BEYOND LATENT SIMILARITY: CREDIT ASSIGNMENT WITH STATE TRANSITIONS FOR LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

In agentic RL, feedback based on final episode outcomes does not directly distinguish the contributions of individual steps. Recent methods derive local credit from explicit interaction records or policy hidden states. For methods that share returns within latent neighborhoods formed by hidden-state similarity, pooling records by action can assign identical credit to steps with different observed transitions. To address this limitation, we introduce `CAST`, a method for step-level credit assignment motivated by state abstraction theory. `CAST` forms candidate neighborhoods using policy hidden-state similarity and selects references that match the target step in both action and observed transition, while retaining records of other actions for local comparison. It estimates step-level credit from the retained records without additional environment interaction or a learned value model. Experiments with Qwen3.5-4B and Qwen3.5-9B across four interactive environments show that `CAST` outperforms GiGPO and BiPACE in all eight settings, with success rate gains of up to 8.9 and 22.0 percentage points, respectively. Additional experiments extend these gains to Llama in two environments. Evaluations on 11 general capability benchmarks further show that both Qwen models retain overall performance after training.

1 INTRODUCTION

Agentic reinforcement learning (RL) trains large language models (LLMs) to make sequential decisions through interaction with an environment (Zhou et al., 2024; Qi et al., 2025). Within an episode, an agent may advance the task, make ineffective attempts, and correct earlier mistakes. When training relies on the final episode outcome, however, the feedback does not directly distinguish the contributions of individual steps (Zhang et al., 2025; Xie et al., 2026). This motivates step-level credit assignment to provide local learning signals for intermediate decisions.

Recent work derives local learning signals from data collected during training. One line of work uses explicit interaction records, including observations, actions, and interaction histories, to relate steps and estimate credit (Feng et al., 2025; He et al., 2026; Li et al., 2026; Liu et al., 2026). Another uses comparisons of policy hidden states to estimate or reweight advantages in single-turn reasoning tasks (Chen et al., 2026b;c; Lafuente-Mercado, 2026). For interactive agents, records from the same task can be grouped by hidden-state similarity into *latent neighborhoods*, allowing return information to be shared across different observations. Within a neighborhood, one approach compares the mean return-to-go of records taking the same action with that of all records and derives step-level credit from their difference (Wang et al., 2026a).

However, the same action can produce different observed transitions at steps with similar hidden states. Figure 1 shows two ScienceWorld records from the same task and latent neighborhood. The action `go to hallway` is blocked when the door is closed but moves the agent to the hallway when it is open. Under this approach, the two records share both return means and therefore receive identical step-level credit. We observe such transition differences among records with the same action within latent neighborhoods in four interactive environments (Table 1). These observations raise the question: *can recorded transitions guide reference selection to improve step-level credit estimation within latent neighborhoods?*

In Markov decision processes (MDPs) (Puterman, 1994), classical state abstraction theory provides sufficient conditions on rewards and action-conditioned transition distributions under which state

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.

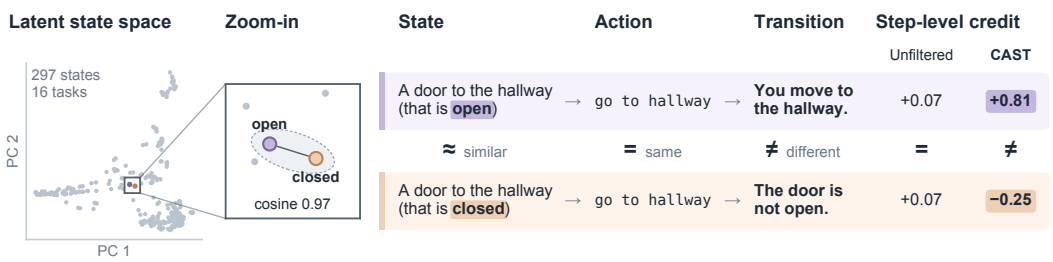

Figure 1: Similar hidden states, different observed transitions in ScienceWorld. The two records share a task, latent neighborhood, and action but differ in their observed transitions. Without transition filtering, both receive identical step-level credit; CAST assigns distinct values. Both estimators use the same recorded batch, and only the step-level component is shown. Principal component analysis (PCA) is used for visualization; cosine similarity is computed in the original hidden-state space.

aggregation preserves optimal values (Givan et al., 2003; Ferns et al., 2004). Motivated by this perspective, we propose Credit Assignment with State Transitions (CAST), which uses policy hidden-state similarity to form candidate neighborhoods and recorded transitions to select references within them. Each observed transition is summarized by a signature comprising the immediate reward, termination status, and normalized successor observation. Within each target step's neighborhood, CAST excludes records with the same action but a different signature and retains all records of other actions for local comparison. It then recomputes both return means using the retained records to estimate step-level credit. This credit is combined with the episode-level signal for policy optimization. The method uses records already collected during training and requires no additional environment interaction or learned value model.

We train Qwen3.5-4B and Qwen3.5-9B with CAST separately in each of four interactive environments. CAST outperforms GiGPO and BiPACE on all eight model–environment combinations, improving the average score across the four environments by up to 5.0 and 11.5 points, respectively. Additional experiments with Llama-3.2-3B-Instruct on BabyAI and WebShop show gains over both baselines, extending these results to another model family. On ALFWorld, Qwen3.5-9B trained with CAST achieves an 86.5% success rate, compared with 83.5% for random reference selection with matched reference counts, supporting the use of transition information beyond simply reducing the number of references. Both Qwen models also maintain or improve their average accuracy across 11 general capability benchmarks after training.

Our main contributions are as follows:

- *Conceptually*, we frame return sharing within latent neighborhoods as a reference selection problem, highlighting observed transitions as additional evidence beyond hidden-state similarity and action matching.
- *Methodologically*, we propose CAST, which filters references with the same action by transition signature and retains records of other actions for local comparison. It estimates credit without additional environment interaction or a learned value model.
- *Empirically*, we demonstrate consistent gains across interactive environments and model families. Ablation studies support the reference selection design, while evaluations on general capability benchmarks show retained overall performance.

2 Related Work

Reinforcement learning for LLM agents. LLM agent policies can be optimized using interaction rewards or preferences derived from episode outcomes. ArCHer (Zhou et al., 2024) combines value learning across interaction steps with token-level policy optimization to handle delayed rewards. WebRL (Qi et al., 2025) trains web agents from outcome rewards using value-based advantage estimates and KL-constrained updates within a self-evolving curriculum. RAGEN (Wang et al., 2025) introduces StarPO to optimize complete interaction episodes and studies the stability of multi-turn RL. Search-R1 (Jin et al., 2025) and ReSearch (Chen et al., 2025) use final-answer rewards to train

