Extended Abstract

Background and Our Project Goal: Large language models (LLMs) have demonstrated impressive reasoning capabilities after augmented by reinforcement learning methods like DeepSeek-R1 Guo and et al. (2025). However, they remain fundamentally closed-world systems, trained on static corpora and prone to hallucinations in LLM responses when tasks require external knowledge such as enterprise internal documents or real time information from Internet. Retrieval-Augmented Generation (RAG)Lewis et al. (2021) mitigates such limitations at inference time by first retrieving relevant passages from documents/Internet and then feeding them to LLM for generating outputs. Our project is interested in enhancing LLM reasoning over external knowledge through explicitly integrating reinforcement learning with search/retrieval interactions during both model training and inference.

Our Method: The recent work Search-R1 Jin et al. (2025b) has the first attempt to integrate reinforcement learning with search engine, and uses a simple outcome-based reward based on the exact matching between predicted and ground truth answers. Our method improves the Search-R1 reinforcement learning framework by leveraging an LLM-as-a-judge to evaluate the relevance of search query during training, a.k.a. **RLAIF** (Reinforcement Learning from AI Feedback).

In our RLAIF, we utilize a separate LLM to compute **search-question relevance score** r_{Rel}, based on the relevance between question and the search query from model thinking process. This score reflects semantic alignment and factual support of the search query to the question.

We design a new reward function by combining both exact matching reward and search-question relevance reward $R = \max\big(\mathrm{EM},\ \alpha \cdot r_{\mathrm{Rel}}\big)$, where $\mathrm{EM} \in \{0, 1\}$ indicates answer Exact Matching and α is a scaling factor that balances these two reward functions.

Setting: We follow the experiment setting in Search-R1 code base Jin et al. (2025a). The Wikipedia-18 corpus is indexed using the E5 retriever (with passages stored in a JSONL file and an ANN index built) and served locally via a retrieval server. GRPO algorithm is used for both reproducing Search-R1 method and our new method. We use the same hyperparameters as in the Search-R1 code base and conduct each experiment on Qwen2.5-3B LLM base model using 2 A100 80GB GPUs. We evaluate our method on two datasets: Natural Questions (NQ) dataset Kwiatkowski et al. (2019) and HotpotQA dataset Yang et al. (2018). We utilize two evaluation metrics: Exact Matching Accuracy (EMA) and Semantic Matching Accuracy (SMA) across varying similarity thresholds.

Results: The results show that our RLAIF method substantially outperforms all prior baselines on both Exact Match QA accuracy and semantic matching metrics. On Natural Questions, it achieves 44.3% EM—3.7% (9% relative) above the strongest Search-R1 baseline—and on HotpotQA it reaches 30.9%, a 2.5% (8.8% relative) improvement. In semantic matching (Figure 4), RLAIF consistently yields about 4% higher accuracy across similarity thresholds on NQ and 1–2% higher on HotpotQA, demonstrating more robust answer quality even as the cosine-similarity requirement tightens. Qualitatively, RLAIF typically requires only a single retrieval step to gather the necessary evidence, leading to more efficient reasoning pipelines. The failures are oftentimes traced back to incorrect or misleading retrieval results that cascade into wrong answers.

During training, RLAIF steadily increases both its EM reward and its relevance-score reward, while Search-R1 oscillates and plateaus lower. In RLAIF training, the policy gradient loss decays to near zero and the KL divergence grows moderately, indicating stable convergence with controlled deviation from the reference policy. Finally, RLAIF's valid-search rate and end-to-end QA completion both climb rapidly where they remain high while Search-R1 collapses after certain training steps.

Discussions: We observe that smaller models such as Qwen2.5-0.5B fails to achieve promising results for both Search-R1 and our RLAIF method. Moreover, extending training duration beyond a certain point produces diminishing returns, underscoring the need for a careful balance between training time and performance gains.

Conclusions and Future Work: Our RLAIF shows that integrating an LLM-based evaluator into RL training improves model accuracy while ensuring training stability. Future work will explore scaling to larger LLM models and other reasoning tasks in addition to question answering.

Enhancing LLM Reasoning on External Knowledge

Claire Tang
Department of Computer Science
Stanford University
cltang@stanford.edu

Abstract

Large language models (LLMs) have demonstrated impressive reasoning capabilities, particularly when enhanced with reinforcement learning techniques like DeepSeek-R1. However, existing methods remain limited by their closed-world nature, lacking mechanisms to incorporate external, real-time knowledge during inference or training, causing hallucinations in LLM outputs. In this paper we are interested in enhancing LLM reasoning capabilities over external knowledge. We propose a novel reinforcement learning framework that works with search engine to integrate retrieved knowledge into the training process and adopts an LLM-As-a-Judge to provide feedback (RLAIF). Our RLAIF introduces a new reward function that also considers the relevance of the retrieved passages from external documents in addition to the final output matching, which was used in recent Search-R1 Guo and et al. (2025) work for its reward function. Our experiments demonstrate the effectiveness and scalability of our RLAIF in improving LLM reasoning capability while maintaining superior training stability. When evaluated on the Natural Questions and HotpotQA QA benchmarks, our method significantly outperforms all baselines, with improvements of up to 3.7% in exact match accuracy over the current best Search-R1 method when both trained on the Qwen2.5-3B model.

