A STUDY OF DOWNSTREAM REWARDS FOR RETRIEVAL-ONLY ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Reward design is a central challenge in learning multi-step retrieval policies from downstream feedback. We investigate how different reward signals enable retrieval-only adaptation, where a compact embedding model learns to select evidence for a fixed answer generator. Using Q-RAG as a common framework, we compare rewards based on generated answers, reference likelihoods, and semantic uncertainty, with supporting-fact supervision as a reference. We also introduce Group Information Gain, a contrastive likelihood reward over answer candidates evaluated offline. Experiments on HotpotQA, 2WikiMultiHopQA and MuSiQue yield three observations. Retrieval quality and answer quality are not the same objective: averaged over the two distractor benchmarks, gold-evidence supervision retrieves 5.9 points more evidence F1 than judge feedback and answers no better. Rewards derived from the model's own uncertainty do not improve on exact match anywhere. And the most expensive signal is not always required, since the rewards differ severalfold in online cost while their answer quality spans a few points. Using the selected reward, we then train the retrieval policy over a Wikipedia corpus containing approximately 21 million passages and evaluate it on seven QA datasets. Results show that retriever-only fine-tuning improves downstream answer quality and achieves competitive performance with fine-tuned LLM-based search policies while updating only a compact embedding model. Adapting the retriever already required by the RAG pipeline thus offers a flexible way to specialize search while leaving the LLM unchanged for other tasks. The code is available at `https://anonymous.4open.science/r/Q-RAG-feedbacks-A524/`.

1 INTRODUCTION

Learning to retrieve useful evidence is a central problem in retrieval-augmented generation (Lewis et al., 2020). For multi-step question answering (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022), a retrieval policy must assemble a context that enables a language model to answer, rather than merely identify passages similar to the question (Zhang et al., 2024; Li et al., 2025b). Retrieval-only adaptation addresses this problem by training a compact retrieval model while keeping the answer-generating language model frozen (Shi et al., 2024; Yang et al., 2023). A central question is which downstream reward should guide this adaptation.

Several established signals can provide such feedback. Exact match and token-level F1 score generated answers through normalized string comparison or token overlap (Rajpurkar et al., 2016), while LLM-as-a-Judge evaluates correctness using a language model (Zheng et al., 2023; Zhang et al., 2025a). Likelihood-based rewards measure how retrieved context changes support for a reference answer (Chen et al., 2024; Al-Ghossein et al., 2024), and semantic-uncertainty rewards assess confidence or information gain (Farquhar et al., 2024; Kossen et al., 2024; Hu et al., 2026). Using predictive uncertainty as a training signal is a recurring idea beyond retrieval: in sequential decision making it has been used to decide what a long-term memory should retain, by predicting outcomes the model is uncertain about rather than everything it observes (Sorokin et al., 2022). These alternatives differ in their supervision requirements, sensitivity to answer formulation, access to model internals, and computational cost. Depending on the reward, feedback may require a generated answer, an additional judgment, likelihood evaluation, or multiple samples.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

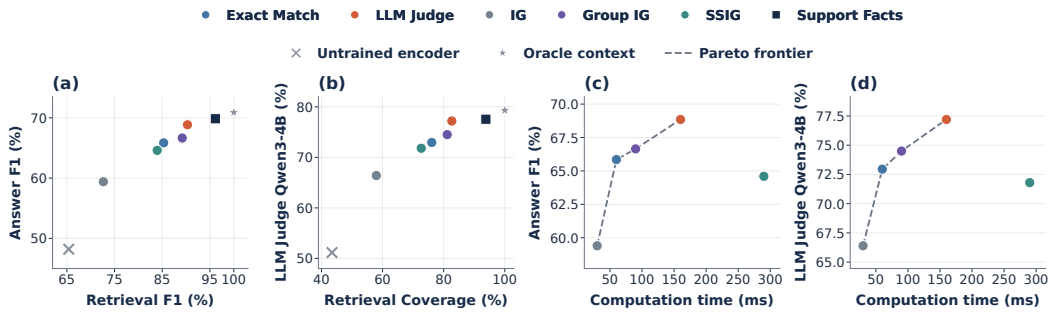


Figure 1: Downstream rewards for retrieval-only training, equal-weight mean over HotpotQA and 2WikiMultiHopQA. (a)–(b) Answer quality versus retrieval quality: LLM-as-a-Judge feedback comes within $1.0\,p.p.$ of answer F1 obtained with gold evidence supervision (Support Facts), despite $5.85\,p.p.$ lower retrieval F1. (c)–(d) Answer quality versus per-episode reward computation time; the dashed line marks the Pareto front, where Group IG sits between Exact Match and the judge.

Successful use of a reward in one system does not establish its effectiveness in another. Existing approaches often investigate a particular signal together with a retrieval or reasoning method (Nguyen et al., 2024; Zhang & Gao, 2024; Cao et al., 2024), leaving its relative merits difficult to separate from the surrounding pipeline. We instead make reward choice the main object of study, comparing representative signals within a common retrieval framework.

We use Q-RAG (Sorokin et al., 2026), an embedding-based, value-learning approach to sequential evidence selection. We extend its original supporting-fact-based training to downstream feedback, retaining supporting-fact supervision as a reference. Alongside established rewards, we introduce Group Information Gain (Group IG), which contrasts context-induced likelihood changes for judged-correct and judged-incorrect answer candidates. Candidates are generated and judged offline, while training evaluates their likelihoods under the retrieved context.

Experiments on HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020) and MuSiQue (Trivedi et al., 2022) compare evidence retrieval, answer quality, and the cost of computing each reward. LLM-as-a-Judge yields the strongest answer quality among the evaluated downstream rewards, but the comparison is more informative in what it reveals about the objective itself. Supervising retrieval directly with gold evidence wins on evidence metrics by a wide margin and yields no better answers, so the reward that best recovers annotated supporting facts is not the reward that best serves the LLM Answerer. Rewards computed from the frozen model's own uncertainty, which have the practical advantage of needing no reference answer, fall below the simplest string-matching signal on every dataset. The rewards also differ severalfold in what a single call costs while their answer quality spans a few points, so the expensive judge is a defensible default rather than a necessary one.

Group IG occupies a different point on that trade-off. Contrasting judged-correct against judged-incorrect candidates repairs the failure mode of single-reference likelihood feedback, which credits any context that raises the likelihood of the reference even when it raises the likelihood of wrong answers as well; the contrast improves answer F1 over single-reference feedback on all three datasets, by 12.4 points on 2WikiMultiHopQA and by 2.1 and 1.6 on the others. Because its candidates are sampled and judged once per dataset, training itself generates nothing.

We then train Q-RAG with judge feedback in a Full-Wiki environment containing approximately 21 million passages. Evaluation on seven QA datasets shows improved average answer quality over the untrained retriever and competitive performance with fine-tuned LLM search policies such as Search-R1 (Jin et al., 2025b) and related reinforcement-learned search agents (Song et al., 2025; Chen et al., 2025), under a shared frozen LLM.

Our contributions are:

- A comparative study of downstream rewards for retrieval-only adaptation under a fixed retrieval framework, LLM Answerer and judge, which isolates the reward from the pipeline around it and shows that evidence retrieval and answer quality rank the rewards differently.
- Evidence that uncertainty-derived rewards, despite needing no reference answer, do not reach the quality of the simplest string-matching signal, and a cost accounting that places each reward on the cost-quality plane.
- An extension of Q-RAG beyond supporting-fact rewards, including Group IG, a contrastive likelihood reward that needs no generation during training and consistently improves on single-reference likelihood feedback.
- A Full-Wiki evaluation demonstrating effective search adaptation and transfer to unseen QA datasets while updating only a compact embedding model rather than an LLM search policy.

2 Related Works

Recent studies have focused on enhancing RAG and ICL pipelines without directly modifying the underlying LLM. Several approaches evaluate the retriever during fine-tuning using LLM-based reward signals, ranging from query-document relevance scoring to comprehensive LLM-as-a-Judge assessments Nguyen et al. (2024); Zhang et al. (2025a). FRRR Zhang & Gao (2024) optimizes the retrieval model by leveraging fine-grained feedback from a black-box LLM. Cao et al. (2024) presents the reward model that constructs rewards by assessing input (retrieved) information on the token-level. This allows to learn which chunks of information bring the most promising impact. Chen et al. (2024) defines rewards via changes in the LLM's likelihood of the reference. The training aims to optimize the iterative selection process to maximize the performance of the downstream LLM generation.