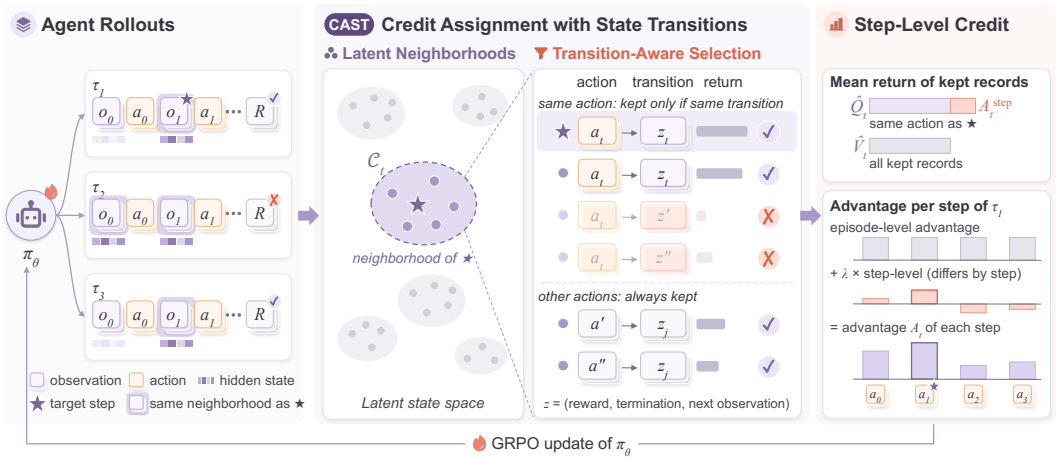

Figure 2: Overview of CAST. Within each task, policy hidden states group interaction records into latent neighborhoods. For each target step, records with the same action are retained only if their transition signatures match the target signature, while all records of other actions in the neighborhood are retained for local comparison. Step-level credit is estimated from the retained records and combined with the episode-level advantage for policy optimization.

agents that interleave reasoning and search. For offline preference optimization, DEPO (Chen et al., 2026a) uses labels derived from episode outcomes and adds an efficiency bonus based on tokens per step and total interaction steps. Although episode-level feedback supports policy learning, it does not directly specify the contribution of individual decisions. This motivates step-level credit assignment to construct local learning signals for policy optimization.

Step-level credit assignment. Methods based on explicit interaction records derive local supervision through continuation sampling, process reward modeling, or reward shaping (Xiong et al., 2024; Choudhury, 2025; Zhang et al., 2025; Xie et al., 2026; Zhang et al., 2026). Other methods use observations, histories, and transitions to form comparison groups or organize records into graphs and trees for credit estimation (Feng et al., 2025; He et al., 2026; Li et al., 2026; Liu et al., 2026; Wang et al., 2026b; Ren et al., 2026). Beyond explicit records, hidden-state methods estimate state values or reweight token-level advantages in reasoning tasks (Chen et al., 2026c;b; Lafuente-Mercado, 2026). For interactive agents, BiPACE (Wang et al., 2026a) groups steps by policy hidden-state similarity. However, steps with similar hidden states can exhibit different observed transitions under the same action.

3 METHOD

CAST estimates step-level credit using policy hidden states to form latent neighborhoods and recorded transitions to select reference records within them (Figure 2). We first define the interaction setting and credit estimation problem (§3.1). We then motivate transition-guided reference selection through empirical analysis and state-abstraction theory (§3.2). Finally, we present the credit estimator and its use in policy optimization (§3.3).

3.1 PROBLEM SETUP

We consider an LLM agent interacting with an environment whose underlying state s_t follows Markov dynamics (Puterman, 1994). At step t, the policy input c_t contains the task instruction, current observation o_t, and available interaction history. The policy samples an environment action $a_t \sim \pi_\theta(\cdot \mid c_t)$. Executing a_t induces a transition to s_{t+1}, yielding an immediate reward r_t, a termination indicator d_t, and a successor observation o_{t+1}. An episode τ consists of T interaction steps and ends upon environment termination or at a prescribed horizon.

Each training batch contains multiple episodes per task. The undiscounted episode reward $R(\tau) = \sum_{t=0}^{T-1} r_t$ is used to compute the episode-level advantage. For step t, the discounted return-to-go

is $G_t = \sum_{k=t}^{T-1} \gamma^{k-t} r_k$, where $\gamma \in [0, 1]$ is the discount factor.[1] We study how to select reference records from the same task and aggregate their return-to-go values to estimate step-level credit.

3.2 Revisiting Credit Estimation in Latent Neighborhoods

Local value estimates can be constructed by aggregating returns across records matched by observations or learned representations (Pritzel et al., 2017; Feng et al., 2025). Here, we group records from the same task by policy hidden-state similarity (Chen et al., 2026c; Wang et al., 2026a), forming *latent neighborhoods*. For a target step, the unfiltered estimator uses the mean return-to-go over same-action records in its neighborhood as the local action value estimate and the mean over all neighborhood records as the local baseline. Their difference provides the step-level credit.

Table 1: Transition differences before filtering over the first 25 training updates of Qwen3.5-9B in each environment. Candidate pairs share a task, latent neighborhood, and executed action. Affected records belong to at least one differing pair. Cosine distance thresholds are 0.08 for WebShop and 0.1 otherwise.

Environment	Records	Candidate pairs	Differing pairs (%)	Affected records (%)
ALFWorld	105,872	509,264	18.9	37.8
BabyAI	19,152	167,333	45.7	60.0
ScienceWorld	137,640	1,739,985	39.2	48.0
WebShop	47,248	134,402	24.5	22.0

Empirically, same-action records within a latent neighborhood can differ in their observed transitions. We quantify these differences over the first 25 training updates of Qwen3.5-9B in four environments (Table 1). Across these environments, 18.9%–45.7% of same-action candidate pairs differ in immediate reward, termination status, or normalized successor observation. However, the unfiltered estimator does not use these differences in reference selection. Same-action records within each neighborhood therefore share the same local action value estimate and local baseline, yielding identical step-level credit (Figure 1).

Theoretically, we examine reference selection through the value-preservation conditions of state abstraction. For a fixed task, optimal action values in the underlying MDP satisfy the Bellman relation (Puterman, 1994)

$$Q_t^*(s, a) = \bar{r}(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[V_{t+1}^*(s') \right], \tag{1}$$

where $\bar{r}(s, a)$ is the expected immediate reward, P is the transition distribution, and V_{t+1}^* is the optimal state value at the next step. Terminal states are modeled as absorbing states with zero subsequent reward, and continuation values are zero at the prescribed horizon. Consider a partition of the MDP state space into groups. At a fixed remaining horizon, this aggregation preserves optimal values if any two states $s, \tilde{s}$ in the same group satisfy the bisimulation conditions for every action a and group B (Givan et al., 2003):

$$\bar{r}(s, a) = \bar{r}(\tilde{s}, a), \qquad P(B \mid s, a) = P(B \mid \tilde{s}, a), \tag{2}$$

where $P(B \mid s, a)$ is the probability that the successor state belongs to group B. For finite discounted MDPs, suitable bisimulation metrics further bound optimal value differences using reward and transition discrepancies (Ferns et al., 2004). Hidden-state similarity alone does not enforce these conditions. This motivates using recorded transitions as additional evidence for reference selection before return aggregation. Agreement between individual observed transitions, however, does not guarantee value equivalence.

3.3 Credit Assignment With State Transitions

We encode each current observation o_t alone with the policy model and normalize the resulting hidden state to unit length. Within each task in the current batch, we form latent neighborhoods by greedy clustering based on cosine distance. Each record is assigned to the cluster with the nearest centroid if the distance is within a fixed threshold; otherwise, a new cluster is created (Wang et al., 2026a). For a target step t, let $\mathcal{C}_t$ denote the indices of records in its latent neighborhood.

