ATTRIBUTION IS NOT OPTIMIZATION: COUNTERFACTUAL REWARDS FOR LANGUAGE AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Holding a tool action fixed and replacing its output with a null observation isolates the output's contribution. It does not necessarily identify which action a policy should repeat. We formalize this attribution–optimization gap and introduce a finite-action audit that compares reward and terminal-utility update directions. Across three retrieval datasets, total belief gain is more aligned with terminal success than observation-residual credit (cosine 0.430 versus 0.114). The ordering persists in trainable-parameter gradients for retrieval (0.33 versus 0.09) and calculator use (0.51 versus 0.26). Yet residual credit closely tracks programmatic expression correctness (0.934). This contrast separates tool-output attribution from action learning. Restoring action-side credit improves local alignment throughout a controlled interpolation. We therefore introduce environment-marginalized action credit (EMAC), which preserves action-side credit while averaging tool-output variation. EMAC matches total belief gain when retrieval is deterministic. When the same query produces different relevant passage blocks, it raises held-out exact match from 0.263 to 0.312 and reduces normalized observation variance from 1.00 to 0.35. Under a matched reward-scoring budget, EMAC also improves exact match by 6.0 points in corrupted retrieval. On a 3B actor, the gain is 1.6 points.

1 INTRODUCTION

Language agents learn to search, browse, and use tools from outcome feedback Yao et al. (2023); Nakano et al. (2021); Asai et al. (2023); Qi et al. (2024); Jin et al. (2025). Dense rewards often turn changes in answer belief into step-level credit Wang et al. (2026); Auzina et al. (2026); Xie et al. (2026). The underlying intervention determines what that credit means. A whole-turn contrast includes reasoning, action, and observation. An observation-only contrast keeps the action and replaces the returned observation with a null value.

The observation-only contrast answers a precise question: what did the tool output add after the action was chosen? Policy learning asks a different question: which action should become more likely? The null-observation score varies across candidate actions and thus carries action-selection signal. Subtracting it can sharpen attribution while changing the learning direction (Figure 1).

This distinction matters when observation credit trains the action that produced it. The counterfactual may remove the part of belief change that separates a promising query or expression from an unhelpful one. Reward design must therefore test both the attributed component and the induced action preference. We connect these two views at the decision itself.

We formalize this *attribution–optimization gap*. At each state, we sample candidate actions and terminal continuations. A categorical policy over these candidates gives the exact local gradient of each reward and of sampled terminal success. Their alignment shows whether both objectives favor the same redistribution of action probability. We then test the same ordering in trainable-parameter gradients and full RL.

We audit retrieval queries and calculator expressions. In both settings, residual credit tracks whether the conditional tool output is useful. In the calculator, it closely aligns with programmatic expression correctness. Total belief gain is instead more aligned with final-answer success. The retrieval gap persists across 96 informative action sets and two scorers.

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.

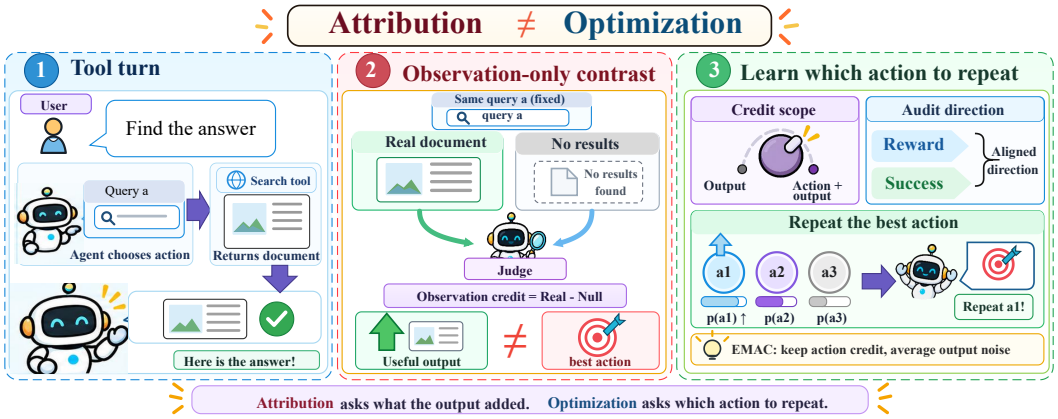

Figure 1: Observation attribution and action learning ask different questions. Holding a tool action fixed while replacing its output by a null observation isolates the output's contribution to answer belief. The action-side contribution also varies across candidate actions and can matter for the policy update. We test local reward directions and full RL outcomes separately.

The paper makes three contributions. First, it identifies the estimand mismatch created when an observation-attribution counterfactual becomes an action reward, with an exact compatibility condition for the audited decision. Second, it traces this mismatch from candidate logits to trainable parameters and learned policies. Third, it introduces EMAC, which preserves the action signal while marginalizing observation variation. EMAC matches total gain when retrieval is deterministic and improves exact match by 4.9 points when retrieval outputs vary.

