user query and a candidate paper based on its title and abstract, using three independent evaluators—DeepSeek-V3.2, Qwen3-Max, and GPT-5.1. To ensure reliable and unbiased judgments, the evaluators are explicitly instructed to focus solely on semantic relevance, while ignoring factors such as writing quality, length, style, or popularity. Relevance is rated on a discrete four-level scale from 0 to 3, with clearly defined semantic criteria for each level, which helps constrain the models' decision boundaries and reduce ambiguity in scoring. The evaluators are further required to output results in a strict JSON format, ensuring consistent, structured, and easily parsable relevance scores for downstream analysis. The detailed evaluation prompt is provided in Appendix F.

Inter-model Consistency. We randomly sample 500 query–paper pairs from each method's test results, yielding a total of 3,000 samples, and analyze the consistency of the three LLM-based relevance scores for each sample. For each query–paper pair, we compute the standard deviation (STD) of the three scores as well as their maximum score gap (Max-Gap) to quantify inter-model agreement. As shown in Table 6, most samples exhibit strong consistency: 2,629 cases receive identical scores from all three models. The remaining discrepancies are predominantly minor, with only small score differences, while larger disagreements are rare and no samples exhibit extreme conflicts. These results indicate that the LLM-based scoring provides a stable and reliable signal for relevance evaluation.

STD	0.00	0.22	0.66	0.88	> 0.88
Max-Gap	0	1	2	2	3
Count	2,629	336	25	10	0

Table 6: Consistency analysis of LLM-based relevance scores across three evaluators.

Agreement with Human Judgment. To assess whether LLM-based relevance scoring is aligned with human judgment, we further compare LLM-assigned scores with human annotations. Specifically, we randomly sample QA-pairs and compare their relevance scores on the same four-level scale. To ensure comparability, human annotators are instructed to follow the same scoring criteria as the LLM evaluators, as specified in Scoring Rubric F.

LLM Score	**Human Score**				**Row Total**
	0	**1**	**2**	**3**	
0	**88**	11	1	0	100
1	0	**82**	15	3	100
2	0	8	**87**	5	100
3	0	1	5	**94**	100
Col Total	88	102	108	102	400

Table 7: Distribution of LLM-based versus human relevance scores. Bold numbers indicate exact agreement.

As shown in Table 7, LLM-based scores are highly consistent with human annotations, with most samples falling on the diagonal. The agreement is especially strong for clear cases: 88% of samples assigned a score of '0' by LLMs are also labeled as '0' by humans, and 94% of those assigned '3' are also labeled as '3'. Most remaining discrepancies occur between adjacent scores, mainly around intermediate levels such as '1' and '2', suggesting that disagreements largely come from borderline cases rather than severe misjudgments. Overall, LLM-based scoring is well aligned with human judgment and provides a reliable relevance signal for our evaluation.

[7] https://www.modelscope.cn/models/bytedance-research/pasa-7b-selector

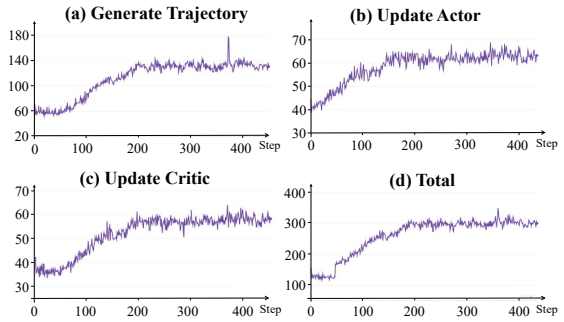


Figure 7: Per-step training time consumption (in seconds) of PaperScout. (a) Rollout phase; (b) Actor update; (c) Critic update; (d) Total time consumption.