A complementary line of work explores richer intrinsic feedback signals. Yuan et al. (2024) propose a method where LLM iteratively generates new prompts, creates multiple candidate responses to them, and then it evaluates the quality of these candidate responses by several criteria. The highest and lowest scoring responses are then used to train the reward model with a Direct Preference Optimization (DPO).Aksitov et al. (2023) further demonstrates iterative self-improvement through LLM-based evaluation and trajectory selection. This scope of work shows solid improvement in RAG pipelines performance, however these pipelines require frequent expensive LLM calls.

In order to understand how reward design influences system performance, several recent studies have compared reward signals in various LLM optimization settings. Zaradoukas et al. (2026) compared binary rewards with continuous exponential and PageRank-based rewards, demonstrating that continuous rewards outperform simple binary threshold signals. Zhang et al. (2025b); Palandye et al. (2026) performed the overall comparison of outcome-level (final answer correctness) and process-level (per-step) rewards for agentic RAG and mathematical reasoning, respectively. The former fine-tuned LLMs to execute multi-step search trajectories, while the latter compared process-only, outcome-only, and hybrid rewards for mathematical reasoning. In the domain of interleaved reasoning, Jin et al. (2025a) analyzed exact match (EM) answer correctness reward baseline, supplemented with format validity or retrieval evidence score. Furthermore, Zhao et al. (2026) studied EM-based outcome-level reward and 3 types of process-level rewards which are then used for optimizing agentic LLM policies. In contrast, our work focuses exclusively on retrieval-only adaptation. We evaluate a spectrum of downstream reward formulations, including rewards based on generated answers, reference likelihoods, semantic uncertainty, and our novel Group Information Gain (Group IG), within a unified retrieval framework. By characterizing how these signals affect evidence selection, answer quality, and computational cost, our study provides a comprehensive analysis of reward design tailored to the challenges of optimizing compact retrievers while leaving the LLM unchanged.

3 Methods

3.1 Preliminaries

Multi-step retrieval. Let $\mathcal{D} = \{(x_i, y_i, \mathcal{C}_i)\}_{i=1}^{M}$ be a dataset of input queries x with reference answers y, and $\mathcal{C}$ a collection of text chunks. Multi-step retrieval task is formulated as a finite-

horizon Markov decision process. At step t, the state $s_t = (x, c_t)$ contains the query x and the previously selected chunks c_t, with $c_0 = \varnothing$. An action $a_t \in \mathcal{A}(s_t) \subseteq \mathcal{C} \setminus c_t$ selects a chunk and adds it to the context. Candidates are drawn from a document corpus where each chunk is embedded by a chunk embedder. Action set $\mathcal{A}(s_t)$ depends on the state, so early selections change what remains reachable. After T retrieval steps, the selected context c_T (where T is a retrieval budget) is passed to a frozen LLM Answerer $p_\theta(\cdot \mid x, c_T)$, which produces an answer $\hat{y}$. In this work, we consider only downstream rewards assigned at the end of the retrieval episode. All intermediate rewards are zero. The retrieval policy is trained to maximize the expected terminal reward

$$J(\phi) = \mathbb{E}_{(x,y,\mathcal{C})\sim\mathcal{D}}\, \mathbb{E}_{c_T\sim\pi_\phi(\cdot|x,\mathcal{C})}\big[R(x, c_T, y)\big], \tag{1}$$

over the policy parameters ϕ alone. Only the selection of evidence is learned, since the Answerer LLM stays frozen.

Q-RAG. For a policy π, the action-value function $Q^\pi(s, a)$ is the expected discounted sum of rewards after taking action a in state s and subsequently following π, with discount factor $\gamma \in [0, 1]$. Q-RAG (Sorokin et al., 2026) uses soft Q-learning, which additionally includes policy entropy in the return, and parameterizes the resulting action-value function as $Q_\phi(s, a) = \langle E_s(s; \phi_s),\, E_a(a; \phi_a)\rangle$, where E_s and E_a encode states and candidate chunks into $\mathbb{R}^d$. This means that an embedder model E_a is used to embed all chunks/documents in a document corpus. An embedder model E_s generates search queries used to retriev the most promising candidates from the document corpus. During training, actions are sampled from $\pi_\phi(a \mid s) \propto \exp(Q_\phi(s, a)/\alpha)$, where $\alpha > 0$ is the temperature; inference uses greedy selection. Training follows Parallelized Q-Network (PQN; Gallici et al., 2025), which performs synchronous Q-learning updates using transitions collected across parallel environments without a replay buffer. The retrieval policy is trained by minimizing $\mathbb{E}[(Q_\phi(s_t, a_t) - G_t^\lambda)^2]$, where G_t^λ is a weighted mixture of multi-step temporal-difference targets.

Reward notation. We write $c = c_T$ for the final context obtained by following π_ϕ. Prompt instructions for the Answerer LLM are invisible for the policy $p_\theta(\cdot \mid x, c)$. A frozen judge $v_\tau(x, g, y) \in \{0, 1\}$ assesses whether a candidate answer g is correct relative to y. The context-free log-likelihood of g is denoted by $\beta_x(g) = \log p_\theta(g \mid x, c_0)$, using the initial context c_0. The LLM Answerer, the judge and any probes are part of the environment: the only learnable parameters are $\phi = (\phi_s, \phi_a)$.

3.2 Downstream Rewards

Our study focuses on the effectiveness of different downstream rewards for training a multi-step retriever: Exact Match (EM), LLM-as-a-Judge, Information Gain (IG), Group IG, Semantic Entropy Probes (SEP), Synthetic Semantic Information Gain (SSIG). We organize them into three categories: answer-based rewards, which evaluate generated answers; likelihood-based rewards, which measure how retrieved context affects the probability of reference or candidate answers; and uncertainty-based rewards, which assess the model's uncertainty when answering. We also include supporting-fact supervision as a reference. Every reward scores the final context c once per episode and none propagates gradients through p_θ. We write out only Group IG below; the remaining rewards are standard, and Appendix A gives all of them in full.

Answer-based rewards. These draw an answer $\hat{y} \sim p_\theta(\cdot \mid x, c)$ from the frozen LLM Answerer and compare it with the reference. Exact Match (Appendix A.2) is an indicator on normalized strings, which makes it brittle: for $y =$"Paris" and $\hat{y} =$"The answer is Paris" normalization leaves *answer is paris* against *paris*, so the reward is 0 although the answer is correct. Token-level F1 (Appendix A.3) softens this but still pays for surface overlap, and cosine similarity between answer embeddings (Appendix A.4) replaces string matching with a similarity that is not calibrated to correctness. LLM-as-a-Judge reward $r_{\text{judge}} = v_\tau(x, \hat{y}, y)$ (Appendix A.5) drops the string comparison altogether, at the price of one LLM call per episode on top of the generation.

Likelihood-based rewards. These score fixed strings under the LLM Answerer and so require no sampling during training. Writing $\beta_x(g) = \log p_\theta(g \mid x, c_0)$ for the context-free baseline, the single-reference variant is the likelihood shift of the reference alone, $r_{\text{IG}}(x, c) = \log p_\theta(y \mid x, c) - \beta_x(y)$ (Appendix A.6). It is dense and cheap, but it credits any context that makes y more predictable,

including contexts that make competing answers more predictable by as much, because no string other than y is ever evaluated.

Group IG removes this confound by scoring a set of candidates and contrasting the two sides of the judge's verdict. For each query we sample K answers under the context-free conditioning, add the reference, and let the judge partition the result:

$$\mathcal{G}_x = \{g_1, \ldots, g_K\} \cup \{y\}, \quad g_k \sim p_\theta(\cdot \mid x, c_0), \qquad \mathcal{G}_x^+ = \{g \in \mathcal{G}_x : v_\tau(x, g, y) = 1\}, \tag{2}$$

with $\mathcal{G}_x^- = \mathcal{G}_x \setminus \mathcal{G}_x^+$ the judged-incorrect remainder. The reward is the difference between the mean likelihood shift of the judged-correct and the judged-incorrect candidates:

$$r_{\mathrm{GroupIG}}(x, c) = \frac{1}{|\mathcal{G}_x^+|} \sum_{g \in \mathcal{G}_x^+} \Big(\log p_\theta(g \mid x, c) - \beta_x(g) \Big) - \frac{1}{|\mathcal{G}_x^-|} \sum_{g \in \mathcal{G}_x^-} \Big(\log p_\theta(g \mid x, c) - \beta_x(g) \Big), \tag{3}$$

where an average over an empty subset is defined as zero. The reward is positive when the context moves probability mass towards answers the judge accepts and away from those it rejects, so a context that lifts correct and incorrect answers alike scores near zero rather than being rewarded. Sampling, judging and the baselines β_x are computed once per dataset before training, so an episode costs $|\mathcal{G}_x|$ forward passes and no generation.

