CWME: CONTINUAL WORLD MODEL EVOLUTION VIA INTERACTION EXPERIENCE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

World models provide interactive agents with environmental knowledge to support reasoning and decision making. However, continual interaction introduces new states and outcomes that may reveal limitations in previously acquired knowledge. Two key challenges arise in continual world modeling: (1) Interaction feedback is insufficiently integrated to validate and refine existing world knowledge, limiting its continual evolution. (2) Task-relevant historical experience fails to adapt to evolving environmental states, making direct reuse unreliable and repeated reasoning costly. To address these challenges, we propose **CWME**, a framework for **C**ontinual **W**orld **M**odel **E**volution via Interaction Experience Learning. CWME maintains Situational Knowledge, Action Trajectory Experience, and Failure-Aware Knowledge. These forms of knowledge provide environmental constraints, behavioral priors, and reuse boundaries for future decisions. Knowledge-Informed Trajectory Generation adapts historical behavioral structure to the current interaction instead of replaying past solutions. Experience-Guided Decision Routing distinguishes retrieval relevance from current-state applicability and selects direct reuse, lightweight verification, or fresh reasoning accordingly. Evolving Interaction Experience Updating incorporates interaction outcomes to refine accumulated knowledge and the conditions under which it can be reused. Extensive experiments on the public interactive benchmarks demonstrate that CWME outperforms all evaluated baselines in task performance while improving reasoning efficiency, further demonstrating the benefits of continual knowledge refinement and state-aware experience reuse.

1 INTRODUCTION

World models, updating internal knowledge and environmental state as agents interact with the environment, learn the dynamics of environmental evolution and predict action outcomes to support future reasoning and decision making. Recently, world models have broad applications in game playing, autonomous driving, and embodied intelligence.

Many world model approaches primarily learn environment dynamics to predict future states and action outcomes (Liu et al., 2025; Fu et al., 2025). The acquired knowledge is often encoded implicitly in model parameters or latent representations, which limits selective updates as the environment changes. In contrast, structured world models provide more direct access to environmental knowledge for reasoning and updating (Feng et al., 2025; Bai et al., 2026). However, these approaches make it difficult to maintain and adjust existing knowledge as new interactions occur. More recent continual learning and self-evolving approaches accumulate interaction experience and historical knowledge to update world models and support future decisions (Feng et al., 2026; Zhang et al., 2026b). Although these approaches have made substantial progress, they still face two key challenges.

(1) Insufficient integration of interaction feedback to drive continual evolution of world knowledge. Existing approaches can continuously accumulate experience through interaction with new environments, but they often focus on knowledge expansion and local revision rather than continually integrating interaction feedback. Such feedback not only provides new environmental information but also reflects the actual outcomes of past actions, revealing biases and limitations in existing knowledge under new conditions. If this feedback is retained only as isolated historical experience

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.

instead of being used to validate existing knowledge, the world model cannot effectively transform multi-round interaction experience into continuously evolving world knowledge.

(2) Historical experience relevant to the task is not adaptive to the dynamic evolution of states. Existing experience-reuse methods commonly retrieve historical knowledge or trajectories based on task or semantic relevance to support current decisions. However, relevance does not guarantee current-state applicability, as changes in object states, action preconditions, or environmental relations may invalidate previously successful trajectories. Moreover, an executable trajectory may still contribute little to task progress. Therefore, direct reuse can lead to inappropriate decisions, whereas repeatedly reasoning from scratch incurs unnecessary computational cost.

To address these challenges, we propose **CWME**, a framework for **C**ontinual **W**orld **M**odel **E**volution via Interaction Experience Learning. CWME aims to balance effective experience reuse with reasoning efficiency by adapting prior experience to the current interaction and dynamically allocating reasoning effort according to its applicability. Knowledge-Informed Trajectory Generation uses prior experience as structured guidance and adapts reusable behavioral structure to the current interaction. Experience-Guided Decision Routing separates retrieval relevance from current-state applicability and determines how much additional reasoning is needed before execution. Evolving Interaction Experience Updating incorporates new interaction outcomes to refine both accumulated knowledge and the conditions under which it can be reused. Together, these components connect reliable experience reuse with continual world model evolution.

The main contributions of this work are summarized as follows:

- We propose **CWME**, a framework for continual world model evolution, which gracefully exploits accumulated interaction knowledge to guide current decision-making while informing knowledge evolution.
- We introduce situational knowledge, action-trajectory experience, and failure-aware knowledge via interaction experience, which equips the world model with environmental constraints, behavioral priors, and reuse boundaries to guide future decision-making.
- We propose an applicability-aware experience reuse strategy capable of distinguishing retrieval relevance from current-scene applicability, dynamically adjusting inference overhead based on their discrepancy, and avoiding unreasonable reuse behaviors.
- Experiments on ScienceWorld and ALFWorld show that CWME improves task performance while reducing reliance on full reasoning. Controlled analyses further validate continual world model evolution and applicability-aware experience reuse.

2 Related Work

2.1 World Models for Interactive Reasoning

World models represent environment states and action effects to support reasoning and planning. Predictive approaches model environment dynamics to anticipate future states and outcomes (Liu et al., 2025; Fu et al., 2025). RAP uses a language model as a world model to evaluate prospective reasoning trajectories (Hao et al., 2023). Recent studies further explore explicit world representations. Graph World Model represents states and actions with structured graphs (Feng et al., 2025). PatchWorld induces executable symbolic world models from interaction trajectories (Bai et al., 2026). These methods demonstrate the value of structured world knowledge, but mainly emphasize world model construction or its use in current reasoning. CWME instead treats the world model as an evolving representation that is continually refined through interaction experience.

2.2 Continual and Self-Evolving World Models

Continual world modeling updates environment knowledge as interaction experience accumulates. CLIN learns causal abstractions across repeated trials (Majumder et al., 2024). EvolvingAgent introduces a continual world model for long-horizon interaction (Feng et al., 2026). WorldEvolver revises episodic and semantic knowledge from interaction outcomes (Zhang et al., 2026b). These methods show that interaction feedback can support continual knowledge evolution without repeatedly updating the underlying language model. However, they place less emphasis on jointly main-

taining complementary forms of interaction knowledge and the conditions that govern their validity. CWME organizes evolving knowledge into Situational Knowledge, Action Trajectory Experience, and Failure-Aware Knowledge to capture environmental constraints, behavioral priors, and reuse boundaries.

2.3 Experience Reuse and Adaptive Reasoning

Experience-based agents reuse prior interactions to reduce repeated reasoning. ExpeL distills transferable knowledge (Zhao et al., 2024). Agent Workflow Memory abstracts successful trajectories into reusable workflows (Wang et al., 2025). MemP maintains continually updated procedural memory (Fang et al., 2026). Reflexion incorporates interaction feedback into future reasoning (Shinn et al., 2023). MemRL learns the utility of retrieved memories from environmental feedback (Zhang et al., 2026a). SwiftSage adapts reasoning effort between lightweight and deliberative modes (Lin et al., 2023). However, retrieval relevance and historical utility alone do not establish whether a grounded trajectory candidate remains applicable to the current state. CWME evaluates the current-state applicability of grounded trajectory candidates and adapts reasoning effort among direct reuse, lightweight verification, and fresh reasoning.