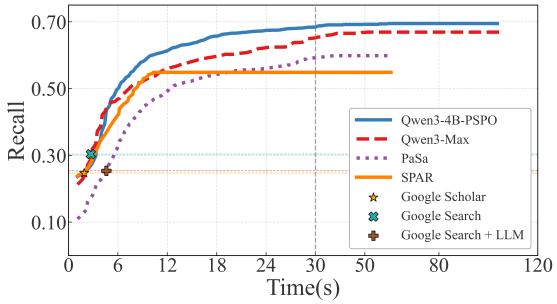


Figure 8: Recall–time curves on RealScholarQuery. PaperScout reaches near-optimal recall within 30 seconds, outperforming both agent-based and traditional retrieval baselines over runtime.

C Computational Cost and Efficiency

We analyze the computational cost of training and the runtime efficiency of PaperScout with PSPO.

Training Cost. PSPO training is conducted on four NVIDIA H800 (80GB) GPUs, consisting of 100 critic pretraining steps followed by 200 joint actor–critic optimization steps, with a total training time of approximately 12.6 hours (≈4,000s for critic pretraining and ≈39,750s for PSPO training). As shown in Figure 7, after the warm-up phase the per-step training time stabilizes at about 290–310 seconds. Among the components, rollout dominates the cost due to multi-turn interaction with the retrieval environment, requiring approximately 135–145 seconds per step, while actor and critic updates take about 60–65 seconds and 55–60 seconds, respectively. Overall, policy optimization accounts for less than half of the step time, indicating that most computational overhead arises from environment interaction rather than policy updates.

Inference Efficiency. Beyond training cost, we also examine whether the learned policy leads to efficient inference behavior. Figure 8 presents recall as a function of runtime on RealScholarQuery, reflecting how much retrieval gain each system obtains per unit time. Compared with other agent-based baselines, including Qwen3-Max, PaSa, and SPAR, Qwen3-4B-PSPO achieves faster recall improvement and higher final recall under similar or shorter time budgets, showing that PSPO improves not only final effectiveness but also the efficiency of the retrieval process.

Moreover, despite requiring multi-turn reasoning and repeated tool interaction, Qwen3-4B-PSPO remains competitive with conventional retrieval systems such as Google Scholar and Google Search in terms of recall under comparable runtime budgets. This indicates that the additional interaction cost introduced by the agent framework is effectively offset by more informed and productive search decisions. In other words, while PSPO incurs a moderate one-time training cost, it yields a policy with substantially improved inference efficiency, leading to a better effectiveness–efficiency trade-off overall.

D Case Study

D.1 Case: Adaptive Tool-call Behavior

Figure 9 presents a representative case illustrating PaperScout's multi-turn retrieval behavior on a complex query involving CLIP, ViLD, and RegionCLIP. At the initial stage, the agent decomposes query into multiple semantic facets and issues parallel `Search` tool calls targeting different aspects of vision–language models. Based on the retrieved seed papers, the agent performs several rounds of `Expand` tool calls, progressively exploring region-level and fine-grained vision–language pretraining methods with strong semantic coherence.

Notably, after expansion along existing directions becomes saturated, the agent initiates a new `Search` call in a later turn, opening a previously unexplored research direction, fine-grained region understanding. This re-search step moves the agent beyond local expansion by introducing fresh candidate papers, which are further refined through subsequent `Expand` calls. Such behavior demonstrates that PaperScout does not follow a fixed search–then–expand pipeline, but instead automatically alternates between `Search` and `Expand` based on accumulated context, constructing a structured and evolving paper exploration trajectory.

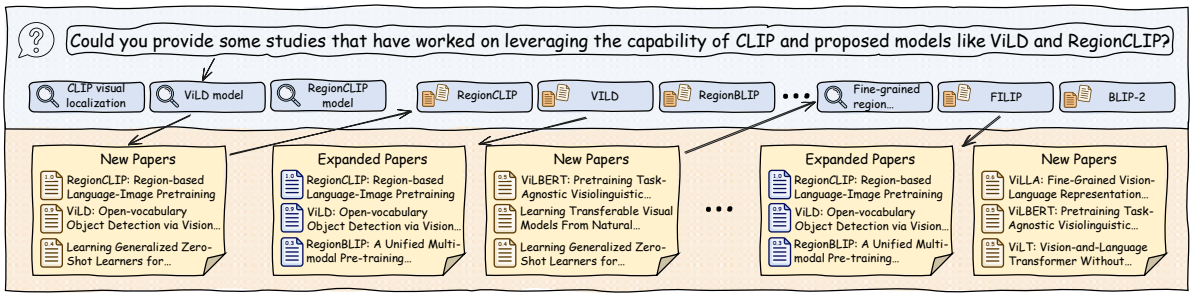


Figure 9: Case study of PaperScout. The agent alternates between `Search` and `Expand` based on accumulated retrieval context, re-issuing `Search` when `Expand` yields diminishing returns to introduce new directions.