Uncertainty-based rewards. These need no reference answer. Both draw M answers under a given conditioning and partition them into semantic classes by bidirectional entailment, but they weight the classes differently. Semantic Entropy Probes (Appendix A.8) use cluster frequencies and, since M generations per reward call are expensive, replace the entropy by a logistic probe on the hidden state of a single generation. Synthetic Semantic Information Gain (Appendix A.9) weights classes by their model-likelihood mass instead, and rewards the change in the resulting uncertainty between the context-free prior and the retrieved posterior, either over all classes or over the class containing the reference, combined with exact match through a weight η.

Support facts. When an annotated evidence set $F^\star \subseteq \mathcal{C}$ is available, retrieval is supervised directly and the LLM Answerer is never queried: the reward is $\mathbb{I}[F^\star \subseteq c]$, so partial coverage receives nothing and extra chunks are not penalized (Appendix A.1). This is the only reward here that needs evidence labels, so it serves as a reference and is not comparable to the LLM-feedback signals.

4 Experiments

Our experiments consist of three parts. First, we probe the contrastiveness of downstream rewards, using annotated supporting facts to assess how well each reward distinguishes chunks with useful evidence from irrelevant distractor chunks. Second, we train Q-RAG with the rewards described in 3.2 on HotpotQA and 2WikiMultiHopQA and compare retrieval and answer quality against two important references: oracle retrieval using gold evidence and training with supporting-fact rewards, as in the original Q-RAG. Third, we apply the best-performing downstream reward to more practical settings, Full-Wiki retrieval and multi-turn dialogue QA, and compare the resulting performance with Q-RAG trained using supporting-fact rewards.

4.1 Assessing Reward Contrastiveness

One question in our study is whether annotated supporting facts can help identify effective rewards without training a retriever with each reward. We examine evidence contrastiveness as a potential indicator: a useful retrieval reward should favor contexts containing the evidence needed to answer a question over contexts containing irrelevant information. We therefore test whether the evaluated rewards distinguish these contexts reliably.

Using the supporting-fact annotations in HotpotQA, we construct three types of contexts for each question. *Full-evidence* contexts contain only chunks corresponding to annotated supporting facts. *Partial-evidence* contexts combine supporting chunks with randomly selected non-supporting chunks. *No-evidence* contexts contain only randomly selected chunks that are not annotated as supporting facts. For each reward, we denote its values under these conditions by $R(q, c_{\mathrm{full}})$, $R(q, c_{\mathrm{partial}})$, and $R(q, c_{\mathrm{noise}})$, respectively.

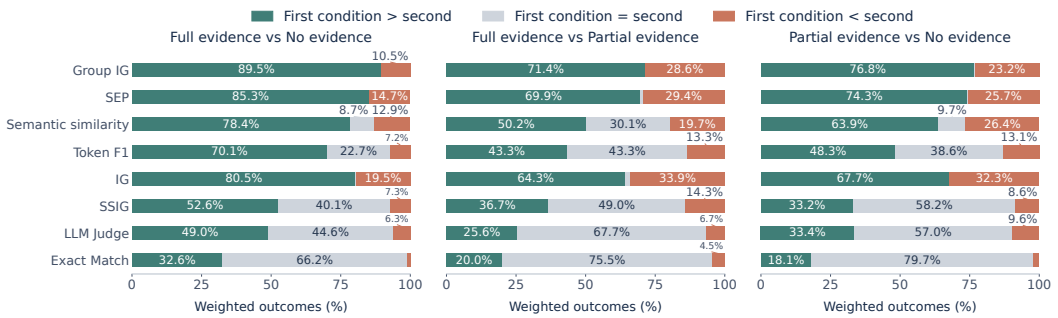


Figure 2: Comparison of reward values across three context types: full evidence, partial evidence, and no evidence. Each bar separates higher, equal, and lower reward outcomes. Results computed on 1000 question samples from HotPotQA dataset Yang et al. (2018)

As shown in Figure 2 Group IG most often assigns the higher reward to the condition with more annotated evidence: 89.5%, 71.4%, and 76.8% across the three comparisons. Binary answer rewards are less contrastive: full and partial evidence tie in 75.5% of comparisons for Exact Match and 67.7% for LLM Judge. Such ties can occur when one paragraph already suffices for a correct answer. Conversely, even Group IG reverses full versus no evidence in 10.5% of questions.

4.2 Training with Downstream Rewards

In this experiments we train Q-RAG with the downstream rewards described in Section 3.2 on three multi-hop QA datasets: HotpotQA, 2WikiMultiHopQA. Training updates only the retriever, while the Answerer LLM is frozen. These datasets provide annotated supporting facts, allowing us to compare downstream rewards against supporting-fact reward and assess their effectiveness. HotpotQA and 2WikiMultiHopQA use different trainable embedding backbones. This allows us to examine whether the relative effectiveness of rewards is consistent across different dataset and backbone combinations. Training and evaluation details are provided in Appendices B and C and in the accompanying code.

The results are shown in Table 2. LLM-as-a-Judge reward gives the highest answer scores among downstream rewards on all datasets and answer metrics (EM, F1, LLM Judge). Group IG improves over single-reference IG everywhere, most visibly on 2Wiki (Retrieval F1 91.6 versus 63.1), but stays below direct judging on answer quality. The uncertainty-based signals are weakest: SEP and SSIG do not improve Answer F1 over Exact Match on both datasets. Support Fact Reward exceeds LLM Judge by 5.85 percentage points in Retrieval F1, but only 1.00 point in Answer F1. Thus, substantially better recovery of annotated evidence corresponds to a much smaller gain in answer quality.

One of the more interesting results shows that the controlled probing experiment does not recover the ranking obtained after training. Rewards that more reliably distinguish full evidence from no evidence do not consistently achieve higher Answer F1, motivating a direct comparison of probing and training outcomes shown in Figure 3a,b.

We quantify reward ability to separate supporting evidence from noise with A_{contr} metric, the fraction of comparisons in which full evidence (two supporting paragraphs) receives a higher reward than no evidence (zero supporting paragraphs), with ties counted as one half:

$$A_{contr} = \frac{1}{N}\sum_{q=1}^{N}\left[\mathbf{1}[R(q, c_{\text{full}}) > R(q, c_{\text{noise}})] + \frac{1}{2}\mathbf{1}[R(q, c_{\text{full}}) = R(q, c_{\text{noise}})]\right]. \tag{4}$$

Higher A_{contr} indicates a more consistent preference for full evidence. Across the five rewards, A_{contr} and mean Answer F1 have Spearman's $\rho = -0.20$ (a measure of rank agreement, with $+1$ indicating identical rankings and -1 reversed rankings; Spearman, 1904). We compute it as the Pearson correlation between the five rewards' ranks in HotpotQA A_{contr} and in Answer F1 averaged equally over HotpotQA and 2Wiki, using average ranks for ties. Group IG leads the probe ($A_{contr} = 0.895$), but LLM Judge gives the highest mean Answer F1 (68.85 versus 66.65). IG

is second in probe separation and last after training. Thus, selecting the most contrastive reward would not select the best training outcome in this comparison. In contrast, the observed training rankings agree closely between datasets: $\rho = 0.975$ for Answer F1 and $\rho = 1.000$ for Retrieval F1 (Figure 3c,d). Here, we correlate the five rewards' within-dataset ranks for each metric. We also observe a strong positive correlation between retrieval and answer metrics across rewards (see Figure 1a,b).