3 Preliminaries

We consider a continual interactive setting consisting of a sequence of episodes $\{\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_N\}$. In episode e, the agent receives a task instruction g_e and interacts with the environment over T_e steps. The resulting trajectory is

$$\tau_e = (o_{e,0}, a_{e,0}, o_{e,1}, \ldots, a_{e,T_e-1}, o_{e,T_e}), \tag{1}$$

where $o_{e,t}$ and $a_{e,t}$ denote the observation and action at step t.

Unlike independent episodes, knowledge acquired from earlier interactions can support future reasoning. Let $\mathcal{M}_{e,t}$ denote the world model available at step t of episode e, combining the current interaction state with world knowledge accumulated from previous interactions. The agent selects actions according to

$$a_{e,t} \sim \pi_\theta(\cdot \mid g_e, \mathcal{M}_{e,t}), \tag{2}$$

where π_θ is the language-model-based reasoning policy.

During continual interaction, new states, transitions, and outcomes update the world model, while the resulting knowledge is retained across episodes but not used to fine-tune the large language model. The world model's representation and evolution mechanism are introduced in Section 4.

4 Continual World Model Evolution

The framework of CWME is illustrated in Figure 1 and consists of three main components. **(1) Knowledge-Informed Trajectory Generation** retrieves reusable historical experience according to the current state, providing trajectory candidates for future decision making. **(2) Experience-Guided Decision Routing** assesses the validity and usefulness of experience in the current state and dynamically selects the required reasoning route. **(3) Evolving Interaction Experience Updating** continuously updates the world model with new interaction experience to support future decision making.

At step t, the world model state is

$$\mathcal{M}_t = (s_t, \mathcal{K}_t), \tag{3}$$

where s_t denotes the interaction state available to the agent, including the current observation and recent interaction history, and $\mathcal{K}_t$ denotes the evolving world knowledge. For simplicity, we omit the episode index when the context is clear.

4.1 Evolving World Model

To support reliable experience reuse under changing states, CWME maintains an evolving world model that combines the current interaction state with continually refined world knowledge. Given

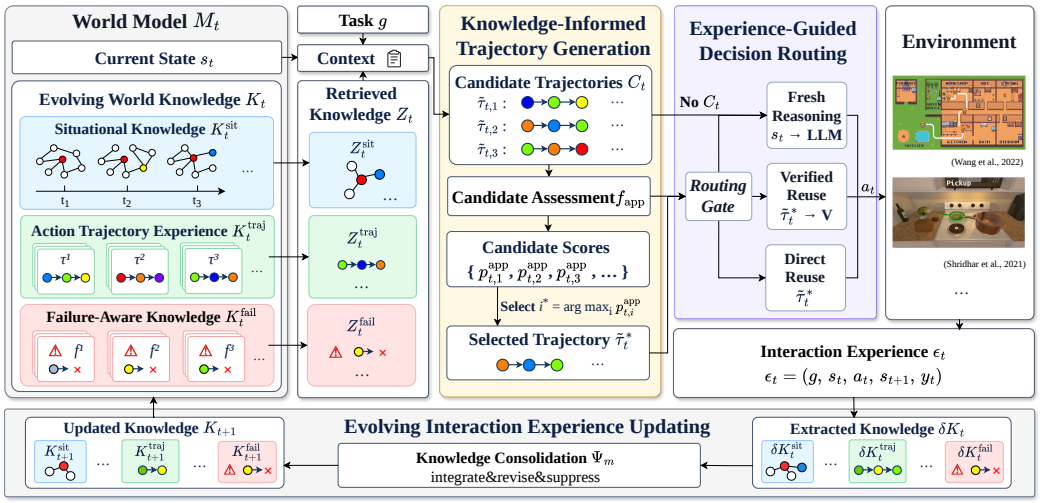


Figure 1: The overall architecture of CWME.

task g and state s_t, the interaction context is

$$q_t = (g, s_t). \tag{4}$$

CWME organizes world knowledge as

$$\mathcal{K}_t = \left\{ \mathcal{K}_t^{\text{sit}}, \mathcal{K}_t^{\text{traj}}, \mathcal{K}_t^{\text{fail}} \right\}. \tag{5}$$

Situational Knowledge. $\mathcal{K}_t^{\text{sit}}$ captures state-dependent environmental information, including entities, relations, action preconditions, and observed action-conditioned transitions, all of which provide state constraints for grounding candidate actions in the current environment.

Action Trajectory Experience. $\mathcal{K}_t^{\text{traj}}$ stores reusable behavior together with its validity conditions. A trajectory experience is represented as

$$\xi_i = (g_i, C_i, A_i, \Gamma_i, \rho_i), \tag{6}$$

where g_i is the task context, C_i denotes validity conditions, A_i contains the ordered action structure, Γ_i records associated state changes, and ρ_i summarizes historical reuse reliability. The implementation additionally maintains outcome and reuse statistics for confidence estimation and consolidation.

Failure-Aware Knowledge. $\mathcal{K}_t^{\text{fail}}$ captures state-dependent reuse boundaries derived from unsuccessful interactions, which identify conditions under which prior actions or trajectory stages should be suppressed or reconsidered during future reuse.

Together, these components provide environment constraints, reusable behavior, and reuse boundaries for future reasoning and decision making.

4.2 Knowledge-Informed Trajectory Generation

Action trajectories derived from historical experience may become incompatible as the environment changes. To address this limitation, CWME retrieves and grounds relevant trajectories according to the current environment rather than directly reusing historical experience. The knowledge relevant to the current environment is retrieved as follows:

$$\mathcal{Z}_t^m = R_m(q_t, \mathcal{K}_t^m), \qquad m \in \{\text{sit}, \text{traj}, \text{fail}\}. \tag{7}$$

Retrieved trajectories may not directly match the current task progress, so CWME identifies the appropriate stage for reuse by

$$j_i = \text{Align}(q_t, \xi_i). \tag{8}$$

Here, $\xi_i \in \mathcal{Z}_t^{\mathrm{traj}}$ denotes a retrieved trajectory experience, and j_i denotes its resume stage. Stage alignment determines the reuse point but does not adapt subsequent actions to the current environment state. CWME therefore grounds a short-horizon trajectory segment using Situational Knowledge and Failure-Aware Knowledge:

$$\widetilde{\tau}_{t,i} = \mathcal{G}\big(q_t, \xi_i, j_i, \mathcal{Z}_t^{\mathrm{sit}}, \mathcal{Z}_t^{\mathrm{fail}}\big), \tag{9}$$

where $\mathcal{G}$ grounds the segment $A_i[j_i : j_i + H_{\mathrm{traj}} - 1]$ to the current state, matching Failure-Aware Knowledge to the retrieved trajectory and aligned stage. The successfully grounded candidates form

$$\mathcal{C}_t = \{\widetilde{\tau}_{t,i}\}_{i=1}^{N_t}. \tag{10}$$

Here, N_t denotes the number of successfully grounded trajectory candidates at step t. To accommodate dynamic changes in the environment, only the first action of each candidate trajectory is considered at the current step, while the candidate set is reconstructed after each environment update.

4.3 Experience-Guided Decision Routing

Task-relevant experience may no longer support current decisions when environment states differ from conditions of previous success. To avoid inappropriate reuse, CWME evaluates each grounded candidate based on first-action validity, contribution to local task progress, and contextual information from short-horizon trajectories. The resulting score then guides dynamic reasoning-route selection.