1 Introduction

Large language models (LLMs) is one of the most significant breakthroughs in human history, and many put it equivalent to industrial revolution. It started with demonstrating remarkable capabilities in natural language understanding and generation and then expanded into visual generations and multimodal LLMs. Recently, we have also witnessed another great advancement in the reasoning capabilities of LLMs. In particular, models such as OpenAI-o1 Jaech and et al (2024) and DeepSeek-R1 Guo and et al. (2025) have used RL techniques Kaelbling et al. (1996)Sutton et al. (1999) to improve logical reasoning and problem solving skills by interacting with environment and learning from trials and feedback from the environment.

Despite these achievements, LLMs are closed system because they are pre-trained on a huge amount of fixed data. They often encounter challenges when tasked with complex reasoning over external knowledge that did not appear in the pre-training data. For example, when needing to answer questions based on knowledge from enterprise internal documents or up-to-date information from the Internet, LLMs may unfortunately only give hallucinated answers based on pre-training data it has seen before. Therefore, it is highly desirable to extend the LLM reasoning capability to external data source and knowledge.

Making closed-world LLMs open to external knowledge would require integrating reasoning abilities with the ability to interact effectively with search engines that can utilize up-to-date external information. Retrieval-augmented generation (RAG) Lewis et al. (2021) Gao et al. (2024) is one of the most popular approaches for integrating LLMs with search engines, either treating the search engine as a tool Schick et al. (2023) or utilizing a separate retrieval model to first retrieve passages from external

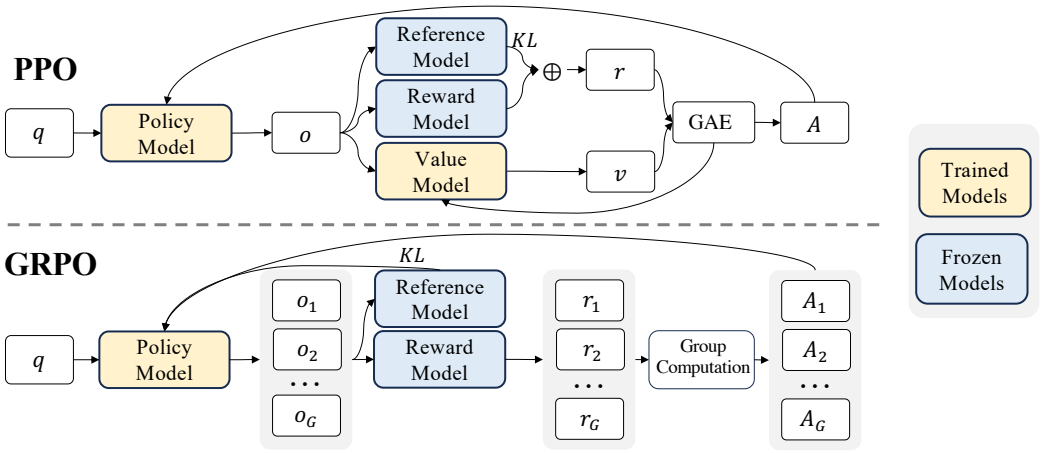


Figure 1: PPO Schulman et al. (2017) and GRPO Shao et al. (2024) Algorithms

documents based on the LLM input as query and then incorporate the retrieved passage as part of the context for LLM generation Lewis et al. (2021). This allows the LLM to leverage external knowledge to correctly answer questions instead of giving hallucinated answers only based on pre-trained data.

Different approaches to integrate search engine as part of the LLM reasoning process have been explored, including simply putting retrieved passages into prompts or actually training LLMs to use search engine as a tool during reasoning Schick et al. (2023) Qu et al. (2025). But these methods are difficult to generalize and scale effectively due to their reliance on task specific prompts, high-quality annotated data, as well as the inherent non-differentiability of the search operation. Recently Search-R1 Jin et al. (2025b) has proposed a novel way to applying RL to the search-and-reasoning scenarios that require LLMs to utilize knowledge from external sources. They demonstrate that even using a straightforward outcome-based reward function is effective in making closed-world models open to external knowledge reasoning.

In this paper we explored ways to improve the reasoning capabilities on external knowledge over Search-R1 method Jin et al. (2025b) by designing a novel RLAIF (RL through AI Feedback) framework that can work with search engine and also novel reward functions for the new framework.