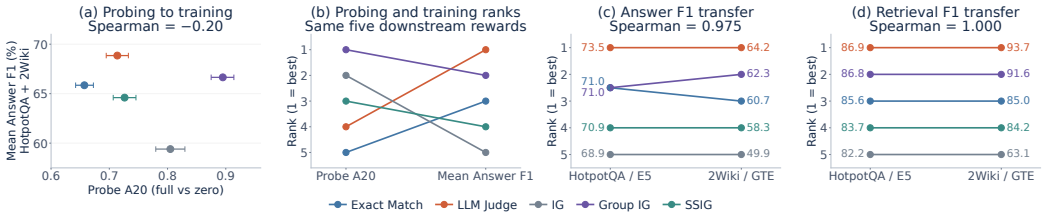


Figure 3: Probing and training outcomes for the same five downstream rewards. (a) HotpotQA A_{20} versus mean Answer F1 over HotpotQA and 2Wiki; error bars are probe-only 95% confidence intervals. (b) Corresponding reward ranks. (c,d) Post training ranks comparison between datasets for Answer F1 and Retrieval F1.

4.3 Retrieval-Only Adaptation in Practical Settings

LLM-as-a-Judge achieves the highest average answer quality among downstream rewards, within 1.0 Answer F1 point of Supporting Fact Reward. We therefore select it for retrieval-only adaptation in two more practical settings: Full-Wiki QA and multi-turn Dialogue QA.

Full-Wiki question answering. Following Search-R1 (Jin et al., 2025b), we retrieve from Wiki-18, containing 21M passages of approximately 100 words each. Training uses 120,631 question–

Training reward	Retrieval		Answer Metrics			
	EM	F1	EM	F1	Judge-4B	Judge-32B
HotpotQA (distractor), backbone multilingual-e5-large, dev (7,405 questions)						
Exact Match (r_{EM})	75.1	85.6	56.7	71.0	79.6	83.4
LLM-as-a-Judge	76.2	**86.9**	**58.5**	**73.5**	**82.7**	**86.2**
Information Gain	69.8	82.2	54.7	68.9	76.9	76.9
Group IG (Eq. 3)	**77.0**	86.8	56.7	71.0	78.9	78.9
Semantic Entropy Probes (r_{SEP})	69.1	82.5	54.5	68.4	77.0	81.1
SSIG (r_{SSIG})	71.7	83.7	56.8	70.9	79.4	83.2
Reference: No training (E5 zero-shot)	55.2	74.1	48.5	61.7	63.5	74.0
Reference: Support facts reward	89.0	93.0	58.7	73.4	81.0	78.9
Reference: Oracle context (gold paragraphs)	100	100	61.4	76.4	85.1	85.2
2WikiMultiHopQA, backbone gte-multilingual-base, dev without `bridge_comparison`						
Exact Match (r_{EM})	77.1	85.0	50.7	60.7	66.3	72.5
LLM-as-a-Judge	**89.1**	**93.7**	**52.2**	**64.2**	**71.7**	**79.4**
IG (single reference)	46.1	63.1	37.5	49.9	55.9	54.9
Group IG (Eq. 3)	85.3	91.6	49.8	62.3	70.1	69.2
SSIG (r_{SSIG})	73.6	84.2	47.7	58.3	64.2	69.6
Reference: No training (GTE zero-shot)	31.7	56.7	26.2	34.7	38.8	41.8
Reference: Support facts reward	98.6	99.3	54.2	66.3	74.1	83.5
Reference: Oracle context (gold paragraphs)	100	100	52.6	65.4	73.5	72.4

Table 1: Results for retrieval only training with downstream rewards. Retrieval EM is the fraction of questions for which all gold paragraphs were selected. Answers are generated by the frozen Qwen3-4B Answerer from the final context. Judge-4B and Judge-32B are accuracies according to Qwen3-4B and Qwen3-32B used as LLM judges. Oracle context represents ideal policy that always selects supporting chunks.

Method	Single-hop QA			Multi-hop QA				Avg.
	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	
Search-R1 paper: Qwen2.5-3B-Base/Instruct								
Direct Inference	10.60	28.80	10.80	14.90	24.40	2.00	2.40	13.40
CoT	2.30	3.20	0.50	2.10	2.10	0.20	0.00	1.50
IRCoT	11.10	31.20	20.00	16.40	17.10	6.70	24.00	18.10
Search-o1	23.80	47.20	26.20	22.10	21.80	5.40	32.00	25.50
RAG	34.80	54.40	38.70	25.50	22.60	4.70	8.00	27.00
SFT	24.90	29.20	10.40	18.60	24.80	4.40	11.20	17.60
R1-base	22.60	45.50	17.30	20.10	26.80	5.50	22.40	22.90
R1-instruct	21.00	44.90	17.10	20.80	27.50	6.00	19.20	22.40
Rejection Sampling	29.40	48.80	33.20	24.00	23.30	5.90	21.00	26.50
Search-R1-base	40.60	58.70	43.50	28.40	27.30	4.90	8.80	30.30
Search-R1-instruct	34.10	54.50	37.80	32.40	31.90	10.30	26.40	32.50
Our evaluation								
GTE zero-shot	34.40	56.83	38.07	27.05	17.40	6.25	19.20	28.46
Q-RAG (Qwen2.5-3B Judge)	30.33	50.00	37.11	32.10	31.78	9.81	26.40	31.08
Q-RAG (Qwen3-4B Judge)	33.41	57.31	39.85	37.74	32.29	11.63	28.80	34.43
Q-RAG (SF reward, OOD)	11.19	25.26	6.13	10.06	5.41	1.16	4.00	9.03

Table 2: Full-Wiki exact match (%) with answer aliases on seven datasets, retrieved over Wiki-18 (21M passages). Upper block: from Search-R1 paper Jin et al. (2025b). Lower block: each method is used purely as a retriever, and its passages are answered by the frozen Answerer. GTE zero-shot is the same 6-step loop with an untrained encoder.

answer pairs from HotpotQA and 2Wiki, with retrieval over the entire corpus and no supporting-fact supervision. Evaluation covers seven benchmarks, five unseen during Q-RAG training. Only the 305M-parameter `gte-multilingual-base` state encoder is updated; corpus embeddings and the Answerer LLM remain frozen. Retrieval uses six steps and at most two passages per article; further details are in Appendix B.

We compare with Search-R1, which trains the LLM itself and updates roughly ten times as many parameters as Q-RAG (3B LLM model vs 300M GTE embedder). We also include the standard retrieval baselines reported in its paper. Untrained GTE serves as a sanity check that training improves answer quality. Finally, Q-RAG with Support Facts Reward requires evidence labels unavailable in our Full-Wiki setup, so we evaluate a Q-RAG checkpoint trained on related labeled data. This tests whether the benefit of evidence supervision outweighs the cost of transferring from fixed candidate pools to Full-Wiki retrieval. Both GTE zero-shot and Q-RAG(SF Reward) use Qwen3-4B Answerer.

With the Qwen3-4B Answerer, training raises mean EM from 28.46 to 34.43 ($+5.97$ points; Table 2). Gains are concentrated on multi-hop datasets, including the unseen MuSiQue and Bamboogle. The transferred Support Facts checkpoint reaches only 9.03, below GTE zero-shot. Thus, its stronger fixed-pool performance (see 4.2) does not translate into better Full-Wiki results, whereas judge feedback enables adaptation directly to the task without evidence annotations. Q-RAG with Qwen2.5-3B provides a closer comparison to Search-R1, although the comparison is not fully controlled. Both approaches use Qwen2.5-3B during training and for answer generation, but Q-RAG keeps the LLM frozen and fine-tunes only the retrieval encoder, updating roughly $1/10$ as many parameters. In this setting, Q-RAG reaches 31.08 EM, between Search-R1-base (30.30) and Search-R1-instruct (32.50), showing competitive answer quality.

Multi-Turn Dialogue QA Laban et al. (2025) show that language model performance drops substantially when a task is revealed gradually across user turns, so the model may loose important information in the long conversation or rely on premature conclusions.

This setting provides annotated evidence, allowing us to compare Q-RAG trained with LLM-as-a-Judge Reward against Q-RAG trained with Support Fact Reward. Both variants are trained and evaluated on the same dialogue distribution, avoiding the distribution shift involved in the Full-Wiki transfer experiment. Q-RAG selects useful information from the accumulated conversation right before the system decides to answer the question. Our User Simulator and Strategy Classifier differ from those in the original work (Laban et al., 2025).

Table 3: Multi-turn dialogue QA on GSM8K: accuracy (%) of the evaluated assistant under different methods. User simulator and Strategy Classifier: Qwen3.6-35B-A3B throughout.