Candidate Assessment. To evaluate how well each grounded candidate matches the current decision state, CWME jointly represents state information, trajectory context, and their pairwise relations. Let x_t^s denote the current task and state, and let $x_{t,i}^\tau$ contain the current grounded stage and up to $H_{\mathrm{app}} - 1$ following stages of candidate $\widetilde{\tau}_{t,i}$. Using a fixed pretrained encoder f_e, CWME constructs

$$\mathbf{h}_{t,i} = \left[\mathbf{e}_t^s; \mathbf{e}_{t,i}^\tau; |\mathbf{e}_t^s - \mathbf{e}_{t,i}^\tau|; \mathbf{e}_t^s \odot \mathbf{e}_{t,i}^\tau\right], \tag{11}$$

where $\mathbf{e}_t^s = f_e(x_t^s)$ and $\mathbf{e}_{t,i}^\tau = f_e(x_{t,i}^\tau)$, $[\cdot;\cdot]$ denotes feature concatenation, and $\odot$ denotes element-wise multiplication. A learned applicability head maps the joint representation to a decision-level applicability score:

$$p_{t,i}^{\mathrm{app}} = \sigma\left(f_{\mathrm{app}}(\mathbf{h}_{t,i})\right), \tag{12}$$

where f_{app} denotes the learned prediction head and σ is the sigmoid function. Candidates with $p_{t,i}^{\mathrm{app}} < \delta_{\mathrm{app}}$ are filtered out, where δ_{app} is selected on the validation set. If no candidate remains, CWME invokes Fresh Reasoning; otherwise, the candidate with the highest applicability score is passed to the router.

Decision Routing. Candidate evaluation determines whether historical experience can support the current decision, while different cases may require different levels of additional reasoning. To determine an appropriate reasoning route, CWME represents the selected candidate using trajectory confidence, historical reuse reliability, world-model support, failure context, interaction progress, and the candidate evaluation score. The router predicts

$$\mathbf{p}_t^{\mathrm{route}} = \mathrm{Softmax}\left(f_{\mathrm{route}}(\mathbf{r}_t)\right), \tag{13}$$

over

$$\mathcal{R} = \{\text{Direct Reuse},\ \text{Verified Reuse},\ \text{Fresh Reasoning}\}.$$

Direct Reuse executes the grounded action without additional reasoning. Verified Reuse performs lightweight verification under the current interaction context before reuse. Fresh Reasoning invokes the full reasoning procedure.

Together, candidate assessment and dynamic reasoning routing constitute an applicability-aware experience reuse strategy that distinguishes retrieval relevance from current-state applicability and dynamically adjusts reasoning effort to avoid inappropriate reuse and unnecessary reasoning.

Learning Assessment and Routing. Candidate assessment is supervised by the outcome of the executed first action, while the remaining short-horizon trajectory provides contextual information for the decision. Routing supervision compares the three reasoning routes under the same decision state with world knowledge fixed and selects the successful route requiring the least additional reasoning. Detailed supervision rules and training objectives are provided in Appendix A.3.

4.4 Evolving Interaction Experience Updating

As interaction continues, newly observed states, transitions, and outcomes may make existing world knowledge incomplete or outdated. To maintain reliable knowledge for future experience reuse and decision making, CWME continually updates the world model from new interaction experience rather than simply accumulating interaction history.

After executing a_t, the resulting interaction experience is represented as $\epsilon_t = (g, s_t, a_t, s_{t+1}, y_t)$, where y_t summarizes the observed outcome and local progress signal. Component-specific updates are extracted as

$$\delta\mathcal{K}_t = \Phi(\epsilon_t) = \left\{\delta\mathcal{K}_t^{\mathrm{sit}}, \delta\mathcal{K}_t^{\mathrm{traj}}, \delta\mathcal{K}_t^{\mathrm{fail}}\right\}. \tag{14}$$

Here, Φ extracts new information for the three world-knowledge components from current interaction experience. Each component is then consolidated as

$$\mathcal{K}_{t+1}^m = \Psi_m(\mathcal{K}_t^m, \delta\mathcal{K}_t^m), \qquad m \in \{\mathrm{sit}, \mathrm{traj}, \mathrm{fail}\}. \tag{15}$$

Here, Ψ_m integrates the newly extracted information with existing knowledge of component m.

Consolidation can *extend*, *reinforce*, *revise*, or *suppress* existing knowledge. State transitions update Situational Knowledge, successful reusable segments strengthen Action Trajectory Experience, while invalid actions or failed reuse update Failure-Aware Knowledge. Later successful interactions can weaken obsolete failure constraints. The resulting $\mathcal{K}_{t+1}$ supports subsequent experience reuse, reasoning, and decision making.

5 Experiments

5.1 Experimental Setting

Datasets. We evaluate CWME on ScienceWorld (Wang et al., 2022) and ALFWorld (Shridhar et al., 2021). ScienceWorld is used for full-benchmark evaluation and controlled mechanism analysis, while ALFWorld is evaluated under the official seen and unseen settings. Detailed task statistics are provided in Appendix B.

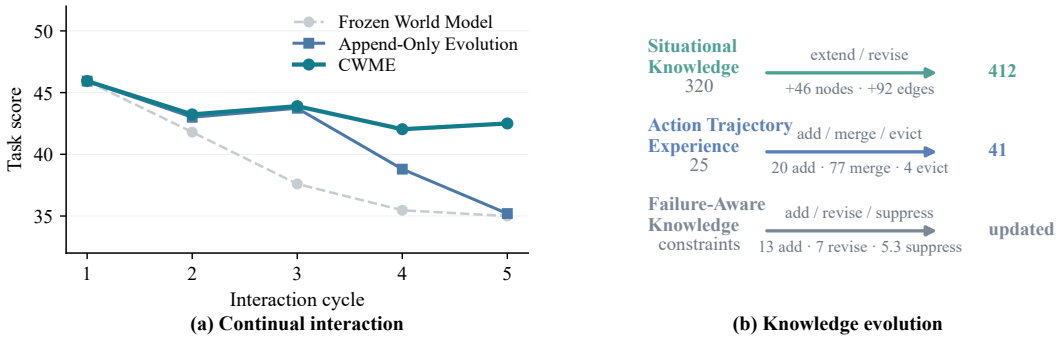
Baselines. We compare CWME with ten representative interactive-agent methods: SayCan (Ichter et al., 2023), ReAct (Yao et al., 2023), Reflexion (Shinn et al., 2023), RAP (Hao et al., 2023), AgentGym (Xi et al., 2025), PatchWorld (Bai et al., 2026), ExpeL (Zhao et al., 2024), CLIN (Majumder et al., 2024), MemP (Fang et al., 2026), and MemRL (Zhang et al., 2026a). All methods use Qwen3.5-9B (Qwen Team, 2026) as the full-reasoning backbone. Further details are provided in Appendix B.2.

Implementation Details. CWME uses Qwen3.5-9B for Fresh Reasoning, Qwen3.5-4B for Verified Reuse, and no additional LLM call for Direct Reuse. The continual evolution experiment adopts a cycle-wise interleaved protocol, whereas ablation and other controlled experiments follow a task-family-grouped protocol. During evaluation, the learned components and language models remain fixed, while only structured world knowledge evolves through interaction. Unless otherwise specified, results are averaged over three seeds and reported as mean $\pm$ standard deviation. More details are provided in Appendix B.3.

Metrics. We report task score on ScienceWorld and success rate on ALFWorld. Applicability is evaluated with Macro-F1, AUROC, and AUPRC. Reasoning efficiency is measured by Fresh Calls, Other Calls, total token usage, and wall-clock time. Fresh Calls count invocations of the full-reasoning role, while Other Calls count the lightweight verifier and any other auxiliary language-model calls.