[1] For simplicity, the notation omits the record-level invalid-action adjustments applied before advantage computation. Return aggregation uses adjusted returns, whereas transition signatures retain the original environment rewards (Appendix B.1).

CAST summarizes the observed transition of step t by the signature $z_t = (r_t, d_t, \mathrm{canon}(o_{t+1}))$. The function canon trims leading and trailing whitespace, lowercases the text, and collapses consecutive whitespace. With j indexing candidate records across episodes and time steps, we select the reference set

$$\mathcal{R}_t = \{j \in \mathcal{C}_t : a_j \neq a_t \text{ or } z_j = z_t\}. \tag{3}$$

This restricts same-action return aggregation to records with matching signatures while retaining all other-action records for local comparison. The target record is always retained, ensuring a nonempty same-action reference set.

Among records in $\mathcal{C}_t$ with action a_t, let $n_t(z)$ and $\bar{G}_t(z)$ denote the count and mean return-to-go for transition signature z, respectively. The unfiltered local action value estimate averages across signatures, whereas CAST uses the mean for the target signature:

$$\hat{Q}_t^{\text{unfiltered}} = \sum_z \frac{n_t(z)}{\sum_{z'} n_t(z')} \bar{G}_t(z), \qquad \hat{Q}_t = \bar{G}_t(z_t), \tag{4}$$

where the sums range over signatures observed among these same-action records. We compute the local baseline over all retained records as $\hat{V}_t = |\mathcal{R}_t|^{-1} \sum_{j \in \mathcal{R}_t} G_j$ and obtain the unnormalized step-level credit $\hat{Q}_t - \hat{V}_t$. When standard-deviation normalization is enabled, we divide this difference by the population standard deviation of returns in $\mathcal{R}_t$ plus a small constant $\epsilon > 0$. Denoting the resulting credit by A_t^{step}, we form the combined advantage $A_t = A_t^{\text{episode}} + \lambda A_t^{\text{step}}$. Here, A_t^{episode} is computed from episode rewards within the same task, and λ controls the step-level contribution. We use A_t as the advantage for all response tokens at step t in the GRPO objective (Shao et al., 2024).

4 Experiments

4.1 Experimental setup

Evaluation benchmarks. We evaluate agent performance in four interactive environments. ALFWorld (Shridhar et al., 2021) involves household tasks through text interaction, while BabyAI (Chevalier-Boisvert et al., 2019) tests instruction following in grid worlds. ScienceWorld (Wang et al., 2022) requires agents to carry out scientific tasks, and WebShop (Yao et al., 2022) requires searching for and purchasing products that satisfy user instructions. We report success rates for ALFWorld, BabyAI, and WebShop, and mean task scores for ScienceWorld after clipping negative scores to zero. To assess general capability retention after training, we additionally evaluate 11 benchmarks across four domains. For logic, we use FOLIO (Han et al., 2024) and ProofWriter (Tafjord et al., 2021); for mathematics, we use MMLU-Math (Hendrycks et al., 2021a), SVAMP (Patel et al., 2021), and MATH500, a 500-problem subset of MATH (Hendrycks et al., 2021b). The medical benchmarks are MedMCQA (Pal et al., 2022), LongHealth (Adams et al., 2025), and MedGUIDE (Li et al., 2025); the science benchmarks are GPQA Diamond (Rein et al., 2024), SuperGPQA (Du et al., 2025), and SciCUEval (Yu et al., 2026). These 11 benchmarks are used only for evaluation, with accuracy as the metric.

Models and baselines. Our main policy backbones are Qwen3.5-4B and Qwen3.5-9B (Qwen Team, 2026a). At each scale, we compare CAST with the original model before task-specific RL (Base), GiGPO (Feng et al., 2025), and BiPACE (Wang et al., 2026a). GiGPO derives step-level advantages through anchor-state grouping, whereas BiPACE estimates local credit from action-conditioned return comparisons within latent neighborhoods. Our BiPACE configuration uses the same latent-neighborhood credit estimator as CAST with transition filtering disabled. We also train Llama-3.2-3B-Instruct (Meta, 2024) on BabyAI and WebShop to test CAST on another model family. For additional reference points, we evaluate nine advanced models: Qwen3.6-27B (Qwen Team, 2026b), Qwen3.8-27B (Qwen Team, 2026c), Qwen3.5-35B-A3B, Qwen3.5-122B-A10B (Qwen Team, 2026a), GPT-OSS-120B (OpenAI, 2025), DeepSeek-v4.1-flash (DeepSeek-AI, 2026a), DeepSeek-v4-flash (DeepSeek-AI, 2026b), GLM-5.3-flash (Z.ai, 2026), and Kimi-k2.6 (Moonshot AI, 2026).

Implementation details. We train CAST with GRPO using a learning rate of 5×10^{-7}, discount factor $\gamma = 0.95$, and step-level credit weight $\lambda = 1$. Each batch samples 16 task inputs and collects eight episodes per input, or twelve on WebShop. Current observations are encoded separately with the policy model to construct latent neighborhoods. Additional experimental details are provided in Appendix B.

Table 2: Agent performance across four interactive environments. CAST leads the overall environment scores among the RL methods at both model scales. Average weights the four environments equally. Bold marks the highest overall environment score and Average within each model block.

Model / method	ALFWorld							BabyAI	Science World	WebShop	Average
	Clean	Cool	Heat	Look	Pick	Pick2	All				
Advanced LLMs											
DeepSeek-v4.1-flash	56.8	50.0	60.0	47.4	67.4	44.4	55.0	93.3	29.6	17.0	48.7
DeepSeek-v4-flash	59.5	21.4	48.0	47.4	76.1	44.4	52.0	87.8	25.8	8.0	43.4
GLM-5.3-flash	45.9	64.3	52.0	73.7	89.1	68.9	67.0	93.3	38.7	15.5	53.6
Kimi-k2.6	62.2	39.3	16.0	26.3	76.1	48.9	50.0	88.9	44.2	5.5	47.1
Qwen3.6-27B	42.2	33.6	22.4	33.7	73.0	47.1	45.9	86.0	38.6	11.7	45.5
Qwen3.8-27B	48.6	52.9	21.6	29.5	84.8	56.0	54.0	91.3	44.5	23.5	53.3
Qwen3.5-35B-A3B	47.0	19.3	24.0	33.7	67.4	29.3	39.7	88.9	20.5	15.9	41.3
Qwen3.5-122B-A10B	41.1	22.9	22.4	56.8	70.4	43.6	45.0	80.0	33.0	12.0	42.5
GPT-OSS-120B	12.4	8.6	12.8	26.3	65.7	36.0	30.8	94.2	41.4	1.8	42.1
Qwen3.5-4B											
Base	16.2	0.0	0.0	36.8	52.2	24.4	24.0	39.6	9.7	4.6	19.5
GiGPO	81.1	89.3	56.0	94.7	93.5	84.4	84.0	88.9	48.7	71.0	73.2
BiPACE	89.2	64.3	84.0	94.7	82.6	75.6	81.0	91.1	38.3	56.5	66.7
CAST	78.4	75.0	80.0	94.7	95.7	86.7	**85.5**	**97.8**	**50.6**	**73.0**	**76.7**
Qwen3.5-9B											
Base	15.1	0.0	8.0	33.7	53.5	29.3	25.9	87.1	11.3	8.3	33.2
GiGPO	81.1	85.7	68.0	100.0	89.1	84.4	84.5	90.0	54.5	65.0	73.5
BiPACE	64.9	71.4	0.0	84.2	80.4	71.1	64.5	87.8	44.3	71.5	67.0
CAST	83.8	78.6	76.0	100.0	91.3	88.9	**86.5**	**92.2**	**62.7**	**72.5**	**78.5**