2 Diagnosing the Attribution–Optimization Gap

At state s, an agent samples action $a \sim \pi_\theta(\cdot \mid s)$, and the environment returns $o \sim P(\cdot \mid s, a)$. A verifier supplies a scalar belief $B(s, a, o) = \log P_M(y^* \mid s, a, o)$ in the gold answer. Let $o_\emptyset$ be a null observation and $B_0(s) = \log P_M(y^* \mid s)$. The common total, residual, and action-side contrasts are

$$R_{\text{tot}}(s, a, o) = B(s, a, o) - B_0(s), \tag{1}$$

$$R_{\text{res}}(s, a, o) = B(s, a, o) - B(s, a, o_\emptyset), \tag{2}$$

$$R_{\text{act}}(s, a) = B(s, a, o_\emptyset) - B_0(s), \tag{3}$$

with the exact decomposition $R_{\text{tot}} = R_{\text{res}} + R_{\text{act}}$. The residual has a valid conditional meaning: the belief contribution of replacing the null observation by the realized output, holding a fixed. It does not follow that R_{res} is a valid advantage for choosing a.

For a state-only baseline $c(s)$, the score-function identity gives

$$\mathbb{E}_{a\sim\pi}[c(s)\nabla_\theta \log \pi_\theta(a \mid s)] = 0. \tag{4}$$

An action-dependent subtraction instead contributes $\mathbb{E}_a[b(s, a)\nabla_\theta \log \pi_\theta(a \mid s)]$, which is generally nonzero. Observation-residual credit uses exactly such a subtraction: the no-observation score $B(s, a, o_\emptyset)$ still depends on the query or expression chosen by the policy.

The classical baseline result identifies the algebraic hazard. Our object is the gap between two estimands: the conditional effect of a realized output and the utility-compatible preference over tool actions. The finite-action construction makes this gap measurable for structured language actions.

Holding the continuation mechanism fixed, let z parameterize the categorical policy over the audited actions. The current-decision target at s is

$$g_U(s) = \nabla_z \mathbb{E}_{a\sim\pi_z, o, \xi}[U(s, a, o, \xi)], \tag{5}$$

where U is terminal task utility and ξ denotes later policy and environment randomness. Continuations estimate the conditional value of each candidate action. A dense reward is locally compatible when its gradient has positive inner product with g_U. Attribution validity and this optimization criterion can agree, but neither implies the other.

Compatibility at a fixed state. For a finite action set, let $\tilde{r}(a) = r(a) - \mathbb{E}_\pi r$ and $\tilde{u}(a) = \mathbb{E}[U \mid s, a] - \mathbb{E}_\pi U$. Under a tabular softmax parameterization, the reward is a strict local ascent direction for terminal utility exactly when

$$\sum_a \pi(a)^2 \tilde{r}(a)\tilde{u}(a) > 0. \tag{6}$$

At a uniform policy, this is the covariance between the action-indexed reward and utility vectors, up to a positive constant. Substituting $R_{\text{res}} = R_{\text{tot}} - R_{\text{act}}$ exposes the ambiguity. Observation-residual credit deletes a term whose covariance with action utility can be positive, negative, or zero. Its attribution semantics do not determine that sign.

Derivation. Write $J_r(z) = \sum_a \pi_z(a) r(a)$ for categorical logits z. The softmax Jacobian gives

$$\frac{\partial J_r}{\partial z_i} = \sum_a r(a)\pi(a)(\mathbb{1}[a = i] - \pi(i)) \tag{7}$$

$$= \pi(i)\left(r(i) - \sum_a \pi(a) r(a)\right) = \pi(i)\tilde{r}(i). \tag{8}$$

The same expression holds for conditional terminal value $u(i) = \mathbb{E}[U \mid s, a_i]$. Their gradient inner product is therefore

$$\langle \nabla_z J_r, \nabla_z J_u \rangle = \sum_i \pi(i)^2 \tilde{r}(i)\tilde{u}(i), \tag{9}$$

which yields Eq. 6. At a uniform policy, the common positive scaling cancels from the cosine, leaving the cosine between the centered reward and utility vectors. The result is exact for the audited categorical current-decision policy: it tests whether moving probability among the sampled actions improves terminal utility while holding the continuation mechanism fixed.

Linearity also gives

$$\nabla J_{\text{res}} = \nabla J_{\text{tot}} - \nabla J_{\text{act}}. \tag{10}$$

Residual credit is more aligned precisely when subtracting the action-side gradient makes the result more parallel to ∇J_u.

The identity separates three cases. A constant action-side term has zero gradient. A positive inner product with ∇J_u means residualization removes a useful direction; a negative one means it removes a conflicting direction. The counterfactual specifies the removed component, but the action–utility relation determines its value for learning.