Table 1: Overall performance on ScienceWorld and ALFWorld. The best results are in bold and the second-best results are underlined. Improvement denotes the relative percentage gain over the strongest baseline in each column.

Method	ScienceWorld	ALFWorld	
	Task Score ↑	Seen SR (%) ↑	Unseen SR (%) ↑
SayCan	20.96 ± 3.12	29.70 ± 0.41	30.13 ± 1.31
ReAct	28.49 ± 0.57	13.91 ± 1.28	13.53 ± 2.14
Reflexion	20.63 ± 2.43	39.72 ± 1.13	$\underline{56.97 \pm 3.37}$
RAP	35.35 ± 0.45	28.60 ± 3.47	33.45 ± 0.79
AgentGym	22.43 ± 1.37	18.65 ± 0.86	12.07 ± 1.67
PatchWorld	16.81 ± 0.56	45.03 ± 1.96	44.25 ± 0.80
ExpeL	11.75 ± 0.33	10.68 ± 1.28	16.87 ± 0.00
CLIN	35.77 ± 0.64	48.07 ± 1.74	54.90 ± 1.33
MemP	13.63 ± 2.53	57.75 ± 1.90	44.11 ± 2.52
MemRL	$\underline{42.63 \pm 2.92}$	$\underline{59.07 \pm 2.22}$	51.03 ± 1.06
CWME (ours)	$\mathbf{52.02 \pm 0.90}$	$\mathbf{60.40 \pm 1.96}$	$\mathbf{68.17 \pm 1.70}$
Improvement	+22.03%	+2.25%	+19.66%



(a) Continual interaction **(b) Knowledge evolution**

Figure 2: Continual world model evolution on ScienceWorld: (a) cycle-wise performance and (b) knowledge evolution.

5.2 Overall Performance

Table 1 presents the overall performance on ScienceWorld and ALFWorld. CWME consistently achieves the best performance across all three settings. In contrast, experience-based methods such as CLIN, MemP, and MemRL remain competitive on individual settings but fail to maintain consistent performance across all evaluations. The limitation arises because retrieved experience can remain task-relevant while becoming inapplicable when object states, action preconditions, or environmental relations change. CWME addresses this issue by grounding historical trajectories in the current state and explicitly assessing their applicability before reuse. Task-wise results are provided in Appendix C.

5.3 Continual World Model Evolution

To evaluate continual world model evolution, we compare CWME with frozen and append-only variants on a controlled ScienceWorld stream. As shown in Figure 2(a), performance varies across interaction cycles as task instances and environment states change. Therefore, we focus on matched within-cycle comparisons to assess the effects of different knowledge-evolution mechanisms. The frozen model retains its initial knowledge without incorporating new outcomes, whereas append-only evolution accumulates new experience without revising existing knowledge. In contrast, CWME uses new interaction outcomes to refine world knowledge, keeping it better aligned with future interaction states and supporting more effective reasoning as experience accumulates. Figure 2(b) further shows how this refinement is realized across the three knowledge components. Situational Knowledge is extended or revised, Action Trajectory Experience is added, merged, or

Table 2: Comparison of different strategies on held-out action trajectory candidates.

Strategy	Macro-F1 ↑	AUROC ↑	AUPRC ↑
Similarity	0.548 ± 0.055	0.750 ± 0.044	0.933 ± 0.034
Reliability	0.624 ± 0.063	0.727 ± 0.093	0.906 ± 0.057
Similarity + Reliability	$\underline{0.672 \pm 0.050}$	$\underline{0.808 \pm 0.060}$	$\underline{0.944 \pm 0.033}$
CWME Applicability (ours)	$\mathbf{0.731 \pm 0.090}$	$\mathbf{0.879 \pm 0.085}$	$\mathbf{0.970 \pm 0.022}$

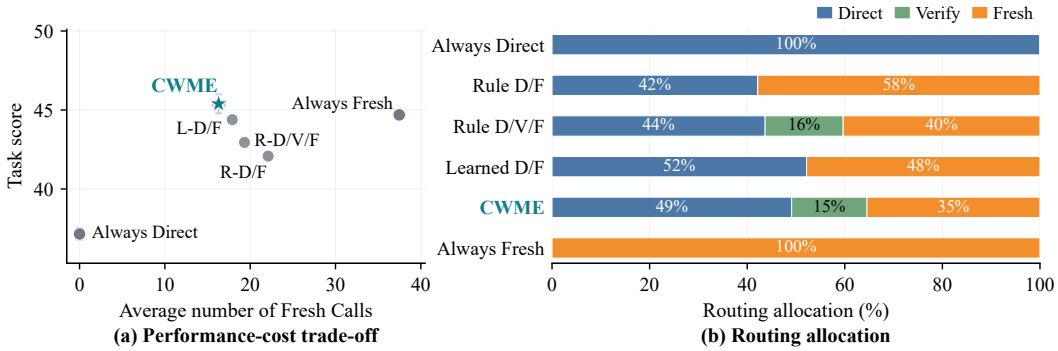


Figure 3: Adaptive decision routing on ScienceWorld: (a) performance–reasoning trade-off and (b) routing distribution.

evicted, and Failure-Aware Knowledge is updated from interaction outcomes. Therefore, CWME evolves the world model through selective refinement rather than simple accumulation, allowing new experience to correct outdated knowledge, consolidate reusable experience, and maintain more reliable world knowledge over time. Detailed results are provided in Appendix D.

5.4 Candidate Assessment and Adaptive Decision Routing

To examine whether retrieved trajectories remain applicable to the current state, we evaluate candidate applicability and its role in allocating decisions among direct reuse, verified reuse, and fresh reasoning. Table 2 shows that similarity and historical reliability both provide useful signals, while combining them further improves prediction. CWME Applicability performs best across all three metrics, indicating that current-state information provides additional guidance beyond retrieval relevance and historical success. This supports explicitly separating trajectory retrieval from Candidate Assessment. Figure 3(a) compares different reasoning-allocation strategies. Always Direct minimizes reasoning cost but performs poorly when previous experience is insufficient, whereas Always Fresh repeatedly invokes full reasoning. Learned routing provides a better trade-off. The three-way CWME router further introduces Verified Reuse as an intermediate option when direct reuse is uncertain but full reasoning is unnecessary. Figure 3(b) shows that CWME makes use of all three routes: direct reuse for applicable experience, lightweight verification for uncertain reuse, and fresh reasoning when previous experience is insufficient. This allocation allows reasoning effort to vary with the estimated reuse condition. Additional results are provided in Appendix E.

5.5 Ablation Studies

To evaluate the contribution of each CWME component, we remove the three forms of world knowledge and the main applicability, routing, and evolution mechanisms. Table 3 shows that the three forms of world knowledge provide complementary information. Removing Action Trajectory Experience causes the largest performance degradation and substantially increases full-reasoning usage, highlighting the importance of reusable behavioral structure. Removing Situational Knowledge also reduces performance and increases reasoning cost, showing that historical experience benefits from grounding in the current state. Failure-Aware Knowledge has a smaller effect on computation but still improves task performance, consistent with its role in limiting ineffective reuse. The remaining ablations reveal distinct roles for applicability, routing, and evolution. Removing Candidate As-

Table 3: Ablation results on ScienceWorld under the task-family-grouped protocol.