2 Related Work

2.1 Reinforcement Learning Algorithms

Reinforcement Learning Kaelbling et al. (1996) Sutton et al. (1999) is a type of machine leaning mechanism that learns by doing (trial and error) and interacting with the environment, getting feedback from the environment through a format of reward for doing it right. Over the time, it learns a behavioral policy that can achieve high reward for long term success. Different policy update algorithms have been developed based on experience and feedback, most popular one being Policy Gradient method Sutton et al. (1999). Proximal Policy Optimization (PPO)Schulman et al. (2017) and Generalized Proximal Policy Optimization (GRPO) Shao et al. (2024), as shown in Figure 1, are two advanced methods in this category.

Proximal Policy Optimization (PPO) Schulman et al. (2017) is a widely used reinforcement learning algorithm. As a policy gradient method, it uses gradient to guide how to update the policy to increase the expected reward. The risk of changing the policy too much in any single step is that the model may forget what it has learned from previous steps and thus cause the model to be unstable. In order to prevent this from happening, PPO uses a *clipping mechanism*, where it compares the new policy to the old one and only allows conservative updates that don't change policy too drastically, typically keeping the probability ratio between new and old policies in a small range $[1-\epsilon, 1+\epsilon]$. This balance

between performance and convergence stability makes PPO popular in modern RL applications.

$$\mathcal{J}_{\mathrm{PPO}}(\theta) = \mathbb{E}_{\substack{x\sim\mathcal{D} \\ y\sim\pi_{\mathrm{old}}(\cdot|x;\mathcal{R})}} \left[\frac{1}{\sum_{t=1}^{|y|} I(y_t)} \sum_{t=1}^{|y|} \min\left(\frac{\pi_\theta(o_{i,t})}{\pi_{\mathrm{old}}(o_{i,t})} A_t,\ \mathrm{clip}\Big(\frac{\pi_\theta(o_{i,t})}{\pi_{\mathrm{old}}(o_{i,t})}, 1-\epsilon,\ 1+\epsilon \Big) A_t \right) \right].$$

While PPO mechanism needs to train a value function to estimate how good an action is in an absolute sense, Generalized Proximal Policy Optimization (GRPO) Shao et al. (2024) removes the need of such value function. Instead, it learns from relative comparisons of actions taken in the same context. More specifically, for each input, the model generates a group of outputs (e.g., multiple different completions or actions), these outputs are then ranked or scored, either by humans or another AI system. The reward is assigned relatively based on how well each output compares to others in the same group. The best-performing outputs get higher rewards; weaker ones get lower rewards. These relative rewards are then normalized within the group to guide policy update. These normalized rewards can naturally avoid big update of the policy in one step and maintain convergence stability. This group-based learning mechanism makes GRPO a powerful and efficient way to improve decision making success, especially in tasks like language generation or reasoning.

$$\mathcal{J}_{\mathrm{GRPO}}(\theta) = \mathbb{E}_{q\sim P(Q),\, \{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\mathrm{old}}}(O|q)}$$
$$\left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \Big\{ \min\Big(\frac{\pi_\theta(o_{i,t})}{\pi_{\theta_{\mathrm{old}}}(o_{i,t})} \hat{A}_{i,t},\ \mathrm{clip}\Big(\frac{\pi_\theta(o_{i,t})}{\pi_{\theta_{\mathrm{old}}}(o_{i,t})},\, 1-\epsilon,\, 1+\epsilon \Big) \hat{A}_{i,t} \Big) - \beta\, \mathrm{KL}\big[\pi_\theta \,\big\|\, \pi_{\theta_{\mathrm{ref}}}\big] \Big\} \right]$$

2.2 LLM and Reinforcement Learning

Large Language Models (LLM) and Reinforcement Learning (RL) intersect in powerful ways, especially post-training to control model behavior. More specifically, after LLMs are pre-trained through vast amount of corpora, Reinforcement Learning from Human Feedback (RLHF)Ouyang et al. (2022) is often used to align the model's outputs with human preferences or desired objectives. In RLHF, a reward model is trained based on human ratings of model outputs, and this reward model then guides the update of the LLM behavior policy through reinforcement learning algorithms (often via PPO or GRPO). This process helps improve the behavior of LLMs to adhere to the human standard and maintain its helpfulness, safety, and ethics. Reinforcement Learning algorithms can also improve LLMs' logic reasoning and problem solving skills for complex problems. In fact, Deepseek-R1 Guo and et al. (2025) uses the GRPO algorithm to drastically improve the LLM reasoning capabilities.