3 Auditing the current decision

For each state, we sample candidates $\mathcal{A}_s = \{a_1, \ldots, a_m\}$ (Figure 2). We execute every action and draw terminal continuations. Let u_i be mean terminal utility and r_i the proposed reward. At a uniform categorical policy $p_i = 1/m$, the candidate-logit gradient is exact:

$$\nabla_{z_i} \mathbb{E}_{a \sim p}[r(a)] = p_i(r_i - \bar{r}). \tag{11}$$

We report cosine similarity between the centered reward and utility vectors. Constant-utility sets are non-informative and excluded from the average.

With L continuations, the audit estimates utility as $\hat{u}_i = L^{-1} \sum_{\ell=1}^{L} U_{i\ell}$. Reward and utility are centered after all candidates are executed. The resulting gradient is exact for the sampled categorical decision; a negative cosine identifies a direction conflict.

Candidates are i.i.d. draws from the behavior sampler, so the primary retrieval and calculator audits assign equal mass to sampled actions. The audit-to-RL comparison in Section 4 instead uses $q_i \propto \exp\{m_i^{-1} \log \pi_\theta(a_i \mid s)\}$, where m_i is the number of action tokens, and computes the exact categorical-logit gradient $q_i(r_i - \mathbb{E}_q r)$. This keeps the candidate distribution close to the actor while avoiding the length concentration of unnormalized sequence likelihood.

We score all rewards on the same frozen actor, actions, and continuations. Candidate space tests the preferred alternatives; parameter space adds the actor's shared representation; RL measures the learned policy.

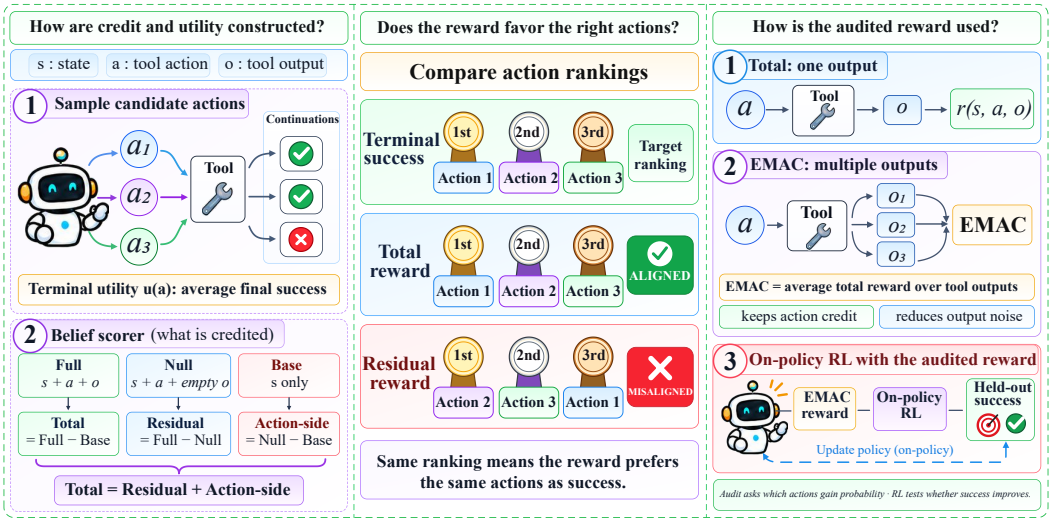

Figure 2: We construct total, observation-residual, and action-side belief rewards from a tool action and its output. The finite-action audit compares each categorical update direction with sampled terminal utility; RL measures held-out success.

3.1 Evaluation and Results

Evaluation setup. The gold answer is never included in the action or continuation prompt; it is used only to compute strict exact match and the teacher-forced verifier score. Candidate actions are sampled at temperature 0.7. We retain invalid tool calls in the raw counts but exclude them from finite categorical sets because no observation or reward is defined for them. Within a valid action set, we do not select candidates by reward or outcome. Terminal samples share the action and observation but use independent generation randomness. Appendix A gives the exact tool prompts and teacher-forced scoring target.

The primary statistic averages per-question gradient cosine. This avoids pooling candidate vectors with different reward scales. All rewards are evaluated on the same actions and terminal continuations within each question.

Retrieval. We audit Qwen3.5-9B Qwen Team (2026) query actions on Hotpot-style, 2WikiMultihopQA, and MuSiQue tasks Yang et al. (2018); Ho et al. (2020); Trivedi et al. (2022). The model samples four actions for each of 60 questions per dataset, and BM25 returns the top three passages from each closed corpus. A null context replaces only the returned passages. Each valid action receives ten sampled final answers. Across 96 informative action sets, total gain obtains cosine 0.430 versus 0.114 for residual document credit; the paired difference is 0.316. A second Qwen2.5-7B scorer Team (2024) evaluates the same base, full, and null contexts without regenerating actions, retrievals, or terminal answers. It preserves the ordering (0.364 versus 0.110), with a paired difference of 0.254. The gap is positive in every dataset and under both scorers (Table 1).