D.2 Case: Comparison with Other Agents

To further understand the gains of PaperScout over structured multi-turn baselines such as PaSa and SPAR, we summarize its tool-use patterns and retrieval performance on representative RealScholarQuery cases in Table 8. Beyond the single case discussed above, this analysis reveals common patterns across query types. PaperScout shows larger advantages on enumerative and multi-constraint queries, where relevant papers are often scattered across different terminologies, subfields, and citation neighborhoods.

This pattern can be understood from the agents' different retrieval strategies. PaSa learns to invoke search, expansion, and stop actions, but its behavior is still organized around a paper-queue traversal protocol: it first collects candidate papers and then expands their citation neighborhoods. SPAR improves retrieval through a structured modular pipeline, mainly strengthening query understanding and retrieval enhancement. By contrast, PaperScout dynamically controls retrieval according to the evolving candidate pool. Rather than following a prescribed traversal pattern, it can continue searching, expand promising seed papers, or reformulate the query when current results are insufficient. This flexibility helps explain its higher Recall@all: PaperScout can decompose long constrained queries into searchable facets and progressively explore different sub-domains for enumerative queries. As a result, it is especially effective for academic paper discovery, where the goal is to build a more complete set of relevant literature.

E Discussion

E.1 Distinction from General Search Agents

PaperScout differs from general search or deep research agents such as Search-R1 (Jin et al., 2025) and WebThinker (Li et al., 2025) in task structure and optimization objective. While these agents usually retrieve information to support a final textual answer or report, academic paper search takes the retrieved paper set itself as the target output. The task therefore directly emphasizes relevance, coverage, and ranking quality, and naturally provides verifiable process feedback: each Search or Expand action updates the paper pool and yields measurable marginal retrieval gains. This structure makes paper search particularly suitable for multi-turn decision-making, process-level rewards, and interaction-level optimization.

E.2 Discussion on the Difference from PaSa

PaSa and PaperScout differ mainly in the level at which retrieval control is learned. PaSa is an agentic system: its Crawler can generate search queries, expand references from the current paper, or stop processing, and these behaviors are optimized with reinforcement learning. However, these decisions are made within a predefined paper-queue traversal protocol, where the system first accumulates candidate papers and then processes queued papers through local expansion or stopping decisions. Thus, PaSa mainly learns local actions inside a fixed interaction scaffold, and its retrieval behavior is naturally biased toward search-then-expand citation traversal. PaperScout instead treats paper search as an autonomous sequential decision-making process over an evolving paper pool. At each step, the agent can decide whether to broaden the search with a new query, deepen the search by expanding promising seed papers, or shift exploration direction according to the current retrieval context. This allows PaperScout to adaptively balance search breadth and expansion depth based on each action's marginal gain to the paper pool.