Experiments	Dataset	Evaluated Assistant	Acc. %
Original paper	gsm8k	phi4	60.1
	gsm8k	Llama 3.1-8B-Instruct	43.5
Q-RAG LLM-as-a-Judge Reward (ours)	gsm8k	phi4 + Q-RAG	70.0 (+9.9)
	gsm8k	Llama 3.1-8B-Instruct + Q-RAG	53.0 (+9.5)
Q-RAG Support Fact Reward (ours)	gsm8k	phi4 + Q-RAG	**85.0 (+24.9)**
	gsm8k	Llama 3.1-8B-Instruct + Q-RAG	**58.0 (+14.5)**

With LLM-as-a-Judge Reward, accuracy increases from 60.1% to 70.0% for Phi-4 and from 43.5% to 53.0% for Llama 3.1-8B-Instruct. Support Fact Reward achieves 85.0% and 58.0%, respectively, exceeding LLM-as-a-Judge Reward by 15.0 and 5.0 percentage points. The two Q-RAG reward variants, however, share the same evaluation protocol. Their comparison shows that, although LLM-as-a-Judge Reward is useful, it can remain substantially weaker than Support Fact Reward when evidence annotations are available and training and evaluation distributions match. Table 3 summarizes the GSM8K results. Other related methods, including ERGO (Khalid et al., 2025), MAIGO (Zheng et al., 2026), RLAAR (Li et al., 2025a), and Memory-Augmented RL (Luo et al., 2026), also report results on sharded GSM8K, but we do not include them in Table 3: their evaluations omit the system model and user model from the setup, so the numbers are not directly comparable under the multi-turn protocol used here. A breakdown by the number of missing shards at the first correct answer is in Appendix E.

5 Conclusion

We compared downstream rewards for training a multi-step retrieval module. We also investigated whether useful rewards could be identified before retriever training by assessing their ability to distinguish annotated supporting evidence from distractors. Our proposed Group Information Gain reward achieved the strongest evidence contrastiveness in this controlled comparison. However, contrastiveness did not reliably predict answer quality after training: across the five rewards evaluated on both datasets, its rank correlation with mean Answer F1 was $\rho = -0.20$. Thus, stronger discrimination of annotated evidence alone was insufficient to identify the most effective training reward.

One possible explanation is that supporting-fact annotations do not exactly describe the information needed by a particular frozen answerer. Annotations may contain errors or include evidence that is redundant given the model's parametric knowledge. Context selection might also affect distraction and hallucination: even passages without annotated supporting facts could differ in how much they interfere with the model's use of relevant information.

LLM-as-a-Judge achieved the strongest answer quality among the evaluated downstream rewards, using the same frozen model to generate and assess answers. This result is consistent with the benefit of feedback that directly evaluates the final task objective. Group IG outperformed Information Gain and achieved the second-highest mean Answer F1 among the downstream rewards. Experiments on Full-Wiki demonstrate the usefulness of judge feedback for retrieval-only adaptation on the task without support fact annotation. In the multi-turn dialogue setting, supporting-fact supervision outperformed judge feedback. These results support LLM-as-a-Judge as an effective alternative when evidence labels are unavailable, while favoring supporting-fact supervision when reliable annotations are available and training and evaluation conditions are closely matched.

Limitations

Each configuration in Sec. 4.2 is a single run on a single seed, so differences of a few tenths of a point between neighbouring rewards should not be read as ordering them; the gaps we rely on are the large ones and consistency in ranking between different Answer and Retrieval Metrics. All reward feedback also comes from one model family, so whether the results survives differences between Answerer models, Reward models and judges is untested.

AI USE STATEMENT

AI tools were used only to polish the manuscript text, optimize the structure of the tables, to search for relevant literature and improve grammar and readability. All experimental design, all reported results and all claims made in the paper were produced and verified by the authors.

REPRODUCIBILITY STATEMENT

The reward definitions used for training are given in full in Appendix A, one subsection per reward, including the judging prompt and the offline candidate construction for Group IG. Training configurations for all three environments, covering backbones, step budgets, and other information can be found in Appendix B or in the code provided for the paper. The LLM Answerer decoding settings and the definitions of every reported metric are in Appendix C. The retrieval corpus, the evaluation sets and the LLM Answerer are public, and the code is available at the anonymous repository linked in the abstract.

REFERENCES

Renat Aksitov, Sobhan Miryoosefi, Zonglin Li, Daliang Li, Sheila Babayan, Kavya Kopparapu, Zachary Fisher, Ruiqi Guo, Sushant Prakash, Pranesh Srinivasan, Manzil Zaheer, Felix Yu, and Sanjiv Kumar. Rest meets react: Self-improvement for multi-step reasoning LLM agent, 2023. URL https://arxiv.org/abs/2312.10003.

Marie Al-Ghossein, Emile Contal, and Alexandre Robicquet. Iclerb: In-context learning embedding and reranker benchmark. *ArXiv*, abs/2411.18947, 2024. URL https://api.semanticscholar.org/CorpusID:274423043.

Meng Cao, Lei Shu, Lei Yu, Yun Zhu, Nevan Wichers, Yinxiao Liu, and Lei Meng. Beyond sparse rewards: Enhancing reinforcement learning with language model critique in text generation, 2024. URL https://arxiv.org/abs/2401.07382.

Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, Fan Yang, et al. ReSearch: Learning to reason with search for LLMs via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025.

Yunmo Chen, Tongfei Chen, Harsh Jhamtani, Patrick Xia, Richard Shin, Jason Eisner, and Ben Van Durme. Learning to retrieve iteratively for in-context learning. In *EMNLP*, November 2024. URL https://www.microsoft.com/en-us/research/publication/learning-to-retrieve-iteratively-for-in-context-learning/.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630:625–630, 2024.

Matteo Gallici, Mattie Fellows, Benjamin Ellis, Bartomeu Pou, Ivan Masmitja, Jakob Nicolaus Foerster, and Mario Martin. Simplifying deep temporal difference learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Xanh Ho, Anh-Khoa Duc Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, 2020.

Senkang Hu, Yong Dai, Yuzhi Zhao, Yihang Tao, Yu Guo, Zhengru Fang, Sam Tak Wu Kwong, and Yuguang Fang. Optimizing agentic reasoning with retrieval via synthetic semantic information gain reward, 2026. URL https://arxiv.org/abs/2602.00845.

Bowen Jin, Jinsung Yoon, Priyanka Kargupta, Sercan O. Arik, and Jiawei Han. An empirical study on reinforcement learning for reasoning-search interleaved LLM agents, 2025a. URL https://arxiv.org/abs/2505.15117.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025b.

Haziq Mohammad Khalid, Athikash Jeyaganthan, Timothy Do, Yicheng Fu, Sean O'Brien, Vasu Sharma, and Kevin Zhu. ERGO: Entropy-guided resetting for generation optimization in multi-turn language models. *arXiv preprint arXiv:2510.14077*, 2025. URL https://arxiv.org/abs/2510.14077.

Jannik Kossen, Jiatong Han, Muhammed Razzak, Lisa Schut, Shreshth Malik, and Yarin Gal. Semantic entropy probes: Robust and cheap hallucination detection in LLMs, 2024. URL https://arxiv.org/abs/2406.15927.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. *arXiv preprint arXiv:2505.06120*, 2025. URL https://arxiv.org/abs/2505.06120.

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. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, 2020.

Ming Li, Pei Chen, Zhenhao Zhang, Tao Yang, Xinyang Zhang, Han Li, Tianyu Cao, Ming Zeng, Zhuofeng Wu, Meng Jiang, Huasheng Li, Lihong Li, and Bing Yin. Mitigating lost in multi-turn conversation via curriculum RL with verifiable accuracy and abstention rewards. *arXiv preprint arXiv:2510.18731*, 2025a. URL https://arxiv.org/abs/2510.18731.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*, 2025b.

Shu Tong Luo, Wenqin Liu, Rui Liu, Mingming Gong, and Jiaxian Guo. Multi-turn reasoning when context arrives in pieces: Scalable sharding and memory-augmented RL. *arXiv preprint arXiv:2606.12941*, 2026. URL https://arxiv.org/abs/2606.12941.

Thang Nguyen, Peter Chin, and Yu-Wing Tai. Reward-RAG: Enhancing RAG with reward driven supervision. *ArXiv*, abs/2410.03780, 2024. URL https://api.semanticscholar.org/CorpusID:273186680.