Calculator. We evaluate 64 programmatically generated problems requiring five to eight operations, six candidate expressions, safe AST execution, and four final-answer draws. Of 384 candidates, 324 are valid across 59 questions. Terminal utility varies in 44 action sets. Total gain achieves cosine 0.643, residual credit 0.331, and their paired difference is 0.312.

Programmatic expression correctness gives a sharper interpretation. On 26 informative sets, both residual credit and total gain are highly aligned with correctness (0.934 and 0.992). Yet their alignment with terminal utility separates to 0.331 and 0.643. Thus recognizing a correct tool call does not by itself determine which reward best guides complete-agent success.

Table 1: Mean local-gradient cosine by retrieval dataset and scorer. $T - R$ is the paired difference between total and residual credit.

Scorer	Dataset	n	Total	Residual	$T - R$
Qwen3.5-9B	Hotpot-style	30	.360	.090	.270
	2Wiki	36	.380	.120	.260
	MuSiQue	30	.560	.130	.430
	Pooled	96	**.430**	**.114**	**.316**
Qwen2.5-7B	Hotpot-style	30	.320	.070	.250
	2Wiki	36	.330	.110	.220
	MuSiQue	30	.450	.150	.300
	Pooled	96	**.364**	**.110**	**.254**

Table 2: Reward–utility gradient cosine. Candidate-space results include terminal utility in both environments and expression correctness (Program) in the calculator. Parameter space uses the trainable actor gradients for terminal utility.

		Candidate space		
Environment	Utility	Total	Residual	Action-side
Retrieval	Terminal	.430	.114	–
Calculator	Terminal	.643	.331	.440
Calculator	Program	.992	.934	.709

	Parameter space	
Environment	Total	Residual
Retrieval	.33	.09
Calculator	.51	.26

Trainable-parameter gradients. For each question, we compute centered score-function gradients on the parameters updated by the corresponding RL run:

$$g_r^\theta = \frac{1}{m}\sum_i (r_i - \bar{r})\nabla_\theta \log \pi_\theta(a_i \mid s), \qquad g_u^\theta = \frac{1}{m}\sum_i (u_i - \bar{u})\nabla_\theta \log \pi_\theta(a_i \mid s). \tag{12}$$

Here $\log \pi_\theta(a_i \mid s)$ uses the same action-token mask and sequence normalization as the RL objective, evaluated at the checkpoint that generated the candidates. The corresponding cosine gives 0.33 versus 0.09 for retrieval and 0.51 versus 0.26 for calculator use, giving gaps of 0.24 and 0.25 (Table 2). Candidate- and parameter-space signs agree on 84% and 86% of nonzero comparisons. Both parameterizations preserve the total-over-residual preference.

The two spaces use different geometry. If $J_i = \nabla_\theta \log \pi_\theta(a_i \mid s)$ and $D = \mathrm{diag}(p)$, then

$$\langle g_r^\theta, g_u^\theta \rangle = \tilde{r}^\top D J J^\top D \tilde{u}. \tag{13}$$

Candidate-logit cosine treats candidate coordinates equally, whereas parameter-space cosine weights them through the actor's score-feature kernel $JJ^\top$. Their agreement here shows that the observed ordering survives the parameter sharing used by RL.

3.2 What the intervention isolates

Retrieval uses a fixed no-results marker, while calculator execution uses a fixed no-result message. Both counterfactuals retain the action; whole-turn deletion removes it. Calculator execution is deterministic given an expression, so expression correctness isolates action formulation from downstream generation. Both rewards track this tool-level property, but total gain is more aligned with terminal exact match. The gap therefore does not require a stochastic tool output.

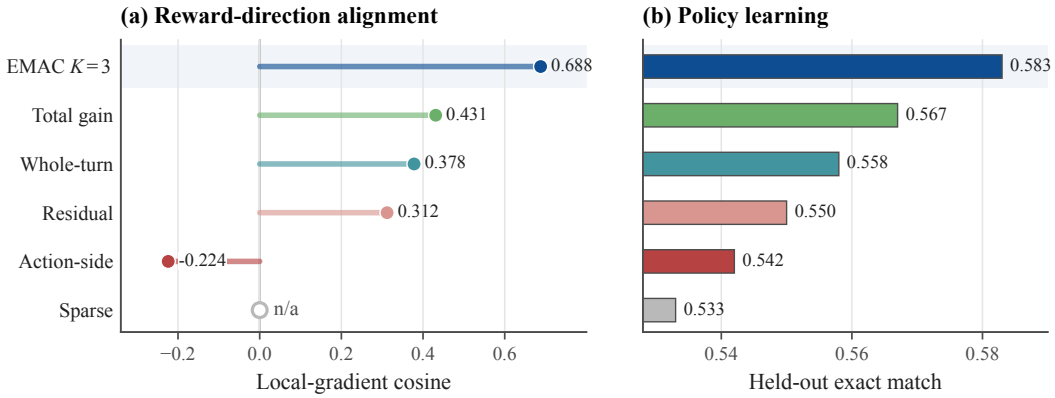

Figure 3: Audit-to-RL test on a 3B actor. (a) Base-policy categorical-gradient cosine; the zero-gradient sparse reward has undefined cosine and is shown at zero only as an open-circle reference. (b) Mean held-out exact match after 75 updates under the same evaluation. Colors identify rewards across panels. The reported means for the five rewards with defined audit gradients have the same order.