Query	Details	Tool-use Strategy of PaperScout
Help me search for the work related to the synthetic data of large language models. I want to know how to automatically generate large-scale, high-quality, diverse, difficult, and valuable long thought data for learning.	**Action/ Search/ Expand cnt. (@PaperScout):** 15 / 60 / 10 **Recall@all (@PaperScout/ PaSa/ SPAR):** 1.0000 / 0.0000 / 0.0000	First searches from multiple angles, including synthetic data, long-thought data, and high-quality, diverse, and difficult data. After finding `DeepSeek-Prover`, it expands related seed papers and recovers `MUSTARD`.
I am looking for research papers on the construction of multimodal foundation models that support both visual and audio inputs. These models should be pre-trained on large-scale datasets, including visual, audio, and audio-visual data. Please exclude survey papers.	**Action/ Search/ Expand cnt. (@PaperScout):** 9 / 19 / 20 **Recall@all (@PaperScout/ PaSa/ SPAR):** 0.7381 / 0.6667 / 0.4524	Starts with broad multimodal and audio-visual foundation-model searches, then heavily expands representative papers, which fits high-recall, multi-constraint queries.
Give me papers about how to rank search results by the use of LLM.	**Action/ Search/ Expand cnt. (@PaperScout):** 14 / 46 / 12 **Recall@all (@PaperScout/ PaSa/ SPAR):** 0.6410 / 0.4359 / 0.3846	Searches LLM ranking and reranking first, then expands core papers on listwise ranking and zero-shot rankers. Later searches are refined toward fine-tuning, self-consistency, setwise prompting, and ranking losses.
Show me research on the long video description. Here, long videos are defined as those with a duration of at least several minutes.	**Action/ Search/ Expand cnt. (@PaperScout):** 15 / 52 / 10 **Recall@all (@PaperScout/ PaSa/ SPAR):** 0.6286 / 0.5143 / 0.3429	Begins with long video description and multi-minute video queries, then refines toward temporal understanding, hierarchical event modeling, episodic memory, scene graphs, and long-form video summarization.
Provide papers on methods that protect the generation quality of LLMs under vocabulary watermarking settings.	**Action/ Search/ Expand cnt. (@PaperScout):** 11 / 28 / 21 **Recall@all (@PaperScout/ PaSa/ SPAR):** 0.7297 / 0.5676 / 0.5676	Uses many Expand calls around core watermarking papers, then supplements them with searches for semantic-aware, adaptive, undetectable, and robust watermarking under quality-preservation constraints.
Give me all visual-LLM models that are MoE architecture.	**Action/ Search/ Expand cnt. (@PaperScout):** 14 / 40 / 20 **Recall@all (@PaperScout/ PaSa/ SPAR):** 0.8500 / 0.8000 / 0.5500	Starts with visual-LLM and MoE architecture searches, expands VLM/MoE seed papers, then explores application subdomains such as medical imaging, robotics perception, autonomous driving, and real-world scene understanding.
Papers that propose methods based on large language models and evaluate their performance through experiments on the HotPotQA dataset.	**Action/ Search/ Expand cnt. (@PaperScout):** 18 / 63 / 12 **Recall@all (@PaperScout/ PaSa/ SPAR):** 0.5417 / 0.4167 / 0.4583	Repeatedly rewrites the query around HotPotQA, multi-hop reasoning, retrieval, chain-of-thought, external knowledge, performance metrics, and error analysis, with occasional Expand calls on relevant seed papers.

Table 8: Tool-use analysis and retrieval performance on representative RealScholarQuery cases, where PaperScout shows clear advantages over PaSa and SPAR.

F Full Prompt Set

Paper Search Agent Prompt

System

You are a research agent. Your goal is to find papers relevant to the User Query.

User Query

(...Detailed User Query...)

History Actions

(...Detailed History Actions...)

Paper List

(...Detailed Paper List...)

Instructions

- Analyze the Paper List and History Actions to determine the next set of actions. Enclose your analysis of the state and decision logic within `<analysis>...</analysis>` tags.
- You support parallel tool calling. You should output multiple tool calls in a single step if several independent actions are valuable at the current state.
- Attend to the history actions and avoid expanding the same papers.