Anagha Radhakrishna Palandye, Rebecca Glick, and Osheen Kaul. Reward granularity in rlvr: Comparing process and outcome reward structures for mathematical reasoning in small language models, 2026. URL https://arxiv.org/abs/2607.02869.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. Replug: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8364–8377, 2024.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-Searcher: Incentivizing the search capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

Artyom Sorokin, Nazar Buzun, Leonid Pugachev, and Mikhail Burtsev. Explain my surprise: Learning efficient long-term memory by predicting uncertain outcomes. In *Advances in Neural Information Processing Systems*, volume 35, pp. 36875–36888. Curran Associates, Inc., 2022.

Artyom Sorokin, Nazar Buzun, Alexander Anokhin, Oleg Inozemcev, Egor Vedernikov, Petr Anokhin, Mikhail Burtsev, Alexey Trushkov, Wenshuai Yin, and Evgeny Burnaev. Q-RAG: Long context multi-step retrieval via value-based embedder training. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2511.07328.

Charles Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904. doi: 10.2307/1412159. URL https://doi.org/10.2307/1412159.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Haoyan Yang, Zhitao Li, Yong Zhang, Jianzong Wang, Ning Cheng, Ming Li, and Jing Xiao. PRCA: Fitting black-box large language models for retrieval question answering via pluggable reward-driven contextual adapter. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5364–5375, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.326. URL https://aclanthology.org/2023.emnlp-main.326/.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, 2018.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models, 2024. URL https://arxiv.org/abs/2401.10020.

Efstratios Zaradoukas, Davide Gabrielli, Bardh Prenkaj, and Gjergji Kasneci. Beyond binary rewards: A comparative study of reward design for reinforcement unlearning, 2026. URL https://arxiv.org/abs/2607.27968.

Hanning Zhang, Juntong Song, Juno Zhu, Yuanhao Wu, Tong Zhang, and Cheng Niu. Opengenalign: A preference dataset and benchmark for trustworthy reward modeling in open-ended, long-context generation, 2025a. URL https://arxiv.org/abs/2501.13264.

Jiahao Zhang, Haiyang Zhang, Dongmei Zhang, Liu Yong, and Shen Huang. End-to-end beam retrieval for multi-hop question answering. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1718–1731, 2024.

Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, Ruiming Tang, and Xiangyu Zhao. Process vs. outcome reward: Which is better for agentic RAG reinforcement learning, 2025b. URL https://arxiv.org/abs/2505.14069.

Xuan Zhang and Wei Gao. Reinforcement retrieval leveraging fine-grained feedback for fact checking news claims with black-box LLM. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (eds.), *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 13861–13873, Torino, Italia, May 2024. ELRA and ICCL. URL https://aclanthology.org/2024.lrec-main.1209/.

Yibo Zhao, Zichen Ding, Jiayi Wu, Zun Wang, and Xiang Li. Retrieval, reward, and training protocols: What matters in training search agents?, 2026. URL https://arxiv.org/abs/2605.27881.

Haoyu Zheng, Yun Zhu, Shu Yuan, Shangming Chen, Qing Wang, Wenqiao Zhang, Jun Xiao, and Yueting Zhuang. MAIGO: Mitigating lost-in-conversation with history-cleaned on-policy self-distillation. *arXiv preprint arXiv:2605.27186*, 2026. URL https://arxiv.org/abs/2605.27186.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, 2023.

A Detailed Description of Experimental Rewards

Shared notation. We use the notation of Section 3.1 and omit the example index. Let $\mathbb{I}[\cdot]$ denote an indicator, and let $\mathrm{norm}(\cdot)$ lowercase a string, remove punctuation and the English articles `a`, `an`, and `the`, and collapse whitespace. A small $\varepsilon > 0$ stabilizes denominators. For semantic-uncertainty rewards, $\mathcal{K}(x, z)$ denotes the clusters of M independent answers $g^{(1)}, \ldots, g^{(M)} \sim p_\theta(\cdot \mid x, z)$, formed using bidirectional entailment checks Farquhar et al. (2024), where $z \in \{c_0, c\}$. For nonnegative cluster weights $\mathbf{u}$, write $\mathsf{H}(\mathbf{u}) = -\sum_\kappa u_\kappa \log u_\kappa$, with $0 \log 0 = 0$; this is entropy when the weights sum to one. Frequency estimates and model-likelihood masses are distinguished below.

A.1 Support Facts (reference)

This reward supervises retrieval directly and never queries the LLM Answerer. Let $F^\star \subseteq \mathcal{C}$ be the annotated evidence set for the query, obtained from the `supporting_facts` field and matched by article title. The reward is granted only when the final context covers all of it:

$$r_{\mathrm{SF}}(c) = \mathbb{I}[F^\star \subseteq c]\,. \tag{5}$$

Additional chunks are not penalized, so partial coverage receives nothing while full coverage plus distractors receives the same reward as an exact match. Two properties make this a reference rather than a comparable signal. It requires evidence annotations, which exist in the fixed-pool setting but not for an open corpus. And it coincides with the Retrieval EM metric used for evaluation, so a policy trained on it is optimizing the retrieval measure it is scored by, which the answer-based and likelihood-based rewards are not.

A.2 Exact Match (EM)

The reward compares the normalized generated and reference answers:

$$r_{\mathrm{EM}}(\hat{y}, y) = \mathrm{EM}(\hat{y}, y) = \mathbb{I}[\mathrm{norm}(\hat{y}) = \mathrm{norm}(y)]\,. \tag{6}$$

A.3 Token-level F1

Let $\mathcal{T}(\cdot)$ be the multiset of tokens after normalization. Precision and recall are

$$P(\hat{y}, y) = \frac{|\mathcal{T}(\hat{y}) \cap \mathcal{T}(y)|}{\max(1, |\mathcal{T}(\hat{y})|)}, \tag{7}$$

$$R(\hat{y}, y) = \frac{|\mathcal{T}(\hat{y}) \cap \mathcal{T}(y)|}{\max(1, |\mathcal{T}(y)|)}, \tag{8}$$

where the intersection counts token multiplicities. The reward is

$$r_{F_1}(\hat{y}, y) = F_1(\hat{y}, y) = \frac{2P(\hat{y}, y)R(\hat{y}, y)}{P(\hat{y}, y) + R(\hat{y}, y) + \varepsilon}. \tag{9}$$

A.4 Semantic Similarity

Let $\mathbf{e}$ be an answer encoder with output dimension d_{sem}. The reward is the cosine similarity between the generated and reference answer embeddings:

$$r_{\mathrm{sem}}(\hat{y}, y) = s_{\cos}(\hat{y}, y) = \frac{\mathbf{e}(\hat{y})^\top \mathbf{e}(y)}{\|\mathbf{e}(\hat{y})\|_2 \|\mathbf{e}(y)\|_2 + \varepsilon}. \tag{10}$$

A.5 LLM-as-a-Judge

The judge receives the query, the answer generated from the retrieved context, and the reference answer, and returns a binary verdict:

$$r_{\mathrm{judge}}(x, c) = v_\tau(x, \hat{y}, y)\,, \qquad \hat{y} \sim p_\theta(\cdot \mid x, c). \tag{11}$$

We use the same frozen Qwen3-4B for p_θ and v_τ, with reasoning disabled, so the judge adds no model to the pipeline. One episode therefore costs one generation followed by one judging call, which is the most expensive of the rewards we compare that does not sample repeatedly.

Unlike Exact Match, the verdict does not depend on surface form, so prefatory phrases and formatting differences no longer zero out a correct answer. Two limitations follow from the construction. The reward inherits whatever the judge gets wrong, and since the judge and the LLM Answerer are the same model, their errors are correlated: an answer the LLM Answerer phrases in a way it also finds convincing may be accepted regardless of correctness. At evaluation time we therefore report a second verdict from Qwen3-32B, which is a metric and never a training signal.

A.6 Information Gain

This reward asks the LLM Answerer for a likelihood rather than a generation. It scores how much the retrieved context raises the log-likelihood of the reference answer relative to the context-free baseline $\beta_x(y) = \log p_\theta(y \mid x, c_0)$, which is precomputed once per dataset:

$$r_{\mathrm{IG}}(x, c) = \log p_\theta(y \mid x, c) - \beta_x(y), \tag{12}$$

where $\log p_\theta(y \mid x, c)$ is summed over tokens. One episode costs a single forward pass over the reference and no sampling, making this the cheapest reward we compare. The signal is continuous and unbounded in both directions: positive when the context makes the reference more predictable, negative when it makes it less so.