Method	Score ↑	Fresh Calls ↓	Tokens (K) ↓
CWME (ours)	45.39 ± 0.60	16.30 ± 0.70	56.00 ± 1.80
w/o Situational Knowledge	42.93 ± 0.43	23.77 ± 0.31	77.37 ± 0.78
w/o Action Trajectory Experience	38.58 ± 0.43	35.00 ± 1.95	127.75 ± 2.83
w/o Failure-Aware Knowledge	43.97 ± 0.31	15.70 ± 0.82	55.37 ± 2.00
w/o Candidate Assessment	43.06 ± 0.38	15.70 ± 0.66	55.53 ± 2.61
w/o Decision Routing	42.95 ± 0.33	18.64 ± 1.57	62.37 ± 2.81
w/o Online Evolution	41.16 ± 0.73	19.83 ± 0.23	66.53 ± 0.31

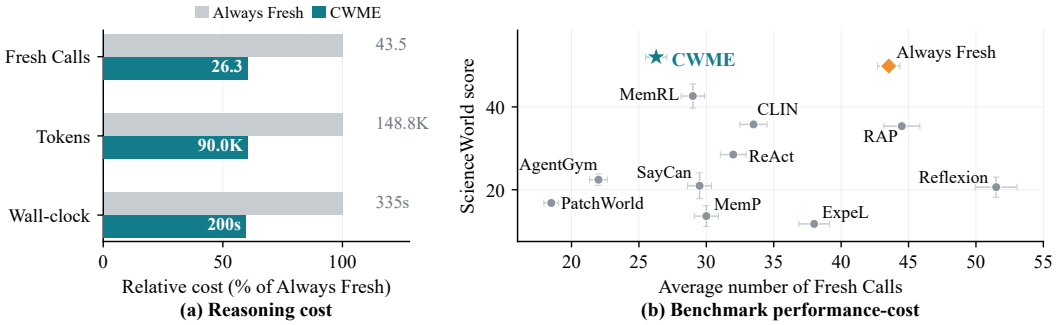


Figure 4: Reasoning efficiency on ScienceWorld: (a) relative reasoning cost and (b) performance versus full-reasoning calls.

sessment weakens selective reuse, showing that retrieval relevance alone is insufficient for deciding whether previous experience should guide the current action. Replacing the learned decision router with a fixed rule-based policy reduces task performance and increases reliance on fresh reasoning, demonstrating the benefit of learned reasoning allocation. Disabling Online Evolution also degrades performance, indicating that accumulated world knowledge should be revised as new interaction outcomes become available. Together, these results support the complementary roles of structured world knowledge, applicability-aware reuse, adaptive routing, and continual knowledge refinement. Additional ablations are provided in Appendix F.

5.6 Reasoning Efficiency

To assess whether CWME improves task performance without excessive reasoning cost, we evaluate its efficiency on the full ScienceWorld benchmark. Figure 4(a) shows that CWME reduces full-reasoning calls, token usage, and wall-clock time compared with Always Fresh. Figure 4(b) further shows that this reduction does not come at the cost of task performance, as several methods use comparable or greater large-model computation while achieving lower scores. CWME therefore provides a more favorable performance–cost trade-off by selectively invoking fresh reasoning and reusing previous experience when appropriate. This behavior is consistent with its design of reserving expensive reasoning for states in which prior experience provides insufficient support. Complete statistics are provided in Appendix G.

6 Conclusion

In this paper, we propose **CWME**, a framework for continual world model evolution via Interaction Experience Learning. CWME consists of three main components: Knowledge-Informed Trajectory Generation adapts historical experience to the current state, Experience-Guided Decision Routing allocates reasoning effort based on current-state applicability, and Evolving Interaction Experience Updating continually refines world knowledge from interaction outcomes. Extensive experiments on ScienceWorld and ALFWorld demonstrate that CWME outperforms all evaluated baselines while reducing reliance on full reasoning, validating the effectiveness of continual world model evolution and applicability-aware experience reuse.

AI USE STATEMENT

Generative AI tools were used to assist with language refinement. All AI-assisted content was reviewed and verified by the authors. The research methodology, experimental design, reported results, and final conclusions were determined by the authors, who take full responsibility for the content of this work.

REPRODUCIBILITY STATEMENT

We provide implementation settings, evaluation protocols, supervision construction, and methodological details in the main paper and appendix. All experiments follow the specified configurations and evaluation metrics, with consistent initialization and interaction settings across controlled experiments. We further record task outcomes, routing decisions, model usage, and world model updates to support reproducibility and further analysis. The source code is provided in the supplementary material.

REFERENCES

Jiaxin Bai, Yue Guo, Yifei Dong, Jiaxuan Xiong, Tianshi Zheng, Yixia Li, Tianqing Fang, Yufei Li, Yisen Gao, Haoyu Huang, Zhongwei Xie, Hong Ting Tsang, Zihao Wang, Lihui Liu, Jeff Z. Pan, and Yangqiu Song. PatchWorld: Gradient-Free Optimization of Executable World Models for Agent Environments, 2026. URL https://arxiv.org/abs/2605.30880.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring Agent Procedural Memory. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17490–17502. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.866. URL https://aclanthology.org/2026.findings-acl.866/.

Tao Feng, Yexin Wu, Guanyu Lin, and Jiaxuan You. Graph World Model. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 16931–16955. PMLR, 2025. URL https://proceedings.mlr.press/v267/feng25p.html.

Tongtong Feng, Xin Wang, Zekai Zhou, Ren Wang, Yuwei Zhan, Guangyao Li, Qing Li, and Wenwu Zhu. EvolvingAgent: Curriculum Self-evolving Agent with Continual World Model for Long-Horizon Tasks, 2026. URL https://arxiv.org/abs/2502.05907.

Haotian Fu, Yixiang Sun, Michael Littman, and George Konidaris. Knowledge Retention in Continual Model-Based Reinforcement Learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 17832–17851. PMLR, 2025. URL https://proceedings.mlr.press/v267/fu25f.html.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. Reasoning with Language Model is Planning with World Model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 8154–8173. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.507. URL https://aclanthology.org/2023.emnlp-main.507/.

Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander T. Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu, Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Nikhil J. Joshi, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. In *Proceedings of the 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 287–318. PMLR, 2023. URL https://proceedings.mlr.press/v205/ichter23a.html.

Bill Yuchen Lin, Yicheng Fu, Karina Yang, Faeze Brahman, Shiyu Huang, Chandra Bhagavatula, Prithviraj Ammanabrolu, Yejin Choi, and Xiang Ren. SwiftSage: A Generative Agent with Fast and Slow Thinking for Complex Interactive Tasks. In *Advances in Neural Information Processing Systems*, volume 36, pp. 23813–23825, 2023. doi: 10.52202/075280-1034. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/4b0eea69deea512c9e2c469187643dc2-Abstract-Conference.html.

Zichen Liu, Guoji Fu, Chao Du, Wee Sun Lee, and Min Lin. Continual Reinforcement Learning by Planning with Online World Models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 38397–38423. PMLR, 2025. URL https://proceedings.mlr.press/v267/liu25p.html.

Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Peter Jansen, Oyvind Tafjord, Niket Tandon, Li Zhang, Chris Callison-Burch, and Peter Clark. CLIN: A Continually Learning Language Agent for Rapid Task Adaptation and Generalization. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=xS6zx1aBI9.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652, 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=0IOX0YcCdTn.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. ScienceWorld: Is Your Agent Smarter than a 5th Grader? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11279–11298. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.emnlp-main.775. URL https://aclanthology.org/2022.emnlp-main.775/.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent Workflow Memory. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 63897–63911. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25bx.html.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Xin Guo, Dingwen Yang, Chenyang Liao, Wei He, Songyang Gao, Lu Chen, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yu-Gang Jiang. AgentGym: Evaluating and Training Large Language Model-based Agents across Diverse Environments. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27914–27961. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.1355. URL https://aclanthology.org/2025.acl-long.1355/.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Shengtao Zhang, Jiaqian Wang, Ruiwen Zhou, Junwei Liao, Yuchen Feng, Zhuo Li, Yujie Zheng, Weinan Zhang, Ying Wen, Zhiyu Li, Feiyu Xiong, Yutao Qi, Bo Tang, and Muning Wen. MemRL: Self-Evolving Agents via Runtime Reinforcement Learning on Episodic Memory, 2026a. URL https://arxiv.org/abs/2601.03192.

Xuan Zhang, Wenxuan Zhang, See-Kiong Ng, and Yang Deng. Self-Evolving World Models for LLM Agent Planning, 2026b. URL https://arxiv.org/abs/2606.30639.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM Agents Are Experiential Learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://ojs.aaai.org/index.php/AAAI/article/view/29936.

APPENDIX

A ADDITIONAL METHOD DETAILS

This appendix provides implementation details for trajectory construction, applicability learning, decision routing, and online world model evolution. The main text describes the mechanisms, while major fixed implementation choices are summarized in Appendix A.5.

A.1 RETRIEVAL, ALIGNMENT, AND GROUNDING

The retrieval, alignment, and grounding operations in Section 4.2 are implemented without an LLM.

Situational-Knowledge Retrieval. The Situational-Knowledge retriever constructs a query from the current task, observation, and working memory and searches the structured entity/state/action graph. Candidate relation e is ranked by

$$R(e) = w_{\text{ent}} S_{\text{entity}}(e) + w_{\text{conf}} C_e + w_{\text{state}} S_{\text{state}}(e), \tag{16}$$

where S_{entity} measures overlap between query content and relation endpoints, C_e denotes relation confidence, and S_{state} measures state/action compatibility. Relations below the configured confidence threshold are removed, and only a bounded number of top-ranked facts or transitions are returned.

Trajectory Retrieval. Action Trajectory Experience candidates are retrieved from the corresponding task/signature bucket after retired or non-reusable records are removed. For trajectory experience ξ_i, the contextual score is

$$\begin{aligned} s_{\text{ctx}}(\xi_i) = w_{\text{act}} s_{\text{act}} + w^{\tau}_{\text{state}} s_{\text{state}} + w_{\text{trans}} s_{\text{trans}} \\ + w_{\text{task}} s_{\text{task}} + w_{\text{traj}} c_i, \end{aligned} \tag{17}$$

where

$$c_i = C(\xi_i) \tag{18}$$

denotes the current composite trajectory confidence. The remaining terms measure action support, state/precondition overlap, transition compatibility, and task compatibility. Online provenance can provide only bounded priority in ties or near-ties and cannot override a substantially better current-state match. The composite confidence $C(\xi_i)$ combines task-outcome statistics with reuse performance when reuse history is available, whereas ρ_i denotes the reuse-specific historical reliability defined below.

Progress Alignment. Align is not a learned similarity module. The stage cursor is an episode-local runtime variable indexed by the stable lineage of each retrieved trajectory, rather than a writable field shared by all episodes. It is reset at the beginning of each episode and remains associated with the same lineage when a frozen training record acquires an online copy. The cursor advances when the selected historical action is actually executed and produces valid reward progress or an enabling transition. Alignment can additionally resume at a later stage when current observations verify that stage's prerequisites and the observed effects of every intervening historical stage. A prior action attempt or the mere availability of a later command is insufficient to skip a stage. When an online revision changes the ordered action sequence, the recorded completed-action prefix is checked against the revised sequence, and the cursor is reduced to the verified common prefix.

Trajectory construction uses the look-ahead horizon H_{traj}, while Candidate Assessment uses the shorter context horizon H_{app}. Their fixed values are reported in Table 4.

Grounding. $\mathcal{G}$ is deterministic and uses the retrieved Situational Knowledge and Failure-Aware Knowledge to ground each aligned trajectory in the current state. For each trajectory ξ_i and its aligned stage j_i, the retrieved Failure-Aware Knowledge is matched against the current state, trajectory lineage, and aligned stage. Candidate-specific constraints are preferred, with generic state/action constraints used when no matching candidate-specific entry is available.

Grounding first accepts an exact currently admissible command that satisfies the applicable knowledge constraints. Otherwise, structured grounding preserves the parsed action family, arity, and ordered semantic roles. Manipulation roles require compatible entity identity, while navigation permits only controlled surface-form aliases. Frozen training trajectory records may contain task roles that are rebound to the current target without modifying the stored record.

For ALFWorld, instance-specific household commands can be abstracted across object identifiers and mapped back to a currently admissible command. Situational-Knowledge `requires`, `precedes`, and `causes` relations are checked during grounding. Because grounding can instantiate concrete action arguments, the matched Failure-Aware constraints are checked again against the grounded command before execution. This prevents failures associated with unrelated trajectory stages from suppressing an otherwise compatible candidate.

Retrieval, alignment, and grounding invoke no LLM. They therefore contribute no LLM calls or logged token usage, although the time spent on these operations is included in wall-clock measurements.

A.2 Candidate Assessment

Input Construction. The Candidate Assessment uses a frozen sentence encoder E. For each decision state, the state text x_t^s contains the task, task identifier, current observation, inventory or agent state, and recent action history.

The trajectory text $x_{t,i}^\tau$ contains the grounded current stage, preconditions, expected transitions, neighboring stages, and the short-horizon applicability context. Both embeddings are ℓ_2 normalized:

$$\mathbf{e}_t^s = E(x_t^s), \qquad \mathbf{e}_{t,i}^\tau = E(x_{t,i}^\tau). \tag{19}$$

The pairwise representation is

$$\mathbf{h}_{t,i} = \left[\mathbf{e}_t^s; \mathbf{e}_{t,i}^\tau; |\mathbf{e}_t^s - \mathbf{e}_{t,i}^\tau|; \mathbf{e}_t^s \odot \mathbf{e}_{t,i}^\tau\right]. \tag{20}$$

A lightweight prediction head maps $\mathbf{h}_{t,i}$ to $p_{t,i}^{\text{app}}$. The sentence encoder remains frozen.

Outcome-Grounded Supervision. The supervision pool is broader than the inference candidate set $\mathcal{C}_t$: candidates rejected by explicit semantic constraints may be retained as negative training examples even though they do not enter $\mathcal{C}_t$ at inference time.

Training labels are constructed from executed candidate outcomes rather than treating every unselected candidate as negative. A selected candidate receives a positive label only when its reuse is valid and produces reward progress, enabling progress, or successful subgoal completion. Failed or invalid executed reuse produces a negative label.

Candidates rejected by explicit semantic conflicts can also provide negative examples. These conflicts include task-family mismatch, target-role mismatch, Situational-Knowledge rejection, and Failure-Aware rejection. Ambiguous no-op outcomes and plausible but unexecuted candidates remain unlabeled and are skipped. If lightweight verification rejects a historical candidate, the ensuing fresh-reasoning action is not attributed to that candidate and does not supply a positive reuse label. Because explicit hard-conflict negatives can appear in the labeled pool even though they are removed before online routing, held-out Assessment metrics characterize the complete labeled test pool rather than only candidates surviving inference-time filters.