Output Format

```
<analysis>
[Your analysis of the current state and decision logic...]
</analysis>
<tool_call>
[Tool call 1]
</tool_call>
```

```
<tool_call>
[Tool call 2]
</tool_call>
...
```

Tool Schema

```
SEARCH_TOOL_SCHEMA = {
    "type": "function",
    "function": {
        "name": "search",
        "description": (
            "Search for relevant papers in the arXiv repository."
        ),
        "parameters": {
            "type": "object",
            "properties": {
                "query": {
                    "type": "string",
                    "description": (
                        "A single search query (natural language or keywords). "
                        "No field scopes (ti:/abs:) or boolean ops. Must differ from all
                        history queries."
                    ),
                }
            },
            "required": ["query"],
        },
    },
}

EXPAND_TOOL_SCHEMA = {
    "type": "function",
    "function": {
        "name": "expand",
        "description": (
            "Expand from an existing paper by following its references to surface additional
            relevant works. "
            "Use this when search is saturated and you want to broaden coverage around a known
            paper."
        ),
        "parameters": {
            "type": "object",
            "properties": {
                "arxiv_id": {
                    "type": "string",
                    "description": (
                        "The arXiv identifier (e.g., '1706.03762') of a paper already in the
                        current paper list."
                    ),
                }
            },
            "required": ["arxiv_id"],
        },
    },
}

PAPERSEARCH_TOOL_SCHEMAS = [SEARCH_TOOL_SCHEMA, EXPAND_TOOL_SCHEMA]
```

Scorer Prompt

System

You are an elite researcher in the field of AI, conducting research on

(...Detailed User Query...)

Evaluate whether the following paper fully satisfies the detailed requirements of the user query and provide your reasoning. Ensure that your decision and reasoning are consistent.

Searched Paper

- Title: (...Detailed Title...)
- Abstract: (...Detailed Abstract...)
- User Query: (...Detailed User Query...)

Output format

- Decision: True / False
- Reason: ...

LLM Judge Prompt

System

You are an expert LLM-as-a-Judge for academic paper search. Your task is to evaluate how relevant a candidate paper is to a given research query. You must strictly follow the scoring rubric and output only the required JSON. You should ignore writing quality, length, style, or popularity. Only evaluate semantic relevance between the query and the paper content.

Query

(...Detailed User Query...)

Paper Candidate

- Title:
- Abstract:

Scoring Rubric (relevance_level)

- - 3 = Highly relevant — Directly addresses the core research question or problem expressed in the query.
- - 2 = Moderately relevant — About a closely related topic, but only partially matches the intent or focuses on a sub-aspect.
- - 1 = Weakly relevant — Belongs to the broader area but only tangentially connected to the query.
- - 0 = Irrelevant — Not relevant to the query in any meaningful way.

Output Format (must be valid JSON)

```
{
  "relevance_level": 0/1/2/3,
  "confidence": "low/medium/high",
  "reason_short": "A brief one-sentence justification"
}
```

You MUST output valid JSON and nothing else.

G An Example for PaperScout

Query

Could you provide some studies that have worked on leveraging the capability of CLIP and proposed models like ViLD and RegionCLIP?

Step 1

Paper List:

Empty list

Analysis:

The initial paper list is empty, and there are no history actions to avoid. The user is interested in studies that leverage CLIP's capabilities, specifically referencing models like ViLD and RegionCLIP. Since there are no existing papers in the pool, we must first search for relevant papers that discuss CLIP-based models or specifically mention ViLD and RegionCLIP. To ensure comprehensive coverage, we should perform a broad search query with keywords such as "CLIP", "ViLD", and "RegionCLIP" to identify foundational and related works. Given that the user is interested in the application and development of CLIP-based models like ViLD and RegionCLIP, initiating multiple search queries with slightly different but complementary keywords (e.g., "CLIP-based visual localization", "ViLD model", "RegionCLIP model") will help capture a diverse range of relevant papers. This ensures that we cover both general CLIP applications and the specific models mentioned in the query. We will perform three initial search queries: "CLIP visual localization" "ViLD model" "RegionCLIP model" These queries will yield a set of foundational and relevant papers directly matching the user's interests, enabling us to form a robust initial pool to proceed with further expansion or refinement.

Tool Call:

```
{"name": "search", "arguments": {"query": "CLIP visual localization"}}
```

```
{"name": "search", "arguments": {"query": "ViLD model"}}
{"name": "search", "arguments": {"query": "RegionCLIP model"}}
```

Step 2

Paper List:

Paper Pool Status:

[EXP]: Paper has been expanded (already used as a seed for more papers).