The weakness is that the reference is scored in isolation. A context that raises $\log p_\theta(y \mid x, c)$ may raise the likelihood of competing answers by as much or more, and the reward cannot tell the two cases apart, because it never evaluates any string other than y. Group IG (Appendix A.7) addresses exactly this by scoring a set of candidates and contrasting the judged-correct ones against the judged-incorrect ones.

A.7 Group Information Gain (Group IG)

For each query, sample K candidate answers under the context-free conditioning and add the reference:

$$g_k \sim p_\theta(\cdot \mid x, c_0), \quad k = 1, \ldots, K, \qquad \mathcal{G} = \{g_1, \ldots, g_K\} \cup \{y\}. \tag{13}$$

The judge partitions the candidates into

$$\mathcal{G}^+ = \{g \in \mathcal{G} : v_\tau(x, g, y) = 1\}, \qquad \mathcal{G}^- = \{g \in \mathcal{G} : v_\tau(x, g, y) = 0\}. \tag{14}$$

Candidate generation, judging, and the baseline likelihoods $\beta_x(g)$ are computed offline. During training, Group IG contrasts the mean log-likelihood shifts of the judged-correct and judged-incorrect candidates:

$$\begin{aligned} r_{\mathrm{GroupIG}}(x, c) = \frac{1}{|\mathcal{G}^+|} \sum_{g \in \mathcal{G}^+} \big[\log p_\theta(g \mid x, c) - \beta_x(g)\big] \\ - \frac{1}{|\mathcal{G}^-|} \sum_{g \in \mathcal{G}^-} \big[\log p_\theta(g \mid x, c) - \beta_x(g)\big]. \end{aligned} \tag{15}$$

An average over an empty subset is defined as zero.

A.8 Semantic Entropy Probes (SEP)

For $\kappa \in \mathcal{K}(x, z)$, let n_κ be the number of sampled answers in that cluster. The empirical semantic entropy uses cluster frequencies:

$$\widehat{H}_{\mathrm{sem}}(x, z) = \mathsf{H}\big(\widehat{\boldsymbol{\ell}}(x, z)\big), \qquad \hat{\ell}_\kappa(x, z) = \frac{n_\kappa}{M}. \tag{16}$$

A logistic probe predicts a high/low uncertainty label obtained by thresholding this estimate offline Kossen et al. (2024). Let $h(x, c) \in \mathbb{R}^{d_h}$ be a hidden representation from a single answer generation, such as the second-last generated token. The probe is

$$\widehat{u}_{\mathrm{SEP}}(x, c) = \sigma\big(w^\top h(x, c) + b\big), \qquad \sigma(u) = \frac{1}{1 + e^{-u}}, \tag{17}$$

where $w \in \mathbb{R}^{d_h}$ and $b \in \mathbb{R}$ are fitted to the binary uncertainty labels. The reward favors low predicted uncertainty:

$$r_{\mathrm{SEP}}(x, c) = 1 - \widehat{u}_{\mathrm{SEP}}(x, c). \tag{18}$$

A.9 Synthetic Semantic Information Gain (SSIG)

SSIG Hu et al. (2026) compares semantic uncertainty before and after retrieval, using $z = c_0$ and $z = c$. For each sampled cluster, define its model-likelihood mass and the associated uncertainty score:

$$\ell_\kappa(x, z) = \sum_{g \in \kappa} p_\theta(g \mid x, z), \qquad H^{\mathrm{lik}}_{\mathrm{sem}}(x, z) = \mathsf{H}\big(\boldsymbol{\ell}(x, z)\big). \tag{19}$$

These masses are not renormalized over the sampled clusters. One variant rewards a decrease in this score:

$$IG^{\mathrm{sem}}(x, c) = H^{\mathrm{lik}}_{\mathrm{sem}}(x, c_0) - H^{\mathrm{lik}}_{\mathrm{sem}}(x, c). \tag{20}$$

A second variant uses the semantic class κ_y containing the reference answer and rewards its log-likelihood increase:

$$IG^{\mathrm{cls}}(x, c) = \log \ell_{\kappa_y}(x, c) - \log \ell_{\kappa_y}(x, c_0). \tag{21}$$

The final reward combines the same normalized EM defined above with either information-gain variant:

$$r_{\mathrm{SSIG}}(x, c) = \mathrm{EM}(\hat{y}, y) + \eta\, IG(x, c), \tag{22}$$

where $IG \in \{IG^{\mathrm{sem}}, IG^{\mathrm{cls}}\}$ and η controls its weight, independently of the TD parameter λ.

B Training details

Policy and algorithm. In all settings the policy is Q-RAG trained with PQN on a terminal reward, so the reward is observed once per episode and the λ-returns carry it back over the T retrieval decisions.

Fixed-pool setting. Each action selects one of the ten HotpotQA paragraphs, or the corresponding 2Wiki pool, with $T = 2$ steps per question. Both encoders are trained. The state and action encoders are initialized from `multilingual-e5-large` on HotpotQA and from `gte-multilingual-base` on 2WikiMultiHopQA; because the backbones differ, rewards are compared within a dataset and never across the two. Training uses the train splits of each dataset. For Group IG, $K = 8$ candidates are sampled per question under the context-free conditioning, judged offline, and cached together with the baseline likelihoods β_x before training starts.

Full-wiki setting. Retrieval runs over Wiki-18 with $T = 6$ steps. A trainable GTE state encoder of 12 layers and dimension 768 supplies both the index query and $E_s(s_t)$, while E_a consists of frozen GTE corpus embeddings, so only the state encoder is updated. Each step retrieves the 100 nearest passages and masks those already selected and those whose article already contributes $N = 2$ passages. Training uses the filtered HotpotQA and 2Wiki mixture of 120,631 questions with LLM-as-a-Judge feedback; the five remaining evaluation datasets are never trained on.

Checkpoint selection. Checkpoints are selected on 1,000 HotpotQA dev questions, which are also part of the final evaluation set; we note this rather than treat the selection split as held out. All reported runs use a single seed without repeats.

Hyperparameters. Hyperparameters (and other implementation details) can be found in the code which is available at `https://anonymous.4open.science/r/Q-RAG-feedbacks-A524/`.

C Evaluation details

LLM Answerer. Every retriever under comparison feeds the same frozen Qwen3-4B LLM Answerer, decoding at temperature 0 with reasoning disabled and at most 1,000 tokens. The LLM

Answerer prompt is identical during training and evaluation, so the reward a policy is trained on and the metric it is scored by differ only in how the answer is compared with the reference. In the fixed-pool setting the selected paragraphs are passed in their original order. In the full-wiki setting the LLM Answerer receives the deduplicated passages returned by each method, and for the Search-R1 baselines their own generated answers are discarded so that only retrieval is compared.

Retrieval metrics. Retrieval EM is the fraction of questions for which every gold paragraph was selected, without penalizing additional selections, so it is the quantity that the Support Facts reward optimizes directly. Retrieval F1 is the set-level F1 between the selected and the gold paragraph sets and does penalize extra selections. Both require evidence annotations and are reported only in the fixed-pool setting.

Answer metrics. Answer EM and Answer F1 apply the normalization of Appendix A.2 to the LLM Answerer output. Judge-4B is the verdict of the same Qwen3-4B used as v_τ during training, and Judge-32B is an independent verdict from Qwen3-32B, reported to check that conclusions do not rest on the judge that produced the training signal. In the full-wiki setting we report exact match against the set of accepted answer aliases, in percent, averaged with equal weight over the seven evaluation datasets.

Evaluation sets. The fixed-pool results use the full HotpotQA dev split of 7,405 questions and the 2Wiki dev split without `bridge_comparison`. The full-wiki results use the seven Search-R1 evaluation sets, 51,713 questions in total, of which only HotpotQA and 2Wiki were seen during training.

Prompts.

Judge prompt

```
You are an answer verification system for question-answering
tasks.  You are given a QUESTION, PREDICTED ANSWER and GROUNDTRUTH
ANSWER. Compare the predicted and ground-truth answers for semantic
equivalence.  Ignore harmless wording, formatting, numeric, and date
representation differences.  If the predicted answer is only partially
correct or misses key information, mark it incorrect.  Your final
answer must be exactly "Final Answer:  CORRECT" or "Final Answer:
INCORRECT".
```

QA prompt

```
You are a precise question answering assistant.  Answer questions
based on the relevant information within the provided context and
ignore misleading text.  Keep your reasoning brief.  Make your final
answer as short as possible.  ALWAYS give your final answer after the
words "Final Answer:".
```

D When to Stop Retrieving

The retriever uses the same number of search steps for every question. We ask how much accuracy an ideal stopping rule could recover, and how much of it rules built on already available signals do recover. For every question we run retrieval once with a horizon of T steps and score the LLM Answerer on every prefix $1, \ldots, T$, which gives a per-question, per-step correctness matrix $y_{q,t} \in \{0, 1\}$ under em_{alias}; rules are then evaluated on this matrix without further LLM Answerer calls, so all rows of Table 4 are comparable. The stand reuses the LLM Answerer, prompt and scoring code of the main pipeline, and reproduces the run's step-6 em_{alias} to within 0.07 p.p..