Search-R1 Jin et al. (2025b) expands the reasoning capability of DeepSeek-R1 Guo and et al. (2025) to the search and reasoning scenarios that require LLMs to retrieve external documents through RAG Lewis et al. (2021)Gao et al. (2024). It models the search engine as part of the environment and makes it compatible with existing RL algorithms, including PPO and GRPO. We can refer to the upper part of Figure 2 without the orange components. Search-R1 adopts a straightforward outcome exact matching based reward function, avoiding the complexity of process-based rewards. They demonstrate that even using such straightforward reward function is effective in search-and-reasoning scenarios to make closed-world models open to external knowledge based reasoning.

Reinforcement Learning with AI Feedback (RLAIF)Lee et al. (2024) is an emerging approach that builds on the principles of Reinforcement Learning from Human Feedback (RLHF), but instead of relying on human-labeled preferences to train the reward model, it uses another AI model to provide feedback. This approach significantly reduces human cost and overcomes the scalability limitations of RLHF by automating the output evaluation process. In RLAIF Lee et al. (2024), an auxiliary AI model, often another LLM, is used to evaluate and rank the outputs generated by the primary LLM, and these rankings are used to train a reward model. The primary LLM is then fine-tuned on its behavior policy via reinforcement learning algorithms such as PPO and GRPO using this reward signal. RLAIF Lee et al. (2024) has recently been shown to be a promising step toward more scalable and automated alignment methods for large language models.

While previous RLAIF Lee et al. (2024) does not work with search engine, in this paper we have designed a novel RLAIF framework on top of the original Search-R1 Jin et al. (2025b) pipeline to work with search engine and also introduced a new reward function for our framework. Our experimental results demonstrate that we achieve better performance than the Search-R1 method.

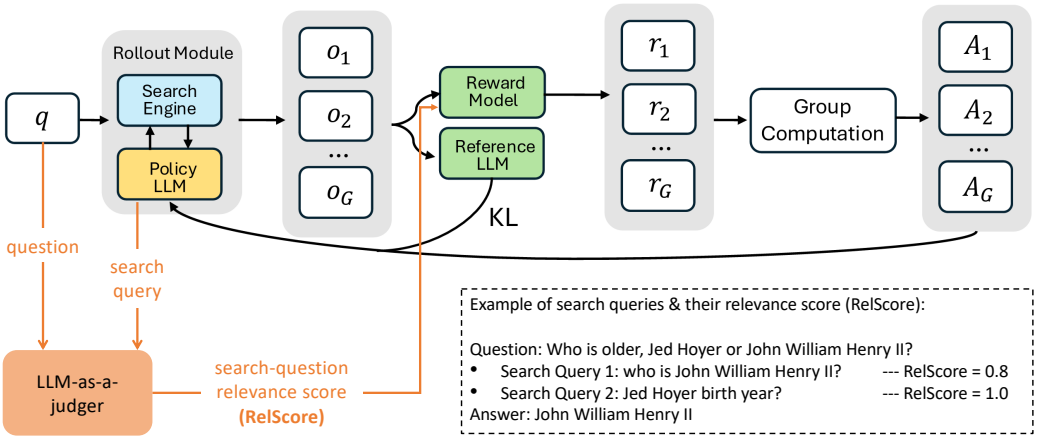


Figure 2: Our RLAIF Method that improves Search-R1 (our new components in orange)

3 Our Method

We designed a new RLAIF (Reinforcement Learning with AI Feedback) method to improve the recent work Search-R1 Jin et al. (2025b), which integrated reinforcement learning with search engine interactions. While Search-R1 teaches the LLM model *when to search* and *search which content*. When the model predicted answer is wrong, it could be two reasons: the wrong search query being used for search or the mistakes from search engines with the correct search query. However, the exact matching reward model in Search-R1 fails to distinguish these two potential reasons, thereby oftentimes leads to suboptimal model.

The key idea of our RLAIF method is to reward both final answer and search query. As shown in Figure 2, we introduce an LLM-as-a-judger module to measure the relevance score between original question and the search query. Then, we leverage this relevance score to adjust the LLM generation process and design a new reward model.

3.1 Search-Question Relevance Score (*RelScore*)

Our new **search-question relevance score** measures the relevance between the original question and the search query from rollout module in model thinking process. That said, when model generates <*search*>*search query*</*search*> that invokes a search engine, we use off-the-shelf LLM model as LLM-as-a-judger to evaluate its relevance to the question, with the following prompt:

You are a relevance evaluator. Given:
Question: {*question*}
Search Query: {*search query*}
Assess how well the search query addresses the question. Output a single decimal number between 0 and 1, where:
1.0 = perfectly relevant
0.0 = not relevant at all
Do NOT output any additional text or explanation. Provide the answer inside <score> and </score>.

RelScore is between 0 and 1. The higher *RelScore* the more relevant between the original question and the search query.

3.2 Generation based on *RelScore*

Figure 3 shows how *RelScore* is used in LLM generation process during training. Compared to Search-R1, we consider only calling search engine when the search query is relevant to the question. That is, *RelScore*({question}, {search Query}) is larger than or equal to some threshold τ, referred to as *search threshold*. Otherwise, we will just ignore the search query from rollout module.