4.2 Main results

Table 2 compares CAST with the RL baselines and advanced LLMs across four interactive environments. We draw the following conclusions:

CAST improves agent performance across environments and model scales. With Qwen3.5-4B and Qwen3.5-9B, CAST achieves average scores of 76.7 and 78.5, exceeding GiGPO by 3.5 and 5.0 points and BiPACE by 10.0 and 11.5 points, respectively. It outperforms both RL baselines on all eight model–environment combinations. The gains extend across different task demands: on ScienceWorld, Qwen3.5-9B scores 62.7 against 54.5 for GiGPO and 44.3 for BiPACE; on BabyAI, Qwen3.5-4B achieves 97.8% success against 88.9% and 91.1%.

A smaller policy trained with CAST can outperform larger alternatives. The 4B model with CAST exceeds both 9B RL baselines in average performance, scoring 76.7 against 73.5 for GiGPO and 67.0 for BiPACE. Its average also exceeds the 53.6 achieved by the highest-scoring advanced LLM in the table. These comparisons show that compact policies trained for environment interaction can achieve competitive task performance.

Figure 3 compares validation performance during training with Qwen3.5-9B on ALFWorld and ScienceWorld. Figure 4 reports success rates with Llama-3.2-3B-Instruct on BabyAI and WebShop. Together, they provide two further findings:

CAST maintains stronger performance during later training. On ALFWorld, BiPACE falls from 91.4% at update 95 to 67.2% at update 175, whereas CAST declines from 90.6% to 84.4% over the same interval. On ScienceWorld, CAST leads both baselines at every validation point from update 55 onward, ending at 63.6 compared with 55.7 for GiGPO and 45.7 for BiPACE. The two environments therefore show sustained advantages over BiPACE, with a smaller late decline on ALFWorld.

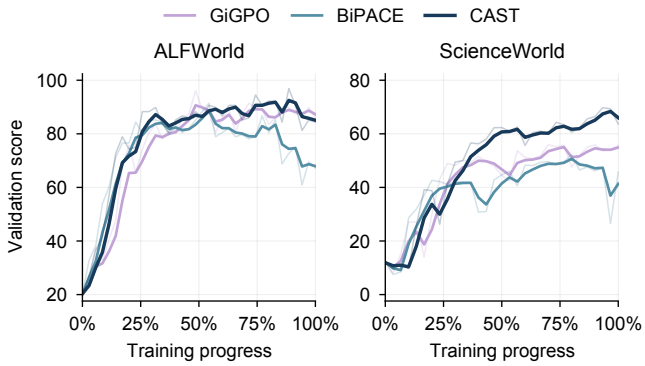

Figure 3: Training dynamics with Qwen3.5-9B. CAST maintains higher late-stage validation performance than BiPACE.

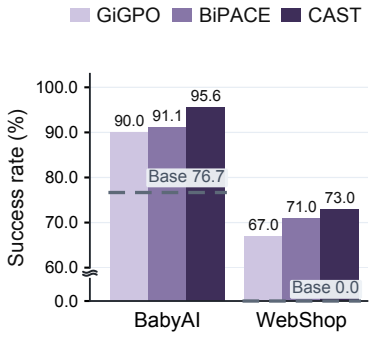

Figure 4: Performance with Llama-3.2-3B-Instruct on BabyAI and WebShop.

Table 3: Tighter clustering leaves transition differences while reducing other-action coverage. Results use unfiltered neighborhoods on the ScienceWorld batches in Figure 5; 0.1 is the default threshold. A differing reference has the same action but a different transition signature. Dark segments (■) mark records with such a reference sharing their current observation; light segments (■) mark the remaining affected records. Matching numbers are percentages of all records. Bars start at zero, with separate scales for the two column groups.

	Records with a differing same-action reference (%)		Records with an other-action reference (%)	
Threshold	Early	Late	Early	Late
0.1 (default)	28.7/19.3	37.6/10.5	95.1	60.7
0.05	28.7/14.1	37.6/5.4	93.3	57.2
0.01	28.7/7.5	37.6/0.8	90.0	51.5
0.005	28.7/5.8	37.6/0.4	89.5	49.3
0.001	28.7/2.3	37.6/0.0	88.8	46.8

The gains extend to another model family. With Llama-3.2-3B-Instruct, CAST achieves 95.6% success on BabyAI and 73.0% on WebShop after 100 training updates. These results exceed BiPACE by 4.5 and 2.0 percentage points and GiGPO by 5.6 and 6.0 points, respectively.

4.3 Analysis of Transition Filtering

We examine early (updates 1–25) and late (updates 126–150) training windows on ScienceWorld with Qwen3.5-9B. Table 3 examines whether tighter latent neighborhoods can remove transition differences. We vary the threshold in offline clustering of the same 50 recorded batches, holding embeddings and record order fixed. The main finding is:

Tighter latent neighborhoods do not replace transition-guided selection. Reducing the threshold from 0.1 to 0.001 removes many differences between records with different current observations. However, 28.7% of early records and 37.6% of late records still have a differing same-action reference with an identical current observation at every scanned threshold. Meanwhile, the fraction of records with other-action references falls from 95.1% to 88.8% early and from 60.7% to 46.8% late. Without other-action references, the local action value and baseline coincide, yielding zero step-level credit. CAST removes same-action references with differing transition signatures while retaining all other-action references at the default threshold. It therefore preserves the available action comparisons while addressing differences that tighter clustering leaves unresolved.

Figure 5 measures reference filtering and the resulting credit changes, while Table 4 reports the retained reference counts. Besides CAST and the unfiltered estimator, we evaluate a variant that omits successor observation (SO) consistency, denoted w/o SO. All estimators are evaluated on the same recorded batches. We find that:

Transition filtering changes credit while preserving reference sharing. Relative to the unfiltered estimator, CAST changes step-level credit for 44.4% of records early in training and 19.3% late in

Table 4: Reference sharing in the windows of Figure 5. Counts exclude the target. Removal rates condition on having other same-action references before filtering.