Reward, enabling-transition, and subgoal-completion signals are used only to construct supervision. They are not provided to the Candidate Assessment as evaluation-time input features.

For a labeled sample (t, i),

$$\mathcal{L}_{\text{BCE}} = -y_{t,i}^{\text{app}} \log p_{t,i}^{\text{app}} - (1 - y_{t,i}^{\text{app}}) \log(1 - p_{t,i}^{\text{app}}). \tag{21}$$

When positive and negative candidates occur in the same decision group, CWME also applies

$$\mathcal{L}_{\text{rank}} = \max\left(0, \gamma - p_{t,+}^{\text{app}} + p_{t,-}^{\text{app}}\right), \tag{22}$$

and optimizes

$$\mathcal{L}_{\text{app}} = \mathcal{L}_{\text{BCE}} + \lambda \mathcal{L}_{\text{rank}}. \tag{23}$$

Optimization and Evaluation Protocol. Training and validation are split by complete episodes rather than individual state–candidate pairs. The applicability threshold is selected only on validation episodes using the configured search range and selection criterion. The selected threshold is stored with the applicability checkpoint and remains fixed during evaluation. A separate episode-disjoint test split is reserved for final applicability evaluation and is not used for fitting, checkpoint selection, or threshold selection.

The trained scorer and encoder remain fixed during formal evaluation. Architecture, optimization, and threshold-selection settings are summarized in Table 4.

A.3 DECISION ROUTER

Routing Features. The router uses

$$\mathbf{r}_t = [p_t^{\text{app}}, c_t^{\text{traj}}, \rho_t^{\text{reuse}}, v_t^{\text{act}}, \bar{f}_t^{\text{recent}}, h_t^{\text{sit}}, c_t^{\text{cross}}, l_t^{\text{plateau}}, d_t^{\text{gain}}, m_t^{\text{app}}]. \tag{24}$$

These terms represent candidate applicability, composite trajectory confidence, reuse-specific historical reliability, action validity, recent-failure context, Situational-Knowledge support, cross-variation consistency, interaction stagnation, distance from recent progress, and the applicability margin between leading candidates.

When the selected trajectory has no reuse history, ρ_t^{reuse} uses the configured default ρ_0. This default affects the routing feature only; it is not blended into trajectory confidence in the absence of reuse observations.

Feature normalization and default encodings are fixed before formal evaluation and are summarized in Table 5. Explicit hard conflicts are handled before learned allocation. The corresponding validity and failure-related features are retained for consistency with logged states that reach the learned router.

Routing Modes. DIRECTREUSE directly executes the first grounded action without an additional LLM reasoning call. VERIFIEDREUSE invokes Qwen3.5-4B as a lightweight verifier of the grounded candidate under the current task and interaction context. The verifier may approve only the exact selected action when it remains admissible. A rejection, malformed response, or response proposing another action triggers immediate FRESHREASONING from the live task and observed state, without supplying the selected trajectory as its execution decision. Such a step is attributed to the route that actually produces the executed action, while both the verification call and any subsequent fresh-reasoning call contribute to their corresponding model-call and token counts. FRESHREASONING invokes the full reasoning procedure without relying on the candidate as the execution decision. The verifier model is fixed during formal evaluation.

Counterfactual Route Supervision. Training labels are built from forced DIRECTREUSE, VERIFIEDREUSE, and FRESHREASONING runs under matched decision conditions. The forced runs use the same random seed, episode schedule, and initial world knowledge. world model writes are frozen while collecting counterfactual outcomes.

A decision state is joined across the three runs only when its episode position, decision context, selected trajectory record and stage, trajectory context, Failure-Aware context, and routing features agree. Feature equality is checked using the fixed numerical tolerance reported in Table 5.

A route is considered successful, i.e., $S_t(z) = 1$, only when that route itself produces reward progress, an enabling transition, or successful subgoal completion in the matched counterfactual interaction. These progress indicators are used only for offline route-label construction and are not provided as router inputs during evaluation.

States whose independently executed forced runs diverge are excluded from matched route supervision rather than converted into artificial negative route labels. This procedure compares only states that occur with matching episode position, decision context, selected trajectory and stage, and routing features across the three runs. If VERIFIEDREUSE rejects its candidate and immediately falls back to FRESHREASONING, any subsequent progress is attributed to the executed fresh route, not to VERIFIEDREUSE.

The supervision ordering is

$$\text{DirectReuse} < \text{VerifiedReuse} < \text{FreshReasoning}. \tag{25}$$

Accordingly,

$$(1,1,1) \to \text{DirectReuse}, \qquad (0,1,1) \to \text{VerifiedReuse}, \qquad (0,0,1) \to \text{FreshReasoning}. \tag{26}$$

This ordering represents reasoning intensity rather than measured runtime.

States satisfying

$$(S_D, S_V, S_F) = (0,0,0) \tag{27}$$

have no successful-route label and are excluded from router classification training. They remain part of end-to-end evaluation.

The router is trained with class-weighted cross-entropy:

$$\mathcal{L}_{\text{route}} = -\sum_{z \in \mathcal{R}} w_z y_t(z) \log p_t^{\text{route}}(z). \tag{28}$$

The class weight for route z is computed from the router-training split as

$$w_z = \frac{N_{\text{train}}}{\max(1, |\mathcal{R}| N_z)}, \tag{29}$$

where N_z is the number of training examples assigned to route z and $|\mathcal{R}| = 3$. Thus, for non-empty classes, the weighting reduces to balanced inverse-frequency weighting.

Inference-Time Constraints. The learned router operates only on candidates that survive retrieval and applicability filtering. Explicit hard conflicts, such as missing reusable trajectory structure, hard Situational-Knowledge rejection, or a strongly supported matching failure, override the learned route and invoke FRESHREASONING.

Hard world model constraints are checked again before execution. The formal learned-policy configuration disables fallback to alternative routing policies. If a required learned checkpoint is unavailable or incompatible, the run terminates rather than silently changing the evaluated policy.

Network architecture and optimization settings are reported in Table 5.

A.4 WORLD-KNOWLEDGE REPRESENTATION AND EVOLUTION

Formal evaluation separates immutable training knowledge from writable online deltas:

$$\mathcal{K}_t^m = \mathcal{K}_{\text{train}}^m \oplus \Delta\mathcal{K}_t^m, \qquad m \in \{\text{sit}, \text{traj}, \text{fail}\}, \tag{30}$$

where $\oplus$ denotes the representation-specific combination of a frozen training component and its accumulated online delta.

The three training components remain immutable during formal evaluation. Online interaction modifies only $\Delta\mathcal{K}_t^{\text{sit}}$, $\Delta\mathcal{K}_t^{\text{traj}}$, and $\Delta\mathcal{K}_t^{\text{fail}}$.

The notation $\delta\mathcal{K}_t^m$ in Section 4.4 denotes the step-local information extracted from interaction t. By contrast, $\Delta\mathcal{K}_t^m$ denotes the accumulated online delta after consolidation of preceding interaction outcomes. The combination operator is component-specific: Situational Knowledge jointly reads frozen and online relations with combined episode-support statistics; Action Trajectory Experience uses copy-on-write online revisions that shadow their frozen parents in the effective retrieval view; Failure-Aware Knowledge adds online outcome counts to matching frozen condition records. Consequently, formal online evolution changes effective knowledge used at subsequent decisions without modifying any frozen training artifact.