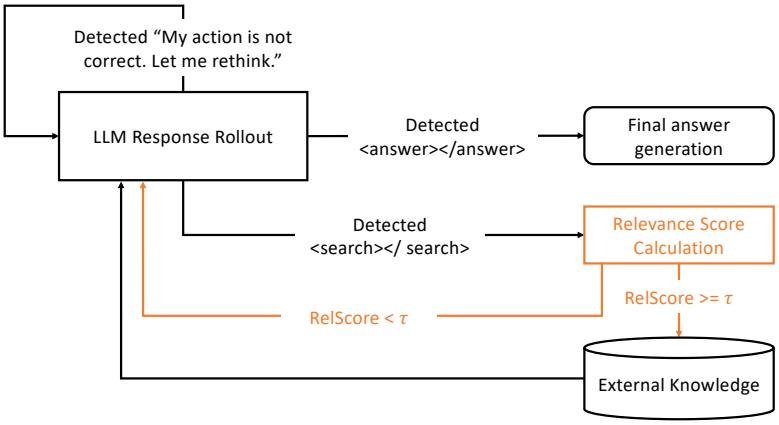


Figure 3: Generation based on *RelScore*

3.3 Reward Modeling

We design a new reward model R based on the search-question relevance and the exact matching. The key idea is to provide more signals about the correctness of the thinking process when the predicted answer does not match the ground truth.

$$R = \max\big(\mathrm{EM},\ \alpha \cdot \mathit{RelScore}\big)$$

where $\mathrm{EM} \in \{0, 1\}$ indicates exact answer match and α is a scaling factor that balances these two reward functions.

The reward model R can be understood more simply as follows:

$$R = \begin{cases} 1, & \text{If the ground truth and predicted answers exactly match;} \\ \alpha * \mathit{RelScore}, & \text{Otherwise} \end{cases}$$

4 Experimental Setup

We follow the experiment setting in Search-R1 code base Jin et al. (2025a). We train and evaluate our RLAIF method on Qwen-2.5-3B Base LLM model. A retriever server is set up locally with E5 as the retriever and wikipedia as the corpus. The number of retrieved passages is set to 3. GRPO reinforcement learning algorithm is used for both reproducing Search-R1 method and our new method. The model training metrics are logged in Weight & Bias. We use the same hyperparameters as in the Search-R1 code base. In addition, we select the scaling factor in reward model $\alpha = 0.1$ and the search threshold $\tau = 0.8$. All experiments are run on 2 A100 80GB GPUs. We select the checkpoints with highest EM-reward during training for evaluation.

For LLM-as-a-judger, we use the off-the-shelf GPT-4o-mini model by calling OpenAI APIs. However, the native OpenAI LLM generation APIs do not support batch processing. We implemented the LLM generation batch processing using Python's asyncio and aiohttp libraries to send multiple OpenAI API requests concurrently. It takes a list of prompts, constructs an asynchronous POST request for each prompt, and gathers all responses in parallel using asyncio.gather. Inside the function, a nested fetch handles the API call and extracts the model's response. This approach significantly improves throughput compared to sequential calls for LLM-as-a-judger and largely accelerates the RLAIF training speed.

We conduct our experiments on two QA datasets: Natural Questions (NQ) dataset Kwiatkowski et al. (2019) and HotpotQA dataset Yang et al. (2018).

- **NQ** is a general question answering benchmark dataset, which consists of real user questions from Google Search and their corresponding Wikipedia pages with manually annotated answers. We train on 79.2k samples and test on its test split with 3.61k samples.

	NQ	HotpotQA
Direct Inference	0.106	0.149
CoT	0.023	0.021
IRCoT	0.111	0.164
Search-o1	0.238	0.221
RAG	0.348	0.255
SFT	0.249	0.186
R1	0.226	0.201
Search-R1	0.406	0.284
Our RLAIF Method	**0.443**	**0.309**

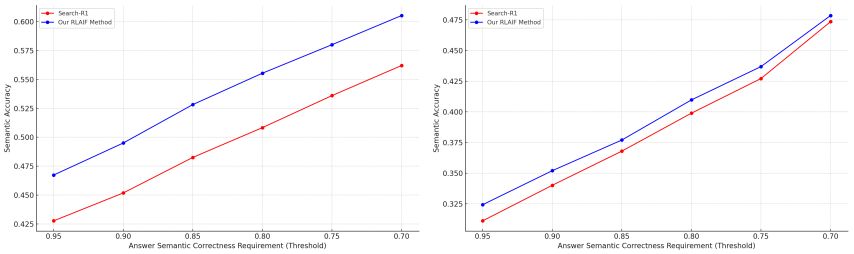
Table 1: EM accuracy of various methods with Qwen2.5-3B LLM on NQ and HotpotQA Datasets

- **HotpotQA** is a multi-hop reasoning question answering benchmark dataset that requires to find and combine information from multiple supporting documents to answer questions. HotpotQA includes diverse question types, such as bridge questions that link information across paragraphs through shared entities, comparison questions that require comparing attributes of different entities, and factoid questions. We train on 90.4k samples and test on its dev split with 7.41k samples.