4 From Local Audit to Policy Learning

Candidate- and parameter-space agreement establishes the local diagnosis. We next test whether the same ordering appears after policy learning. We compare audit and RL outcomes on a Qwen2.5-3B actor using 48 audit questions, 150 training questions, and 120 disjoint held-out questions. The comparison includes six rewards, three tool-output draws, and two terminal continuations per output. Training retrieval is corrupted with probability 0.5. We evaluate clean held-out exact match after 75 updates and use the same exact-match evaluation for both measurements. The 3B runs follow the 1.5B training recipe detailed in Appendix B. They use four on-policy rollouts, a turn-position group baseline, AdamW at 5×10^{-6}, gradient clipping at 1.0, dense-reward weight 0.1, and a per-token KL penalty of 5.0.

We compare sparse terminal credit, total gain R_{tot}, observation residual R_{res}, action-side credit R_{act}, whole-turn deletion $B(s, a, o) - B(s, \text{deleted turn})$, and EMAC $K = 3$ (Eq. 15). Dense values are clipped at 3. Sparse credit has no dense gradient, so it remains an RL baseline but has no audit cosine.

For the five dense rewards, audit and held-out RL means have the same ordering. EMAC has the strongest audit alignment and reaches 0.583 held-out EM. Total gain reaches 0.567, a 1.6-point difference. Action-side credit has cosine -0.224 and reaches 0.542. Sparse terminal reward reaches 0.533 but has no dense-gradient audit score.

Whole-turn deletion and residual credit yield audit cosines of 0.378 and 0.312, with held-out EM of 0.558 and 0.550.

4.1 Action-side credit in policy learning

We compare total belief gain and residual document credit as dense rewards. With an evolving policy judge, total gain reaches 0.337 held-out EM and residual credit reaches 0.187. With a fixed judge, the values are 0.283 and 0.180. Recombining separately clipped residual and action-side components reaches 0.297 and 0.313.

Under 50% training-time retrieval corruption, total gain changes from 0.337 to 0.253 held-out EM. Residual credit changes from 0.187 to 0.217. Corruption introduces irrelevant or conflicting passages, which can alter an otherwise correct response Shi et al. (2023); Wang et al. (2025a).

The total–residual gap persists with a fixed judge, while final search use stays near one call per question. Recombining separately clipped components reaches 0.313. Because clipping is nonlinear, this condition does not reconstruct clipped total gain; it tests whether restoring action-side information improves on residual credit.

4.2 TRACING THE EFFECT TO ACTION-SIDE CREDIT

We vary the retained action-side term with $R_\lambda = R_{\text{res}} + \lambda R_{\text{act}}$. Local alignment increases smoothly as action-side credit returns (Figure 4(a)). Clean held-out EM follows the same direction. Under corrupted training, the best result occurs at $\lambda = 0.75$, which balances action information against sensitivity to noisy tool outputs.

The endpoints recover the two original rewards: $R_0 = R_{\text{res}}$ and $R_1 = R_{\text{tot}}$. At intermediate values,

$$R_\lambda = B(s, a, o) - (1 - \lambda)B(s, a, o_\emptyset) - \lambda B_0(s). \tag{14}$$

The sweep changes only the retained action-side belief. Audit alignment increases monotonically with this component. Clean and corrupted training prefer different weights, showing that its value depends on observation quality.

4.3 THE EMAC FIX: RETAIN THE ACTION, AVERAGE THE ENVIRONMENT

The interpolation identifies two requirements for a useful repair: preserve the action-dependent signal removed by residualization, and avoid tying that signal to one possibly noisy tool realization. Observation residualization cannot satisfy both: it removes the action-dependent term along with the unwanted sensitivity. We instead define the action reward by marginalizing the environment outcome while keeping the chosen action fixed:

$$R_K(s, a) = \frac{1}{K} \sum_{k=1}^{K} [B(s, a, o_k) - B_0(s)], \qquad o_k \sim P(\cdot \mid s, a), \tag{15}$$

which we call environment-marginalized action credit (EMAC). The defining object is the action-conditioned expectation, not averaging alone. Its finite-sample estimator satisfies $R_K = R_{\text{act}} + K^{-1} \sum_k R_{\text{res}}(s, a, o_k)$: the action-side term is preserved exactly while only observation-dependent variation is averaged. At $K = 1$, EMAC recovers total belief gain; larger K increasingly targets the environment-marginalized value of the action.

Output variation. We compare fixed and variable retrieval outputs using the same Wikipedia corpus and data split. Deterministic retrieval always returns the BM25 top three passages. Stochastic retrieval samples three passages without replacement from the top 20, using a softmax over BM25 scores. Table 3 reports held-out exact match, reward–utility alignment, and observation variation for total gain, residual credit, and EMAC with $K = 3$.