	Early	Late
Mean same-action references		
Unfiltered	25.3	56.5
w/o SO	24.9	55.8
CAST	15.4	40.3
All same-action references removed by CAST (%)		
	7.9	2.4

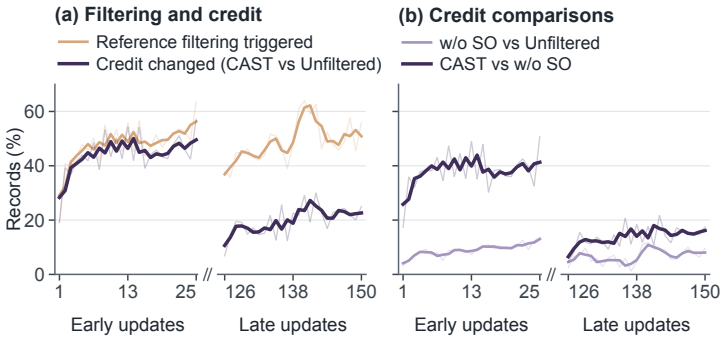

Figure 5: Transition filtering and credit changes on ScienceWorld with Qwen3.5-9B. Early and Late cover updates 1–25 and 126–150.

training. It still retains an average of 15.4 and 40.3 other same-action references, respectively. Among records that initially have such references, only 7.9% and 2.4% lose all of them. Thus, transition filtering modifies local credit without generally reducing estimation to the target record alone.

Successor observations provide distinctions beyond reward and termination. Checking only immediate reward and termination changes credit for 9.1% and 6.5% of records relative to the unfiltered estimator. Adding successor-observation consistency changes credit for 38.3% and 13.6% of records relative to that partial signature. The successor observation therefore affects reference selection even when reward and termination agree. Section 4.4 separately examines whether these credit changes translate into higher final performance.

4.4 Ablation Studies

Table 5: Reference selection on ALFWorld with Qwen3.5-9B. CAST outperforms count-matched random selection.

Method	Success (%)
BiPACE	64.5
Random selection	83.5
CAST	**86.5**

Table 6: Filtering conditions on ScienceWorld with Qwen3.5-9B. Removing immediate-reward consistency causes the larger score decline.

Method	Score
CAST	**62.7**
w/o successor observation	62.3
w/o immediate reward	27.7

Table 5 compares CAST with count-matched random selection on ALFWorld using Qwen3.5-9B. For each target in the control's current batch, random selection retains as many other same-action references as CAST would, sampling uniformly without replacement. The target and all other-action references are retained. We find that:

Transition-guided selection adds value beyond reducing reference counts. Random selection achieves 83.5% success, compared with 64.5% for BiPACE, showing that reference reduction alone can improve performance in this setting. CAST further raises success to 86.5%. This additional 3.0-percentage-point gain supports using transition information to select which references to retain.

Table 6 removes the immediate-reward or successor-observation condition on ScienceWorld with Qwen3.5-9B, retaining termination consistency. Each variant is trained for 150 updates. The results show that:

Immediate-reward consistency has the larger effect on final performance. Removing this condition lowers the score from 62.7 to 27.7, a 35.0-point decline. Removing successor-observation consistency yields 62.3, only 0.4 points below the full method. Although successor observations alter many local credit values (Section 4.3), their additional effect on the final score is limited in this experiment.

4.5 Computational Overhead

Figure 6 reports the training-time shares of observation encoding and the remaining advantage computation, and compares credit-function times with and without transition filtering on cached batches. The main finding is:

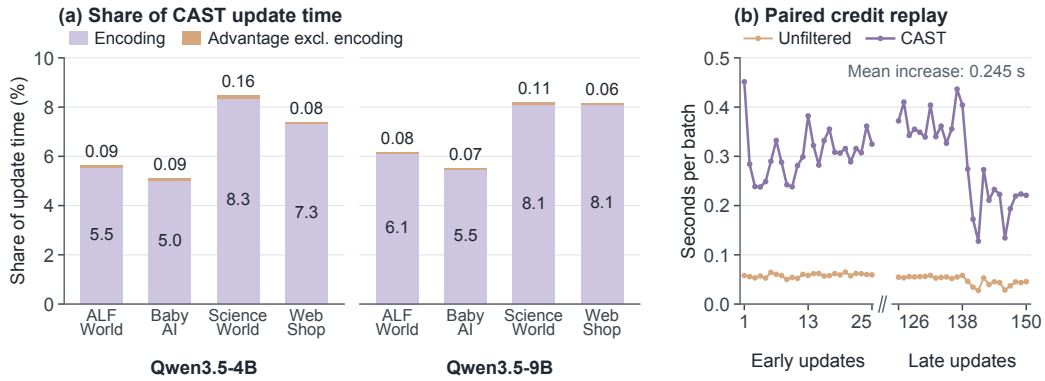

Figure 6: Computational cost of CAST. (a) Observation encoding and the remaining advantage computation as shares of training-update time. Values inside bars denote encoding; values above bars denote the remaining computation, including clustering and filtering. (b) Credit-function times with and without filtering on the same cached ScienceWorld batches with Qwen3.5-9B. Each point represents one batch on a single CPU core; encoding, clustering, and data loading are excluded.

Transition-aware credit computation occupies a small fraction of training time. Across the eight CAST settings, advantage computation excluding observation encoding accounts for 0.06%–0.16% of update time. Observation encoding, which BiPACE also requires, accounts for 5.0%–8.3%. The total advantage-computation cost, including observation encoding, is therefore dominated by encoding. In a separate comparison on the same cached ScienceWorld batches, enabling transition filtering increases credit-function time from 0.053 to 0.299 seconds per batch on one CPU core. The resulting absolute cost remains below 0.3 seconds per batch on average.

4.6 General capability retention

Figure 7 compares Qwen3.5-9B before and after training with CAST on 11 evaluation benchmarks spanning logic, mathematics, medicine, and science. We report mean accuracies over four evaluation runs. We find that:

Training with CAST preserves overall general capability. Average accuracy remains at 71.8% for Qwen3.5-4B and increases from 74.8% to 75.3% for Qwen3.5-9B. The 9B model improves on 10 of the 11 benchmarks, including FOLIO (80.9% to 82.0%) and MedGUIDE (36.9% to 38.3%). The 4B model shows individual trade-offs, with MATH500 increasing from 76.6% to 78.7% and ProofWriter declining from 83.8% to 81.8%.

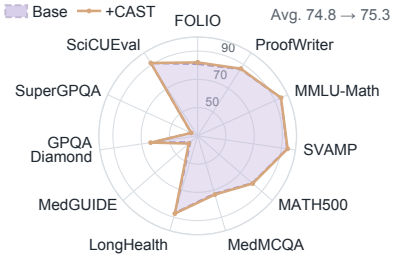

Figure 7: Accuracy (%) of Qwen3.5-9B before and after training with CAST.

5 Conclusion

We present CAST, which uses recorded transitions to guide reference selection for step-level credit assignment within latent neighborhoods. The method estimates credit from existing interaction records without additional environment interaction or a learned value model. Experiments show consistent performance gains across interactive environments and model families, while ablations support the value of transition information beyond reducing reference counts. Both Qwen models retain overall performance on general capability benchmarks. These findings highlight observed transitions as useful evidence for reference selection beyond hidden-state similarity and action matching.

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish the writing of the manuscript. All AI-assisted text was reviewed and revised by the authors, who take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

§3 describes CAST, and §4.1 summarizes the experimental setup. Appendix B provides the full training and evaluation details. The policy backbones, interactive environments, and evaluation benchmarks are all publicly available. We will release the source code upon acceptance of the paper.