[NEW]: New paper found via search or expansion, candidate for further exploration.

Format: [arxiv_id] (score) [STATUS] Title

[2112.09106] (1.00) [NEW] RegionCLIP: Region-based Language-Image Pretraining

Abstract: Contrastive language-image pretraining (CLIP) performs poorly on region-level recognition due to a domain shift between whole-image and region-text alignment. We propose RegionCLIP to learn region-level visual representations aligned with textual concepts, significantly improving open-vocabulary object detection performance. ...

[2104.13921] (0.99) [NEW] Open-vocabulary Object Detection via Vision and Language Knowledge Distillation

Abstract: We propose ViLD, a vision-language knowledge distillation framework that transfers open-vocabulary knowledge from a pretrained image-text model to a two-stage detector. ViLD achieves state-of-the-art performance on LVIS and generalizes well to COCO, PASCAL VOC, and Objects365 without finetuning. ...

[2302.00275] (0.40) [NEW] Learning Generalized Zero-Shot Learners for Open-Domain Image Geolocalization

Abstract: We introduce StreetCLIP, a foundation model that achieves state-of-the-art zero-shot performance for open-domain image geolocalization. By meta-learning with synthetic captions, our method effectively transfers CLIP's generalized zero-shot capability to the geolocalization domain. ...

(...More Papers...)

[2305.05095] (0.12) [NEW] Less is More: Removing Text-regions Improves CLIP Training Efficiency and Robustness

Abstract: We show that removing images with text regions improves both the efficiency and robustness of CLIP training. The proposed filtering strategy boosts classification and retrieval accuracy and defends against typographic adversarial attacks. ...

Analysis:

The user query includes the topic about 'RegionCLIP model" and already performed a search for related topics, including 'CLIP visual localization", 'ViLD model", and 'RegionCLIP model". The current paper list includes several relevant works, with the most directly relevant being [2112.09106] (RegionCLIP: Region-based Language-Image Pretraining), which is the primary paper matching the exact query.

From the list: [2112.09106] is the core paper on RegionCLIP and directly addresses the user's initial interest. [2104.13921] (ViLD) is closely related, as it deals with open-vocabulary object detection via vision and language distillation—complementary to RegionCLIP's goal of fine-grained region understanding. [2308.02299] (RegionBLIP) is also relevant as it extends multimodal models to regional comprehension, implying a similar focus on region-level understanding. [2310.18340] (UrbanCLIP) is less directly relevant, as it applies CLIP to urban region profiling rather than object detection or region alignment. [2302.00275] and [2309.16020] focus on geolocation, which is tangentially related via vision-language alignment but not directly on region-level features.

We should expand beyond the initial RegionCLIP paper to explore related works that enhance region-level understanding, especially those that build upon or extend CLIP for regional visual representations. Given that [2112.09106] is the primary paper, we should expand it to surface more foundational or downstream works that might be relevant.

Moreover, [2104.13921] and [2308.02299] are strong secondary candidates because they also involve region-image-text alignment in a multimodal setting. However, they were not previously expanded, so expanding these papers would provide a more robust understanding of the ecosystem.

Therefore, the next actions should be: Expand the primary paper [2112.09106] to uncover related foundational or downstream works. Expand [2104.13921] (ViLD) to explore its references, as it shares a similar vision (open-vocabulary detection via language vision alignment). Expand [2308.02299] (RegionBLIP) to discover related works on regional comprehension in multimodal models.

These expansions will enrich the paper pool without repeating prior searches or expanding redundant papers (e.g., earlier ones not in the current list).

Tool Call:

```
{"name": "expand", "arguments": {"arxiv_id": "2112.09106"}}
{"name": "expand", "arguments": {"arxiv_id": "2104.13921"}}
{"name": "expand", "arguments": {"arxiv_id": "2308.02299"}}
```

Step 3

...

Step 4

Paper List:

[2112.09106] (1.00) [EXP] RegionCLIP: Region-based Language-Image Pretraining

Abstract: CLIP is ineffective for region-level object detection due to the domain shift between whole-image–text alignment and region–text alignment. RegionCLIP extends CLIP to learn region-level visual representations, significantly improving open-vocabulary and zero-shot object detection performance. ...