Table 3: EMAC benefits from averaging when retrieval outputs vary and reduces to total gain when they do not. Observation variance is zero for deterministic retrieval and is normalized to total gain under stochastic retrieval.

Retrieval	Reward	Held-out EM	Audit cosine	Obs. variance
Deterministic	Total gain	.337	.43	.00
	Residual credit	.193	.15	.00
	EMAC ($K = 3$)	.337	.43	.00
Stochastic	Total gain	.263	.39	1.00
	Residual credit	.218	.17	.74
	EMAC ($K = 3$)	**.312**	**.60**	**.35**

When retrieval is deterministic, EMAC and total gain coincide at 0.337 held-out EM and 0.43 audit cosine. With stochastic retrieval, EMAC reaches 0.312 EM, compared with 0.263 for total gain and 0.218 for residual credit. Its audit cosine rises to 0.60, while normalized observation variance falls from 1.00 to 0.35. The 4.9-point gain appears only when the output varies, matching the role of environment averaging in EMAC.

Under 50% corrupted training retrieval, we compare $K \in \{1, 2, 3, 5\}$ at the same reward-scoring cost, measured by cumulative teacher-forced verifier targets. Each run extends to 150 updates. The matched comparison uses the latest checkpoint whose target count does not exceed the corresponding

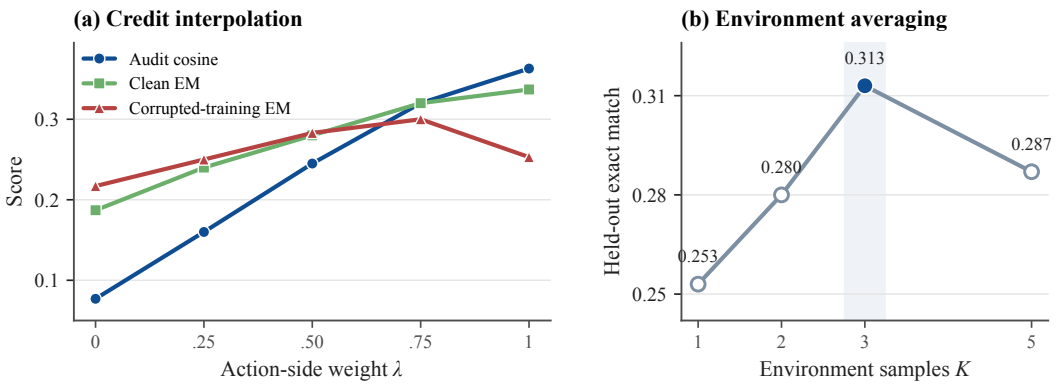

Figure 4: **Reward-content and compute tradeoffs.** Panel (a) shows how audit cosine and held-out EM change as action-side credit is restored. Panel (b) reports the complete EMAC sweep under equal reward-scoring budgets. $K = 3$ gives the best held-out accuracy, while $K = 5$ trades policy updates for additional environment samples.

$K = 1$ budget. $K = 3$ reaches held-out EM 0.313. The results for $K = 1$, $K = 2$, and $K = 5$ are 0.253, 0.280, and 0.287. Thus $K = 3$ improves exact match by 6.0 points over total gain at equal scoring cost. The $K = 5$ result captures the sample–update tradeoff: more environment samples leave fewer policy updates. Figure 4 combines both matched-budget analyses.

Effect of averaging. Conditioned on (s, a), independent environment draws satisfy

$$\mathbb{E}[R_K \mid s, a] = \mathbb{E}_o[B(s, a, o)] - B_0(s), \tag{16}$$

$$\mathrm{Var}(R_K \mid s, a) = K^{-1}\,\mathrm{Var}_o(B(s, a, o)). \tag{17}$$

Thus K reduces observation variance without residualizing away action preference. This distinguishes EMAC from using an average merely as a generic variance-reduction device: the decomposition specifies which term must remain action dependent and which uncertainty is marginalized. Under a fixed compute budget, K also controls the allocation between environment samples and policy updates; the matched-budget sweep selects $K = 3$ here.

Compute accounting. Total gain uses one teacher-forced verifier target and EMAC-K uses K, since

$$R_{\mathrm{act}} + K^{-1} \sum_k R_{\mathrm{res}}(s, a, o_k) = K^{-1} \sum_k B(s, a, o_k) - B_0(s). \tag{18}$$

We select the latest checkpoint within the $K = 1$ cumulative-target budget, so larger K receives fewer updates. Appendix B gives the shared RL setup.

5 Related Work

Tool agents learn search and browsing from outcome feedback Yao et al. (2023); Nakano et al. (2021); Asai et al. (2023); Jin et al. (2025); Qi et al. (2024). We study a complementary question: which current actions an intermediate reward reinforces.