REFERENCES

Lisa Adams, Felix Busch, Tianyu Han, Jean-Baptiste Excoffier, Matthieu Ortala, Alexander Löser, Hugo J. W. L. Aerts, Jakob Nikolas Kather, Daniel Truhn, and Keno Bressem. LongHealth: A question answering benchmark with long clinical documents. *Journal of Healthcare Informatics Research*, 2025.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. ReSearch: Learning to reason with search for LLMs via reinforcement learning. In *NeurIPS*, 2025.

Sirui Chen, Mengshi Zhao, Lei Xu, Yuying Zhao, Beier Zhu, Hanwang Zhang, Shengjie Zhao, and Chaochao Lu. DEPO: Dual-efficiency preference optimization for LLM agents. In *AAAI*, 2026a.

Xinzhu Chen, Wei He, Huichuan Fan, Wenzhe Niu, Zhongxiang Sun, Xuanru Wang, Jiuchong Gao, Jinghua Hao, Renqing He, and Weijie Yu. Hidden states know where reasoning diverges: Credit assignment via span-level Wasserstein distance. *arXiv preprint arXiv:2604.23318*, 2026b.

Zizhe Chen, Jiqian Dong, Yizhou Tian, Garry Yang, Yongqiang Chen, Zhitang Chen, and James Cheng. Hista and Numca: Estimate state value effectively for large language model reinforcement learning. In *ICML*, 2026c.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. BabyAI: A platform to study the sample efficiency of grounded language learning. In *ICLR*, 2019.

Sanjiban Choudhury. Process reward models for LLM agents: Practical framework and directions. *arXiv preprint arXiv:2502.10325*, 2025.

DeepSeek-AI. DeepSeek-V4.1-Flash: Pushing the limits of KV cache compression. *arXiv preprint arXiv:2609.19969*, 2026a.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026b.

Xinrun Du, Yifan Yao, Kaijing Ma, et al. SuperGPQA: Scaling LLM evaluation across 285 graduate disciplines. In *NeurIPS Datasets and Benchmarks*, 2025.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *NeurIPS*, 2025.

Norm Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite Markov decision processes. In *UAI*, 2004.

Robert Givan, Thomas Dean, and Matthew Greig. Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence*, 2003.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alex Wardle-Solano, Hannah Szabó, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, Ansong Ni, Linyong Nan, Jungo Kasai, Tao Yu, Rui Zhang, Alexander R. Fabbri, Wojciech Kryściński, Semih Yavuz, Ye Liu, Xi Victoria Lin, Shafiq Joty, Yingbo Zhou, Caiming Xiong, Rex Ying, Arman Cohan, and Dragomir Radev. FOLIO: Natural language reasoning with first-order logic. In *EMNLP*, 2024.

Shuo He, Lang Feng, Qi Wei, Xin Cheng, Lei Feng, and Bo An. Hierarchy-of-groups policy optimization for long-horizon agentic tasks. In *ICLR*, 2026.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *ICLR*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks*, 2021b.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Ö. Arık, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. In *COLM*, 2025.

Rodney Lafuente-Mercado. ARCA: Adapter-residual credit assignment when token signals degenerate. *arXiv preprint arXiv:2606.00257*, 2026.

Jiazheng Li, Yawei Wang, Qiaojing Yan, Yijun Tian, Zhichao Xu, Huan Song, Panpan Xu, and Lin Lee Cheong. SALT: Step-level advantage assignment for long-horizon agents via trajectory graph. In *Findings of EACL*, 2026.

Xiaomin Li, Mingye Gao, Yuexing Hao, Taoran Li, Guangya Wan, Zihan Wang, and Yijun Wang. MedGUIDE: Benchmarking clinical decision-making in large language models. *arXiv preprint arXiv:2505.11613*, 2025.

Yuze Liu, Chaochao Lu, and Chao Yang. SHADOW: Dynamic-aware credit assignment against long-horizon tasks. In *AAAI*, 2026.

Meta. Llama 3.2 model card. Meta model card, 2024.

Moonshot AI. Kimi K2.6: Advancing open-source coding. Moonshot AI technical blog, 2026.

OpenAI. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *CHIL*, 2022.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In *NAACL*, 2021.

Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adrià Puigdomènech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. In *ICML*, 2017.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, Jie Tang, and Yuxiao Dong. WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning. In *ICLR*, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents. Qwen blog, 2026a.

Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27B dense model. Qwen blog, 2026b.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork. Qwen blog, 2026c.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof Q&A benchmark. In *COLM*, 2024.

Bo Ren, Yirong Mao, Yi Yang, and Wenhui Que. IAPO: Influence-aware policy optimization for credit assignment in multi-turn service agents. *arXiv preprint arXiv:2608.24588*, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *ICLR*, 2021.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of ACL-IJCNLP*, 2021.

Hanyang Wang, Weijieying Ren, Yuxiang Zhang, Ding Cao, Zhizhao Zeng, Ke Zeng, and Tianxiang Zhao. BiPACE: Bisimulation-guided policy optimization with action counterfactual estimation for LLM agents. *arXiv preprint arXiv:2606.25556*, 2026a.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. ScienceWorld: Is your agent smarter than a 5th grader? In *EMNLP*, 2022.

Tao Wang, Suhang Zheng, and Xiaoxiao Xu. RTMC: Step-level credit assignment via rollout trees. *arXiv preprint arXiv:2604.11037*, 2026b.

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

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. In *ICLR*, 2026.

Weimin Xiong, Yifan Song, Xiutian Zhao, Wenhao Wu, Xun Wang, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. Watch every step! LLM agent learning via iterative step-level process refinement. In *EMNLP*, 2024.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *NeurIPS*, 2022.

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

Jing Yu, Yuqi Tang, Kehua Feng, Lei Liang, Qiang Zhang, Keyan Ding, and Huajun Chen. SciCUEval: A comprehensive dataset for evaluating scientific context understanding in large language models. *Scientific Data*, 2026.

Z.ai. GLM-5.3-Flash/FlashX. Z.ai developer documentation, 2026.

Huan Zhang, Mingju Chen, Dongxu Zhou, Can Lv, Heng Chang, Sen Cui, Faguo Wu, and Shiji Zhou. TRCA: Transition-wise rubric credit assignment for long-horizon LLM agents. *arXiv preprint arXiv:2608.16156*, 2026.

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. In *NeurIPS*, 2025.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ArCHer: Training language model agents via hierarchical multi-turn RL. In *ICML*, 2024.

A METHOD DETAILS

A.1 CAST PSEUDOCODE

Algorithm 1 summarizes the advantage computation in §3.3. It constructs latent neighborhoods within each task and selects references separately for each target record.