[2104.13921] (0.99) [EXP] Open-vocabulary Object Detection via Vision and Language Knowledge Distillation

Abstract: We propose ViLD, a vision-language knowledge distillation framework that transfers open-vocabulary knowledge from a pretrained image-text model to a two-stage detector. ViLD achieves strong generalization on multiple detection benchmarks and outperforms prior methods on novel categories without additional finetuning. ...

[1908.02265] (0.54) [NEW] ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks

Abstract: We propose ViLBERT, a task-agnostic vision-and-language pretraining model with a two-stream architecture and co-attentional transformers. Pretrained on large-scale image-text data, ViLBERT achieves state-of-the-art performance across multiple vision-language tasks with minimal task-specific modification. ...

(...More Papers...)

[2301.12597] (0.05) [NEW] BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models

Abstract: We propose BLIP-2, an efficient vision-language pretraining framework that leverages frozen image encoders and frozen large language models. With a lightweight querying transformer, BLIP-2 achieves state-of-the-art performance and strong zero-shot capabilities with significantly fewer trainable parameters. ...

Analysis:

The user cares about the studies that leverage CLIP's capabilities, and we have already identified the primary paper on RegionCLIP (arXiv:2112.09106), which is now marked as [EXP] (expanded). This paper is the central focus and has been used as a seed for expansion via the expand tool (e.g., expanded papers like 2104.13921, 2308.02299, etc.).

Currently, the paper list includes several relevant papers on vision-language pre-training and region-based models, such as: RegionBLIP (2308.02299): focuses on regional comprehension in multi-modal models. UrbanCLIP (2310.18340): uses text-enhanced urban region profiling with CLIP-like training. Oscar (2004.06165): object-semantic alignment pre-training. FILIP (2111.07783): fine-grained language-image pre-training with token-wise alignment. STAIR (2301.13081): sparse text and image representation.

These papers are closely related in theme, focusing on regional or fine-grained visual understanding in vision-language settings, which supports the broader context of RegionCLIP as a foundation. However, many of these have been already expanded (marked [EXP]), so further expansion may not yield new relevant papers.

To deepen the exploration, we should: Identify missing papers from the primary RegionCLIP literature by searching on related keywords to complement what we already found. Expand on other key papers that are relevant but not yet expanded, such as: STAIR (2301.13081): sparse representations, strong performance in retrieval. FILIP (2111.07783): fine-grained alignment, strong localization. BLIP-2 (2301.12597): foundational MLLM, influential in regional vision-language tasks.

Avoid expanding on already expanded papers (e.g., 2112.09106, 2104.13921, 2308.02299, 2310.18340, 2305.05095) to avoid redundancy.

Therefore, the next actionable steps should include: Searching for papers related to 'fine-grained region understanding in vision-language models" to complement existing ones. Expanding on key under-explored candidates like FILIP (2111.07783) and BLIP-2 (2301.12597), as they offer strong theoretical or practical advancements in regional language-image alignment.

This will ensure we cover both complementary and emerging directions in the region-based vision-language pre-training space.

Tool Call:

```
{"name": "search", "arguments": {"query": "fine-grained region understanding in
vision-language models"}}
{"name": "expand", "arguments": {"arxiv_id": "2111.07783"}}
{"name": "expand", "arguments": {"arxiv_id": "2301.12597"}}
```

More Steps

...

Final Result

Paper ID: 2112.09106

- **Title:** RegionCLIP: Region-based Language-Image Pretraining
- **Abstract:** ...
- **Score:** 0.9957

Paper ID: 2104.13921

- **Title:** Open-vocabulary Object Detection via Vision and Language Knowledge Distillatio
- **Abstract:** ...
- **Score:** 0.9949

(...More Papers...)
Paper ID: 2203.16513

- **Title:** PromptDet: Towards Open-vocabulary Detection using Uncurated Images
- **Abstract:** ...
- **Score:** 0.5027