Information-gain rewards turn changes in answer belief into process supervision. Related methods condition the gain on context, rollout structure, memory, or retrieved documents Wang et al. (2026); Auzina et al. (2026); Dou (2026); Hu et al. (2026); Zhang et al. (2026); Han et al. (2026); Wang et al. (2025b); TIPS uses a whole-turn potential difference Xie et al. (2026); Ng et al. (1999). These choices differ in both what they attribute and which action they reinforce.

Other credit-assignment methods use retrospective critics, error localization, turn deletion, sibling rollouts, environment feedback, or Shapley allocation Tao et al. (2026); Zhang et al. (2025); Liang et al. (2026); Lu et al. (2026); Huang et al. (2026); Liu et al. (2026); Zhu et al. (2026); Meng (2026); Meng & Chen (2026); Huo et al. (2026); Li et al. (2026). Our finite-action view links their estimands

to action-dependent baselines Wu et al. (2018); Tucker et al. (2018) and policy-invariant shaping Ng et al. (1999). The new question is not whether the counterfactual isolates an observation. It is whether that estimand induces the intended preference over structured actions.

6 Conclusion

Counterfactual rewards specify an estimand. Removing an observation while keeping its action measures the output's contribution, not which action the policy should repeat. Our finite-action audit exposes this difference in the update direction, and EMAC supplies the corresponding repair: retain action-side credit and marginalize output variation. A useful counterfactual must therefore answer two questions: what does it isolate, and which action does it reinforce? EMAC matches total gain for fixed outputs and improves held-out exact match by 4.9 points when relevant retrieval outputs vary.

Ethics Statement

This work studies reward design for tool-using language agents using established question-answering benchmarks, a Wikipedia corpus, and programmatically generated calculator problems. It involves no human subjects and collects no new personal or sensitive data. The experiments are confined to research evaluation of model-generated tool actions.

AI Use Statement

Generative AI tools provided feedback on experimental design and result interpretation, and assisted with manuscript structure, prose editing, literature organization, and figure refinement. The authors made the scientific decisions, ran the experiments, and manually verified the derivations, citations, figures, and numerical claims. The authors reviewed all AI-assisted work and take responsibility for the final content.

REFERENCES

Akari Asai, Zequi Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*, 2023.

Ilze Amanda Auzina, Joschka Strüber, Sergio Hernández-Gutiérrez, Shashwat Goel, Ameya Prabhu, and Matthias Bethge. Intrinsic credit assignment for long horizon interaction (δbelief-rl). *arXiv:2602.12342*, 2026.

H. Dou. CIGPO: Contextual information-gain policy optimization for multi-turn evidence-reading agents. *arXiv:2607.16244*, 2026.

T. Han, Y. Li, et al. InfoMem: Training long-context memory agents with answer-conditioned information gain. *arXiv:2606.03329*, 2026.

X. Ho, A.-K. D. Nguyen, S. Sugawara, and A. Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *COLING*, 2020.

S. Hu, Y. Dai, et al. InfoReasoner: Agentic reasoning with retrieval via synthetic semantic information gain reward. *arXiv:2602.00845*, 2026.

Yibin Huang, Bin Xu, Hailong Cao, and Conghui Zhu. BiCAA: Bidirectional credit assignment for search-augmented agents. *arXiv:2608.01321*, 2026.

Yifu Huo, Shunjie Xing, Chenglong Wang, Peinan Feng, Qiaozhi He, Yan Ding, Anxiang Ma, Yuxin Gao, Tongran Liu, Tong Xiao, and Jingbo Zhu. Learning from environmental feedback: Credit assignment across multiple timescales for agentic rl. *arXiv preprint arXiv:2608.08255*, 2026. URL https://arxiv.org/abs/2608.08255.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O. 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*, 2025.

Mingxuan Li, Kaizhan Lee, and Elias Bareinboim. Counterfactual shapley credit assignment. *arXiv preprint arXiv:2607.16999*, 2026. URL https://arxiv.org/abs/2607.16999.

Qiao Liang, Yuke Zhu, Chao Ge, Lei Yang, Ying Shen, Bo Zheng, and Sheng Guo. Learning from the irrecoverable: Error-localized policy optimization for tool-integrated llm reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 11008–11028, 2026. doi: 10.18653/v1/2026.acl-long.504. URL https://aclanthology.org/2026.acl-long.504/.

Y. Liu, Y. Feng, et al. Beyond trajectory rewards: Step-level credit assignment for agentic search via graph modeling. *arXiv:2605.29697*, 2026.

Yijun Lu, Rui Ye, Jiajun Wang, Yuwen Du, Tian Jin, Songhua Liu, and Siheng Chen. ABSeeker: Long-horizon search agents via answer-backtracked credit assignment. *arXiv:2608.05102*, 2026.

R. Meng. Policy-conditioned counterfactual credit for verifiable RL of long-horizon language agents (CVT-RL). *arXiv:2606.05263*, 2026.