5 Results

5.1 QA Accuracy Quantitative Results

Table 1 shows the Exact Matching (EM) QA accuracy results of our RLAIF method and other baselines. We use the same baselines as in Search-R1. As we can see, RLAIF delivers the strongest performance of all methods on both benchmarks, achieving 44.3% on NQ dataset and 30.9% on HotpotQA dataset. When comparing to the strongest Search-R1 method, RLAIF surpasses Search-R1(40.6%) by nearly 3.7 points, representing a relative gain of over 9%. On HotpotQA, RLAIF also outperforms Search-R1 (28.4%) by 2.5%, projecting to a relative gain of 8.8%. The results demonstrated the effectiveness of our designed *RelScore*, and its application to both generation and rewarding model modules.



(a) NQ Dataset (b) HotpotQA Dataset

Figure 4: Answer Semantic Matching Accuracy (Blue: Our RLAIF Method, Red: Search R1)

Figure 4 shows *Answer Semantic Matching Accuracy* of our RLAIF and Search-R1 methods. Specifically, for ground truth answer and model predicted answer, we uses a pretrained embedding model (all-MiniLM-L6-v2) to convert each answer into a numeric vector and then computes the cosine similarity score between these vectors. The similarity score is used as *Answer Semantic Correctness Requirement*, and the higher requirement leads to lower answer semantic matching accuracy.

The results in Figure 4 show that both our RLAIF model and Search-R1 exhibit a clear upward trend in semantic accuracy as the correctness threshold is relaxed. Compared to Search-R1, RLAIF consistently outperforms Search-R1 by around 4% on NQ dataset and around 1-2% on HotpotQA dataset. For HotpotQA dataset, due to the complexity of multihop reasoning task, the smaller gain performance gain of our RLAIF method is likely to be because of the small size LLM (Qwen2.5-3B).

5.2 Qualitative Results

Table 2: Success Example in NQ Dataset

Question	Where was the u.s.s maine when it exploded in 1898?
Answer	Havana Harbor
`<think>`	I need to find out where was the u.s.s maine when it exploded in 1898. I'll search for it.
`<search>`	where was the u.s.s maine when it exploded in 1898?
`<information>`	Doc 1(Title: "USS Maine") whose sinking by an explosion on February 15, 1898 precipitated the Spanish–American War. To the south and east is the Atlantic Ocean and to the north and northeast is New Brunswick, a province of Canada... Doc 2(Title: "Charles Dwight Sigsbee") Charles Dwight Sigsbee Charles Dwight Sigsbee (January 16, 1845 – July 13, 1923) was a Rear Admiral in the United States Navy... Doc 3(Title: "USS Maine Mast Memorial") USS Maine Mast Memorial The USS ""Maine"" Mast Memorial is a memorial honoring those who died aboard the on February 15, 1898, after a mysterious explosion destroyed the ship while at anchor in Havana Harbor. It is located ...
`<answer>`	Havana Harbor

Table 3: Failed Example in NQ Dataset

Question	What is the poorest county in the state of florida?
Answer	Union County
`<think>`	I need to find out what is the poorest county in the state of florida. I'll search for it.
`<search>`	what is the poorest county in the state of florida?
`<information>`	Doc 1(Title: Florida) capita income varies widely by geographic region and profession... Doc 2(Title: Florida) "non-tipped" positions, which was higher than the federal rate of $7.25. Florida has 2 cities in the top 25 cities in the U.S. with the highest average credit card debt, Miami and Tampa... Doc 3(Title: "Okeechobee County, Florida") a family was $35,163. Males had a median income of $25,574 versus $20,160 for females. The per capita income for the county was $14,553. About 11.80% of families and 16.00% of the population were below the poverty line, including 19.90% of those under age 18 and 10.30% of those age 65 or over. Okeechobee County is ...
`<answer>`	Okeechobee County

Table 2 and Table 3 show the success and failed examples in NQ dataset. We observe that the model usually only searches once and utilize the retrieved information to derive the answer to the question. The failed example in Table 3 is due to the incorrect information obtained from the retriever.