Algorithm 1: CAST advantage computation

Input: Collected records with returns G_t and episode-level advantages A_t^{episode}; policy π_θ; cosine-distance threshold δ; weight λ; constant $\epsilon > 0$.
Output: Combined advantages A_t for the collected records.

foreach *task in the batch* **do**
  Encode each current observation separately with π_θ and normalize its hidden state
  Form latent neighborhoods by greedy nearest-centroid clustering with threshold δ
  Construct each signature z_t using the fields specified in §3.3
  foreach *target record t of this task* **do**
    Let $\mathcal{C}_t$ be the latent neighborhood containing t
    $\mathcal{R}_t \leftarrow \{j \in \mathcal{C}_t : a_j \neq a_t \text{ or } z_j = z_t\}$
    $\mathcal{S}_t \leftarrow \{j \in \mathcal{R}_t : a_j = a_t\}$
    $\hat{Q}_t \leftarrow |\mathcal{S}_t|^{-1} \sum_{j \in \mathcal{S}_t} G_j, \quad \hat{V}_t \leftarrow |\mathcal{R}_t|^{-1} \sum_{j \in \mathcal{R}_t} G_j$
    $A_t^{\text{step}} \leftarrow \hat{Q}_t - \hat{V}_t$
    if *credit normalization is enabled* **then**
      $\sigma_t \leftarrow \sqrt{|\mathcal{R}_t|^{-1} \sum_{j \in \mathcal{R}_t} (G_j - \hat{V}_t)^2}$
      $A_t^{\text{step}} \leftarrow A_t^{\text{step}} / (\sigma_t + \epsilon)$
    end
    $A_t \leftarrow A_t^{\text{episode}} + \lambda A_t^{\text{step}}$
  end
end

A.2 ALGEBRAIC INTERPRETATION OF LOCAL CREDIT

Partition the retained reference set into same-action records $\mathcal{S}_t$ and other-action records $\mathcal{O}_t$:

$$\mathcal{S}_t = \{j \in \mathcal{R}_t : a_j = a_t\}, \qquad \mathcal{O}_t = \{j \in \mathcal{R}_t : a_j \neq a_t\}. \tag{5}$$

The target record satisfies $t \in \mathcal{S}_t$, so both $\mathcal{S}_t$ and $\mathcal{R}_t$ are nonempty. For a nonempty set $\mathcal{U}$, write $\bar{G}_{\mathcal{U}} = |\mathcal{U}|^{-1} \sum_{j \in \mathcal{U}} G_j$. When $\mathcal{O}_t \neq \varnothing$, the local estimates satisfy

$$\hat{Q}_t = \bar{G}_{\mathcal{S}_t}, \qquad \hat{V}_t = \frac{|\mathcal{S}_t| \bar{G}_{\mathcal{S}_t} + |\mathcal{O}_t| \bar{G}_{\mathcal{O}_t}}{|\mathcal{S}_t| + |\mathcal{O}_t|}. \tag{6}$$

Subtracting the baseline gives

$$\hat{Q}_t - \hat{V}_t = \frac{|\mathcal{O}_t|}{|\mathcal{S}_t| + |\mathcal{O}_t|} \left(\bar{G}_{\mathcal{S}_t} - \bar{G}_{\mathcal{O}_t}\right). \tag{7}$$

Thus, before normalization, local credit is the difference between the two mean returns, scaled by the fraction of other-action records in the retained set. The positive normalization factor $\sigma_t + \epsilon$, when used, preserves its sign.

No other-action references. If $\mathcal{O}_t = \varnothing$, then $\mathcal{R}_t = \mathcal{S}_t$ and $\hat{Q}_t = \hat{V}_t$. The step-level credit is therefore zero, with or without normalization.

Agreement with the unfiltered estimator. If every same-action record in $\mathcal{C}_t$ has signature z_t, filtering removes no record and $\mathcal{R}_t = \mathcal{C}_t$. The local action value, baseline, and step-level credit then equal those of the unfiltered estimator under the same normalization setting.

B EXPERIMENTAL DETAILS

B.1 IMPLEMENTATION AND TRAINING SETTINGS

Observation representations and local credit. For both Qwen backbones, we encode each current observation separately with the policy model, without adding the task instruction or interaction history to this encoding input. The representation is the hidden state at the last non-padding token, taken from the eighth-to-last hidden-state layer and normalized to unit length. Observation encoding uses the same token limit as the policy prompt. Greedy clustering operates within each task in the current batch, with no reference buffer from previous updates. The discount factor is $\gamma = 0.95$, and the coefficient of step-level credit is $\lambda = 1$. The cosine-distance threshold is 0.1 on ALFWorld, BabyAI, and ScienceWorld, and 0.08 on WebShop. When standard-deviation normalization is enabled, we use $\epsilon = 10^{-6}$. Action groups use the executed environment action.

Reward processing. For each invalid action, we subtract 0.1 from the corresponding training record's discounted return and from its score for the episode-level term before computing advantages. The penalty is applied after discounting and is not propagated to earlier steps. The local return aggregation in §3.3 uses these adjusted returns. Transition signatures retain the original environment reward r_t.

Optimization and interaction limits. Table 7 gives the environment-dependent settings shared by Qwen3.5-4B and Qwen3.5-9B. These configurations use a learning rate of 5×10^{-7}, one policy-optimization epoch per batch, a PPO clipping ratio of 0.2, and a KL-loss coefficient of 0.01. The configured optimization minibatch size is 128, with one training record per GPU microbatch. Training uses fully sharded data parallelism on eight GPUs. Each collection batch contains 16 task inputs, with the number of episodes per input specified in the table. Each policy response is limited to 128 tokens.

Table 7: Environment-dependent training settings for both Qwen backbones. Prompt limits are measured in tokens; episode limits count environment interactions.

Environment	Episodes per task input	Prompt limit	Episode limit	Clustering threshold	Credit normalization
ALFWorld	8	2048	50	0.1	Yes
BabyAI	8	2048	20	0.1	Yes
ScienceWorld	8	2048	50	0.1	Yes
WebShop	12	4096	15	0.08	No

Training seeds. For ALFWorld and WebShop, we use a base seed of 1 for training-environment initialization. BabyAI and ScienceWorld instances are determined by fixed task identifiers.

Training budgets. For both Qwen backbones, training uses 100 updates on BabyAI and WebShop, 150 on ScienceWorld, and 175 on ALFWorld. The Llama-3.2-3B-Instruct experiments use 100 updates on both BabyAI and WebShop. Their learning rate, task batch size, episode group size, and interaction limits follow the corresponding Qwen3.5-9B configurations above. The `CAST` and BiPACE configurations also retain the corresponding local-credit settings. For evaluation, we use the checkpoint saved at the prescribed maximum number of training updates.

B.2 INTERACTION PROMPTS

Our interactive agents use ReAct-style prompts (Yao et al., 2023) with `Thought` and `Action` fields. The policy receives the task instruction, current observation, environment-specific action guidance, and recent interaction history when available. Figure 8 shows representative excerpts for ScienceWorld and WebShop.

ScienceWorld

```
You are an expert agent operating in the ScienceWorld environment.
Task: {task_description}
Prior to this step, you have already taken {step_count} step(s). Below
are the most recent {history_length} observations and corresponding
actions:
{action_history}
You are now at step {current_step}.
Observation: {current_observation}

Here are the actions you may take:
[...]
look around - describe the current room
look at OBJ - describe an object in detail
[...]
go to LOC - move to a new location
[...]
focus on OBJ - signal intent on a task object
[...]
Replace OBJ/LOC with a concrete object or location that exists in the
current room;
never write the literal words OBJ or LOC.
[...]
Thought: <one short sentence>
Action: <one environment action>
[...]
```

WebShop

```
You are an expert autonomous agent operating in the WebShop e-commerce
environment.
Your task is to: {task_description}.
Prior to this step, you have already taken {step_count} step(s). Below
are the most recent {history_length} observations and the
corresponding actions you took: {action_history}
You are now at step {current_step} and your current observation is:
{current_observation}.
Your admissible actions of the current situation are:
[
{available_actions}
].
[...]
Thought: <one short sentence>
Action: <one action>
For search[<your query>], replace the placeholder with concrete search
keywords. For click, copy one listed action exactly.
[...]
```

Figure 8: Representative interaction prompt excerpts for ScienceWorld and WebShop. Braced fields are filled at each interaction step; `[...]` indicates omitted instructions.