Zibin Meng and Kani Chen. CRAFT: Counterfactual credit assignment from free sibling rollouts for self-distilled agentic RL. *arXiv preprint arXiv:2606.29476*, 2026. URL https://arxiv.org/abs/2606.29476.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. WebGPT: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 278–287, 1999. URL https://ai.stanford.edu/~ang/papers/shaping-icml99.pdf.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Wenyi Zhao, Yu Yang, Xinyue Yang, Jiadai Sun, Shuntian Yao, Tianjie Zhang, Wei Xu, Jie Tang, and Yuxiao Dong. WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning. *arXiv preprint arXiv:2411.02337*, 2024.

Qwen Team. Qwen3.5-9B model card. Hugging Face model repository, 2026. URL https://huggingface.co/Qwen/Qwen3.5-9B.

F. Shi et al. Large language models can be easily distracted by irrelevant context. In *ICML*, 2023. arXiv:2302.00093.

Y. Tao, Z. Peng, et al. TRACE: Turn-level reward assignment via credit estimation for long-horizon agents. *arXiv:2607.13988*, 2026.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

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

George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard Turner, Zoubin Ghahramani, and Sergey Levine. The mirage of action-dependent baselines in reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5015–5024, 2018. URL https://proceedings.mlr.press/v80/tucker18a.html.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.14967.

Han Wang, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. Retrieval-augmented generation with conflicting evidence (ramdocs, madam-rag). *arXiv:2504.13079*, 2025a.

Zihan Wang, Zihan Liang, Zhou Shao, Yufei Ma, Huangyu Dai, Ben Chen, Lingtao Mao, Chenyi Lei, Yuqing Ding, and Han Li. InfoGain-RAG: Document information gain-based reranking and filtering. *arXiv:2509.12765*, 2025b.

Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M. Bayen, Sham Kakade, Igor Mordatch, and Pieter Abbeel. Variance reduction for policy gradient with action-dependent factorized baselines. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1803.07246.

Yutao Xie, Nathaniel Thomas, Nicklas Hansen, Yang Fu, Li Erran Li, and Xiaolong Wang. TIPS: Turn-level information-potential reward shaping for search-augmented LLMs. *arXiv:2603.22293*, 2026.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W 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 (EMNLP)*, pp. 2369–2380, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Y. Zhang, F. Xu, et al. IGRPO: Information gain-based rollout policy optimization for multi-turn LLM agents. *arXiv:2607.06223*, 2026.

Yaocheng Zhang, Haohuan Huang, Zijun Song, Yuanheng Zhu, Qichao Zhang, Zijie Zhao, and Dongbin Zhao. CriticSearch: Fine-grained credit assignment for search agents via a retrospective critic. *arXiv preprint arXiv:2511.12159*, 2025. URL https://arxiv.org/abs/2511.12159.

Qiang Zhu, Jiajun Wu, and Longyi Wang. LOTAPO: Leave-one-turn attribution for self-generated process rewards in multi-turn search reasoning. *arXiv:2607.13501*, 2026.

A PROMPTS AND SCORING TARGETS

Retrieval system prompt.

```
You are a search assistant with access to a Wikipedia
corpus.  Use <search>query</search> to search or
<answer>answer</answer> to answer.  Final answers are short:
yes/no, a name, or a few words.
```

The retrieval evaluation requires exactly one search before answering. After retrieval, the terminal instruction is: `Give your final short answer now. Do not search again. Use <answer>...</answer>.`

Calculator system prompt.

```
You are an assistant with a calculator tool.  Use
<calculate>arithmetic expression</calculate> once, then
answer after the tool result.  Final answers must be a
single integer in <answer>...</answer>.
```

The calculator evaluation asks for one short reasoning sentence followed by one calculator call. The terminal instruction is: `Give the final integer now. Do not call the calculator again. Use <answer>...</answer>.`

Teacher-forced score. For gold tokens $y^* = (w_1, \ldots, w_K)$ and rendered context h, every belief is $B(h) = \sum_k \log P_M(w_k \mid h, w_{<k})$. Scoring uses no answer sampling and does not length-normalize. All comparisons for a question share the same gold string. Audit utility comes from generated terminal answers, so belief and outcome do not reuse a stochastic sample.

B TRAINING SETUP

The 1.5B experiments use Qwen2.5-1.5B-Instruct with four on-policy trajectories and at most four interaction turns. Return-to-go is centered and scaled across trajectories at the same turn position. We use AdamW with learning rate 5×10^{-6} and gradient clipping at 1.0. Each trajectory receives terminal exact match plus 0.1 times the dense reward clipped to $[-3, 3]$. A frozen reference supplies a per-token KL penalty of 5.0. Each group produces one optimizer step, with no rollout reuse.

These experiments use 240 training and 60 evaluation items. All conditions use the same split and are evaluated greedily every 25 updates; full-budget results use update 150. Training corruption replaces the retrieved block with three non-gold paragraphs with probability 0.5 and is disabled at evaluation. The 3B comparison in Section 4 uses 150 training and 120 held-out items and reports update 75.