Table 4: Success Example in HotpotQA Dataset

Question	Who is older, Jed Hoyer or John William Henry II?
Answer	John William Henry II
`<think>`	I need to find out who is older, Jed Hoyer or John William Henry II. I'll search for it.
`<search>`	Who is older, Jed Hoyer or John William Henry II?
`<information>`	Doc 1(Title: "John W. Henry") the firm's management had fallen from $2.5 billion in 2006 to less than $100 million as of late 2012. As of July 2017, ""Forbes"" estimated his net worth to be $2.6 billion. John William Henry II was born on September 13, 1949 in Quincy, Illinois... Doc 2(Title: "Jed Hoyer") career saves record with Sam Elias and Nick Miceli. During his time at Wesleyan University he was also initiated into and is a brother of Delta Kappa Epsilon. He spent Summer 1995 ...
`<answer>`	John William Henry II

Table 4 and Table 5 show the success and failed examples in HotpotQA dataset. In the success example, although the question requires reasoning, the retrieved top 3 documents covered the both persons' information and the model is able to predict the correct answer based on these documents. In the failed example, the model is misled by the incorrect retrieved information, then searches more irrelevant information and finally provides a wrong answer.

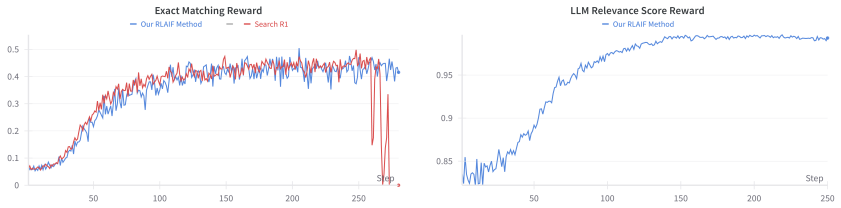
5.3 Result Analysis

We further analyze the trends of different metrics in Search-R1 and our RLAIF method during model training phase.

Figure 5 shows that our RLAIF method consistently drives both exact-match and relevance-score rewards upward throughout training. On the left (EM Reward), the blue RLAIF curve begins near 0.1 and climbs steeply in the first 50 steps, then more gradually up to around 0.45–0.50 by step 150, where it stabilizes with only minor fluctuations even as the Search-R1 baseline (red) oscillates sharply toward the end. On the right (Relevance Score Reward), our method starts around 0.84 and

Table 5: Failed Example in HotpotQA Dataset

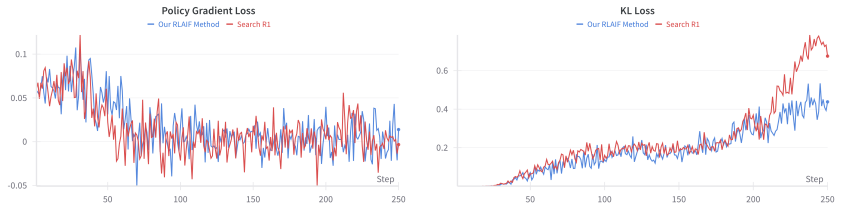
Question	Where was the first governor after the The Missouri Compromise from?
Answer	Bath, Maine
`<think>`	I need to find out where was the first governor after the The Missouri Compromise from. I'll search for it.
`<search>`	Where was the first governor after the The Missouri Compromise from?
`<information>`	Doc 1(Title: "John W. Henry") the firm's management had fallen from $2.5 billion in 2006 to less than $100 million as of late 2012. As of July 2017, ""Forbes"" estimated his net worth to be $2.6 billion. John William Henry II was born on September 13, 1949 in Quincy, Illinois...
`<think>`	I need to find out Kentucky. I'll search for it.
`<search>`	Search Kentucky
`<information>`	Doc 1(Title: "Kentucky") ...
`<answer>`	Kentucky



(a) EM Reward (b) Relevance Score Reward

Figure 5: Reward Trend during Training (Blue: Our RLAIF Method, Red: Search R1)

steadily improves, surpassing 0.90 by step 60 and plateauing above 0.97 by step 150, demonstrating that RLAIF not only learns to produce more exact answers but also increasingly aligns the relevance between search query and original question over the course of training.

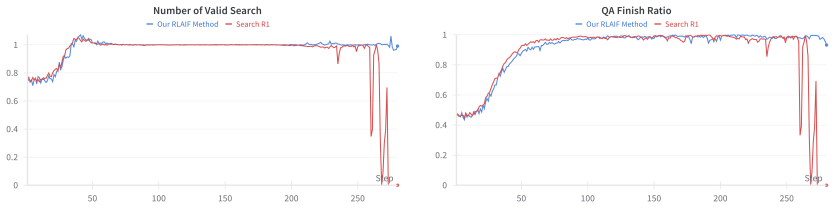


(a) Policy Gradient Loss (b) KL Divergence Loss

Figure 6: Loss Trend during Training (Blue: Our RLAIF Method, Red: Search R1)

As we can see in Figure 6, during the training process, our RLAIF method's policy gradient loss (left) steadily decays from around 0.08 in the early steps down toward zero by step 100 and then fluctuates tightly around zero for the remainder of training, indicating that the policy updates become small and stable as the model converges. Meanwhile, the KL divergence loss (right) for RLAIF starts near zero and grows gradually, reaching roughly 0.25 by step 150 and plateauing around 0.35–0.40 by step 250, suggesting a moderate divergence from the reference policy that increases as the model refines its behavior without blowing up. Both trends show that RLAIF achieves stable policy updates while smoothly trading off against the KL penalty throughout training.