Headroom is smaller than it first appears. An oracle stopping at any prefix the LLM answers correctly gains 6.87 and 3.31 p.p. at $T = 6$, and 8.85 and 5.83 p.p. at $T = 10$. Most of this is not learnable: of the 509 HotpotQA questions it recovers at $T = 6$, 288 (56.6%) are correct on exactly one prefix out of six, a single lucky sample from the LLM Answerer. Restricting the oracle to two consecutive correct prefixes leaves $+2.98$ and $+1.37$ p.p. at $T = 6$, and $+4.79$ and $+3.06$ p.p. at

Stopping rule	T	HotpotQA (7405 q.) em$_{\text{alias}}$	$\bar{t}$	Δ_6	$\Delta_\star$	MuSiQue (2417 q.) em$_{\text{alias}}$	$\bar{t}$	Δ_6	$\Delta_\star$
Fixed horizon (reference)									
Stop at step 6	10	37.91	6.00	—	—	11.75	6.00	—	—
Stop at step 9	10	37.84	9.00	-0.07	-0.07	12.62	9.00	$+0.87$	$+0.00$
Best fixed step	10	37.91	6.00	$+0.00$	—	12.62	9.00	$+0.87$	—
Oracle stopping (upper bounds)									
Any correct prefix	6	44.75	—	$+6.87$	$+6.84$	15.02	—	$+3.31$	$+2.40$
≥ 2 consecutive	6	40.86	—	$+2.98$	$+2.95$	13.07	—	$+1.37$	$+0.45$
Any correct prefix	10	46.75	—	$+8.85$	$+8.84$	17.58	—	$+5.83$	$+4.96$
≥ 2 consecutive	10	42.70	—	$+4.79$	$+4.79$	14.81	—	$+3.06$	$+2.19$
Heuristic and probe-based rules, original checkpoint									
Critic value, $\arg\max_t Q(s_t, a_t)$	6	30.63	1.68	-7.25	-7.28	6.45	1.47	-5.26	-6.17
Probe on frozen states†	6	36.53	—	-1.82	-1.82	—	—	—	—
Answer stability, 3 in a row	6	37.33	4.22	-0.55	-0.58	11.25	4.90	-0.46	-1.37
LLM Answerer judgement $P(\text{STOP})$	9	37.96	6.94	$+0.05$	$+0.05$	12.29	7.58	$+0.54$	-0.33
Learned STOP *action, retrained state tower*									
Learned STOP action‡	9	37.14	7.51	-0.77	-0.77	12.08	6.92	$+0.33$	-0.54
same tower, fixed horizon (control)	9	37.02	9.00	-0.89	-0.89	12.25	9.00	$+0.50$	-0.37

Table 4: Adaptive stopping: headroom and realised gains. em$_{\text{alias}}$ in percent, $\bar{t}$ the mean stopping step, Δ_6 the difference against the six-step baseline and $\Delta_\star$ against the best fixed horizon on that dataset (step 6 for HotpotQA, step 9 for MuSiQue), both from unrounded values. †Trained on HotpotQA states, evaluated on a held-out split of 1481 questions whose six-step baseline is 38.35; not run on MuSiQue. ‡The action fires on 43.3% of HotpotQA and 68.6% of MuSiQue questions. Its control row is the same retrained tower run to the full horizon, which is the comparison that isolates the stopping decision from the retraining.

$T = 10$, the increase reflecting only the wider choice of moments. We take this restricted oracle as the target.

No available signal reaches that target. The four rules draw on the critic's action values, a probe on the frozen state representation, the stability of the LLM Answerer's answer across prefixes, and a judgement elicited from the LLM Answerer; none beats the best fixed horizon. Pooled over question-step pairs they look informative, with an AUC of 0.68 for the critic and 0.73 for the probe. Centred per question, so that only the *choice of step* is scored, the critic falls to 0.43 on HotpotQA and 0.38 on MuSiQue, below chance, while the probe reaches 0.685 against 0.687 for the step index alone: the 768-dimensional embedding contributes -0.002. These signals predict *which questions are hard*, not *when enough evidence has been collected*.

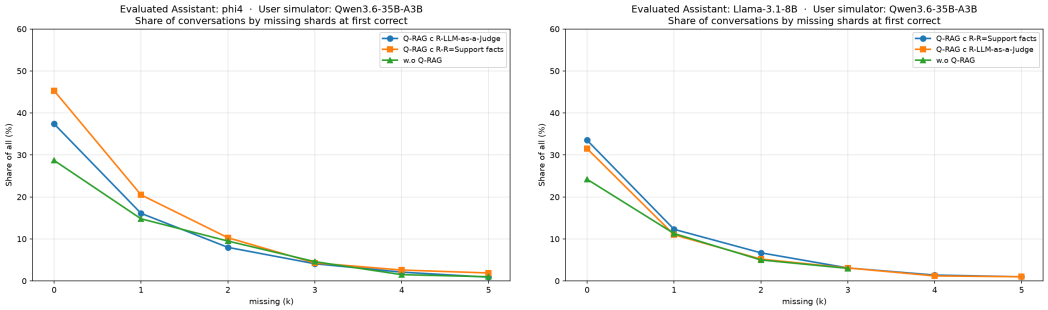
Compared at equal depth. Every rule stops earlier than the baseline, so the fair comparison is the fixed horizon of the same depth. Interpolating the per-prefix curve at $\bar{t}$ and subtracting leaves between $+0.15$ and $+1.05$ p.p. on HotpotQA and between $+0.16$ and $+0.91$ p.p. on MuSiQue. Each rule is thus slightly better than stopping blindly at its own average depth, but by an order of magnitude less than the restricted oracle, and the gain comes from stopping early rather than from stopping well.

A learned STOP action buys context, not accuracy. The state tower was trained to encode what to retrieve next, not whether what is retrieved suffices, so the last block of Table 4 learns the decision explicitly, extending the action space with a terminal STOP action scored by the same terminal reward. Retraining the tower for this action moves accuracy on its own, by -0.82 and -0.37 p.p. at nine steps, so the comparison that isolates the stopping decision is against the control row, the same tower run to the full horizon, and not against the original checkpoint. Measured that way the action changes accuracy by $+0.12$ p.p. on HotpotQA, over the 159 questions where the two rows disagree ($p = 0.53$), and by -0.17 p.p. on MuSiQue over 38 disagreements ($p = 0.63$); neither is distinguishable from stopping at the fixed horizon. What it does change is the context handed to the LLM Answerer: the action fires on 43.3% and 68.6% of questions and shortens retrieval to 7.51 and 6.92 chunks against 9, between 17 and 23% fewer. A learned stopping rule is therefore a cheap

substitute for the full horizon rather than a source of accuracy, which is where the heuristic rules also arrived by a different route.

E MULTI-TURN DIALOGUE QA: MISSING SHARDS AT FIRST CORRECT

For the sharded GSM8K setting of Section 4.3, Figure 4 reports the share of conversations by how many shards are still missing when the evaluated assistant first answers correctly. We compare the same three conditions as in Table 3: no Q-RAG, Q-RAG trained with LLM-as-a-Judge, and Q-RAG trained with Support Facts, for Phi-4 and Llama-3.1-8B-Instruct.



(a) Phi-4.

(b) Llama-3.1-8B-Instruct.

Figure 4: Share of sharded GSM8K conversations by the number of missing shards k at the first correct answer. User simulator: Qwen3.6-35B-A3B. Curves: without Q-RAG, Q-RAG with LLM-as-a-Judge reward, and Q-RAG with Support Facts reward.

On both assistants, Support Facts yields the largest mass at $k{=}0$ (correct with no shards missing), with LLM-as-a-Judge between Support Facts and the no-Q-RAG baseline. At higher k the three curves converge, so most of the accuracy gains in Table 3 come from answering earlier in the dialogue rather than from recovering conversations that remain missing many shards.