B.3 Evaluation protocol

Interactive environments. The training pools for ALFWorld, BabyAI, ScienceWorld, and WebShop contain 3,553, 810, 2,120, and 6,410 instances, respectively. The corresponding validation sets contain 140, 90, 200, and 500 instances. The training, validation, and final evaluation sets are mutually disjoint within each environment. ALFWorld and BabyAI report the percentage of episodes with a positive environment success score. WebShop reports strict success: an episode is successful when its final score exceeds 0.999999. ScienceWorld reports the mean final task score on the 0–100 scale after clipping negative scores to zero. In Table 2, ALFWorld All pools the 200 tasks rather than averaging the six task-type scores. Average gives equal weight to ALFWorld All, BabyAI, ScienceWorld, and WebShop.

Table 8: Evaluation sets for general capability retention, with the number of questions per benchmark.

Benchmark	Evaluation set	Questions
FOLIO	Development set	204
ProofWriter	Test subset	600
MMLU-Math	Mathematics test questions	1,064
SVAMP	Test set	300
MATH500	MATH-500 test set	500
MedMCQA	Single-answer subset	1,000
LongHealth	Benchmark version 5	400
MedGUIDE	Questions with at least seven options	1,732
GPQA Diamond	Diamond subset	198
SuperGPQA	Hard science and medicine subset	1,000
SciCUEval	Multiple-choice questions	1,675

General capability benchmarks. Table 8 specifies the evaluation sets used in Figure 7. The overall Average gives equal weight to the 11 benchmark accuracies.

Answer scoring. Multiple-choice prompts request a final line in the form `ANSWER: <LETTER>`. Answer options are shuffled deterministically per question, with the same ordering used across models. The parser first checks an explicit answer field, then common final-answer formats and isolated option letters. LongHealth questions include the associated clinical records, and SciCUEval questions include their scientific context. SVAMP and MATH500 use open-form answers: the prompt requests the final answer inside `\boxed{}`, and the grader checks the extracted answer against the reference using Math-Verify 0.8.0 with strict whole-answer parsing. All questions are scored using these rules, with responses lacking a parsable answer counted as incorrect.

B.4 Analysis protocols

Recorded-batch diagnostics. The mechanism analysis uses 50 recorded ScienceWorld batches with Qwen3.5-9B: updates 1–25 and 126–150. The early and late windows contain 137,640 and 130,392 production records, respectively. Only `CAST` updates the policy; the unfiltered estimator and *w/o SO* are evaluated on the same representations, neighborhoods, actions, and returns. The latter uses immediate-reward and termination consistency without checking the successor observation. A credit change is an absolute difference greater than 10^{-5} between the normalized step-level credits from two estimators. Each comparison is computed directly; the two pairwise change rates cannot be added to obtain the full-versus-unfiltered rate.

Window-level rates pool the corresponding numerators and denominators across updates. Mean reference counts are weighted by production-record counts and exclude the target record. The rate of losing all same-action references is conditioned on having at least one non-target same-action reference before filtering. Faint curves retain the per-update measurements. Solid curves pool three adjacent updates within each window, using two updates at a window boundary.

Clustering-threshold scan. We apply the production greedy-clustering procedure offline to the same recorded batches, holding embeddings and record order fixed. All scan entries describe unfiltered reference sets. For each record, we check whether its neighborhood contains a same-action reference with a different transition signature. Affected records are partitioned according to whether any such reference shares the target's current observation. Current observations are compared after lowercasing, trimming leading and trailing whitespace, and collapsing consecutive whitespace. Both percentages use all production records as their denominator, so their sum gives the total affected fraction. Other-action coverage is the fraction with at least one reference taking a different action. This scan measures reference-set properties without retraining the policy.

Training-stage timing. We measure stage times over 900 training updates across the eight `CAST` model–environment settings, excluding updates with validation or checkpoint saving. Observation encoding is timed within the advantage-computation stage. Subtracting its time gives the remaining stage cost, which includes reward retrieval, invalid-action penalties, action processing, latent

clustering, and credit computation. Each reported share divides summed stage time by summed training-update time within the corresponding setting.

Credit-function timing. We compare the unfiltered estimator with CAST on the 50 cached ScienceWorld batches using a single CPU core and one thread. Both conditions use identical cached inputs, including cluster assignments and returns. Encoding, clustering, and data loading occur outside the timed function. Points in Figure 6(b) represent batch-level mean wall times; the reported overall values weight the batches equally. These measurements isolate credit-function cost and are separate from the training-stage shares in panel (a).

B.5 Additional results

B.5.1 General capability retention

Table 9 reports general capability retention for both model sizes. Each model is evaluated four times on the same set of questions for each benchmark, with fixed model weights and evaluation seeds 42, 43, 44, and 45. For each benchmark, we report the mean accuracy over the four evaluations, rounded to one decimal place. Average is the equally weighted mean of the 11 unrounded benchmark means, rounded to one decimal place.

Table 9: General capability retention after training with CAST. Base and +CAST report accuracy (%) before and after training. Average gives equal weight to the 11 evaluation benchmarks.

Domain	Benchmark	Qwen3.5-4B		Qwen3.5-9B	
		Base	+CAST	Base	+CAST
Logical	FOLIO	78.4	78.6	80.9	82.0
	ProofWriter	83.8	81.8	86.1	86.5
Math	MMLU-Math	93.8	94.4	94.8	94.9
	SVAMP	93.3	92.8	94.0	94.3
	MATH500	76.6	78.7	81.2	81.6
Medical	MedMCQA	66.9	65.6	72.7	73.3
	LongHealth	85.7	86.4	86.9	87.4
	MedGUIDE	29.5	30.4	36.9	38.3
Science	GPQA Diamond	59.1	59.0	62.9	63.6
	SuperGPQA	31.2	31.4	35.1	35.0
	SciCUEval	91.1	91.0	91.2	91.3
Average		**71.8**	**71.8**	**74.8**	**75.3**

B.5.2 Validation dynamics of w/o immediate reward

To examine the low final performance of *w/o immediate reward*, we compare two training runs on ScienceWorld with Qwen3.5-9B, each lasting 150 updates. The transition signature retains termination status and the normalized successor observation; environment rewards remain unchanged. The final-checkpoint test score of 27.7 in Table 6 corresponds to the curve labeled Original run in Figure 9.

Figure 9 shows validation scores throughout training. The original run declines from 47.9 at update 55 to 28.8 at the final update. The additional run reaches 41.2 at update 140 and finishes at 26.7. Both runs exhibit late validation degradation, although its onset and magnitude differ.



Figure 9: Late validation degradation without immediate-reward consistency on ScienceWorld with Qwen3.5-9B. Both training runs end below their earlier scores. Curves show unsmoothed mean task scores (0–100) every five updates, clipping negative scores to zero. These validation scores are distinct from the test score in Table 6.