In terms of the success rate, we measure the number of valid search and QA finish ratio, in Figure 7. Our RLAIF method (blue) rapidly increases the proportion of valid searches from about 0.50 at step 0 to over 0.90 by step 40, then continues climbing to nearly 1.0 by step 80, where it remains consistently high while the Search-R1 (red) collapses near the end. Similarly, the QA finish ratio under RLAIF mirrors this pattern. These trends demonstrate that our RLAIF method quickly and robustly optimizes both search validity and end-to-end QA completion.



(a) Number of Valid Search (b) QA Finish Ratio

Figure 7: Success Rate during Training (Blue: Our RLAIF Method, Red: Search R1)

6 Discussion

We find that extremely small size LLM like Qwen2.5-0.5B consistently fall short of delivering strong performance for both Search-R1 and our proposed RLAIF method. Despite their efficiency and lower computational footprint, these smaller models struggle to perform high-quality retrieval and reasoning, resulting in noticeably weaker accuracy and robustness.

Moreover, simply increasing the number of training epochs does not yield proportional improvements. Beyond a certain threshold, further optimization leads to only marginal gains while incurring substantial additional compute cost, or even worse the collapse of model training. This plateau effect highlights that increasing model training steps alone is sufficient; instead, achieving optimal results requires carefully designing new reinforcement learning training algorithm.

7 Conclusion

Our RLAIF method demonstrates that incorporating an LLM-as-a-judger evaluator into the reinforcement learning loop not only enhances the model's overall accuracy but also contributes significantly to maintaining training stability. By leveraging the evaluator's feedback during policy updates, we are able to guide the model training toward more reliable reasoning and retrieval behaviors without introducing instability or mode collapse. Looking ahead, we plan to investigate how this approach scales when paired with larger language models. Moreover, we plan to extend our method beyond question answering to a broader range of complex reasoning tasks.

8 Team Contributions

This is an individual project. All aspects of the project - including literature review, algorithm development, implementation, and evaluation - were conducted by Claire Tang.

Changes from Proposal There is no change of research objective. After the careful study of the existing work after proposal, we specified our fine-grained project scope from two aspects:

- In practice, there are two types of external knowledge: (1) general search engine, (2) retrieval from a predefined set of documents. In this project, we focus on the second type of external knowledge, following the existing Search-R1 paper Jin et al. (2025b).
- In terms of choosing LLM models, we use the same Qwen2.5-3B-Base model as Search-R1.

References

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv:2312.10997 [cs.CL] `https://arxiv.org/abs/2312.10997`

Daya Guo and et al. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *CoRR* abs/2501.12948 (2025). `https://doi.org/10.48550/ARXIV.2501.12948` arXiv:2501.12948

Aaron Jaech and et al. 2024. OpenAI o1 System Card. arXiv:2412.16720 [cs.AI] https://arxiv.org/abs/2412.16720

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025a. Search-R1. https://github.com/PeterGriffinJin/Search-R1.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025b. Search-R1: Training LLMs to Reason and Leverage Search Engines with Reinforcement Learning. arXiv:2503.09516 [cs.CL] https://arxiv.org/abs/2503.09516

L. P. Kaelbling, M. L. Littman, and A. W. Moore. 1996. Reinforcement Learning: A Survey. arXiv:cs/9605103 [cs.AI] https://arxiv.org/abs/cs/9605103

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* 7 (2019), 452–466. https://doi.org/10.1162/tacl_a_00276

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2024. RLAIF vs. RLHF: Scaling Reinforcement Learning from Human Feedback with AI Feedback. arXiv:2309.00267 [cs.CL] https://arxiv.org/abs/2309.00267

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv:2005.11401 [cs.CL] https://arxiv.org/abs/2005.11401

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. arXiv:2203.02155 [cs.CL]

Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-rong Wen. 2025. Tool learning with large language models: a survey. *Frontiers of Computer Science* 19, 8 (Jan. 2025). https://doi.org/10.1007/s11704-024-40678-2

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. arXiv:2302.04761 [cs.CL] https://arxiv.org/abs/2302.04761

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347 (2017). arXiv:1707.06347 http://arxiv.org/abs/1707.06347

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv:2402.03300 [cs.CL] https://arxiv.org/abs/2402.03300

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems*, S. Solla, T. Leen, and K. Müller (Eds.), Vol. 12. MIT Press. https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 2369–2380. https://doi.org/10.18653/v1/D18-1259