Situational Knowledge. Situational Knowledge stores typed entity/state/action relations and action-conditioned transitions. Each relation maintains observation and outcome statistics. Its episode confidence is

$$C(e) = \frac{N_{\text{episodes}}(e)}{N_{\text{episodes}}}, \tag{31}$$

while outcome-conditioned success confidence is

$$C_{\text{succ}}(e) = \frac{N_{\text{succ}}(e)}{N_{\text{succ}}(e) + N_{\text{fail}}(e)} \tag{32}$$

when outcome statistics are available.

Repeated compatible observations reinforce an existing relation, while previously unseen supported relations extend $\Delta\mathcal{K}_t^{\text{sit}}$. For single-valued observed state or location slots, a conflicting outcome is recorded with its current validity context as a contextual revision or an alternative condition; it does not globally erase an earlier observation made under another condition. The effective retrieval view combines support for identical frozen and online relations, rejects facts contradicted by explicitly observed current-state values, and filters relations against their recorded contexts when those conditions are observable. When context is unavailable, the implementation retains uncertainty rather than declaring an older relation globally false. Relations such as `requires`, `precedes`, and `causes` also provide feasibility support during trajectory grounding and routing.

Action Trajectory Experience. As in the main text, a trajectory record is represented by

$$\xi_i = (g_i, C_i, A_i, \Gamma_i, \rho_i).$$

The implementation additionally maintains task-outcome statistics, reuse statistics, trajectory confidence, and maintenance metadata used for consolidation and retention.

When reuse history is available, reuse-specific historical reliability is defined as

$$\rho_i = \frac{N_{\text{reuse_succ}}^{(i)}}{N_{\text{reuse}}^{(i)}}, \qquad N_{\text{reuse}}^{(i)} > 0. \tag{33}$$

A new trajectory is compared with records from the corresponding task/signature bucket. Candidate records are compared only within the corresponding task/signature bucket and must share global-condition semantics. Merge compatibility requires both sufficient action-set overlap and order-preserving, state-compatible stage alignment:

$$J(A_i, A_j) \geq \theta_{\text{merge}} \quad \wedge \quad \text{SeqCompat}(\xi_i, \xi_j), \tag{34}$$

where J is Jaccard similarity of canonical action sets. SeqCompat uses constraint-aware longest-common-subsequence alignment, requires alignment coverage of at least θ_{merge} relative to the shorter trajectory, and preserves the relative order of all shared action occurrences. Stages with conflicting preconditions cannot be aligned. During consolidation, indexed stage metadata is transferred only through unambiguous optimal stage correspondences; unmatched or ambiguous source-stage evidence cannot be reassigned to a different occurrence of the same action. A compatible frozen training record is revised through an online copy that shadows the original record in effective retrieval, leaving the frozen source unchanged. The most similar compatible record is updated; otherwise a new online record is created.

Trajectory confidence first summarizes task outcomes as

$$C_0(\xi_i) = \frac{S_i + \omega_{\text{partial}} P_i^{\text{partial}} + \alpha_0}{S_i + P_i^{\text{partial}} + F_i + \alpha_0 + \beta_0}. \tag{35}$$

When reuse history is available, the composite trajectory confidence becomes

$$C(\xi_i) = (1 - \omega_{\text{reuse}}) C_0(\xi_i) + \omega_{\text{reuse}} \rho_i. \tag{36}$$

Otherwise,

$$C(\xi_i) = C_0(\xi_i). \tag{37}$$

Successful, partial, failed, or invalid reuse therefore revises stored statistics once reuse observations become available, while trajectories without reuse history retain their task-outcome confidence.

Here, $C(\xi_i)$ is denoted by c_i during trajectory retrieval and by c_t^{traj} for the selected candidate during routing. When reuse history exists, ρ_i directly represents reuse-specific reliability and corresponds to ρ_t^{reuse} for the selected candidate. When reuse history is absent, only the router feature uses the configured default ρ_0; trajectory confidence remains $C_0(\xi_i)$.

Bounded Retention. Writable online trajectory buckets have bounded capacity. For each trajectory record, CWME first computes

$$q(\xi_i) = w_C C(\xi_i) + w_s s(\xi_i) + w_{\text{status}} q_{\text{status}}(\xi_i) + w_{\text{outcome}} q_{\text{outcome}}(\xi_i) + w_{\text{reuse}} q_{\text{reuse}}(\xi_i) + w_{\text{recent}} q_{\text{recent}}(\xi_i), \tag{38}$$

where the terms summarize trajectory confidence, task quality, status, outcome reliability, reuse reliability, and recency.

After retaining the highest-quality record, future records are selected greedily according to

$$q_{\text{MMR}}(\xi_i) = \alpha_{\text{MMR}} q(\xi_i) + (1 - \alpha_{\text{MMR}}) \left(1 - \max_{\xi_j \in \mathcal{S}} J(F_i, F_j)\right), \tag{39}$$

where F_i is the state/action fingerprint of ξ_i and $\mathcal{S}$ is the already retained set. This balances record quality with state/action diversity.

Failure-Aware Knowledge. Failure-Aware Knowledge contains a frozen training component and a writable online delta. Each entry uses a state-conditioned key derived from task identity, state, action, trajectory source, and stage when available. Candidate-specific lookup uses the aligned trajectory record and stage. After grounding, the concrete action-level condition is checked again. When a source-specific entry is unavailable, lookup can back off to a generic state/action condition.

For an entry with N^- failed or invalid observations and N^+ successful observations, the smoothed failure score is

$$F_{\text{fail}} = \frac{N^- + \alpha_{\text{fail}}}{N^- + N^+ + \alpha_{\text{fail}} + \beta_{\text{fail}}}. \tag{40}$$

Reuse is hard-blocked when

$$N^- \geq n_{\text{fail}} \qquad \text{and} \qquad F_{\text{fail}} \geq \theta_{\text{fail}}. \tag{41}$$

Repeated failures reinforce reuse boundaries, while subsequent successful interactions can weaken matching failure constraints without affecting those associated with unrelated actions. Failure conditions are associated with the stable lineage of historical trajectories and, when available, their aligned stages, states, and actions. This allows copy-on-write trajectory revisions to preserve inherited reuse boundaries.

New failure conditions extend $\Delta\mathcal{K}_t^{\text{fail}}$. The writable online Failure-Aware store has bounded capacity; the frozen training component remains unchanged.

These mechanisms instantiate the four conceptual evolution operations *extend*, *reinforce*, *revise*, and *suppress* in a representation-specific manner.

A.5 Implementation Configuration

The major fixed implementation choices used in the reported experiments are summarized in Tables 4–6. Benchmark-specific interaction-step limits are reported separately in Appendix B.3.

A.6 Closed-Loop Inference

CWME operates in a receding-horizon manner. At each interaction step, CWME retrieves world knowledge, aligns historical trajectories to current progress, and grounds short-horizon trajectory segments by matching the retrieved Failure-Aware Knowledge to each aligned candidate. It then evaluates current-state applicability and selects a reasoning route. Only the current action is executed.

The resulting transition is incorporated into the evolving world model before the next decision. A trajectory segment that is applicable at step t may become invalid after the transition to s_{t+1}. Applicability and routing are therefore recomputed under every newly observed state rather than committing to a retrieved trajectory open-loop.

During formal evaluation, the pretrained encoder, Candidate Assessment, decision router, Qwen3.5-4B verifier, and underlying reasoning model remain fixed. Online updates are restricted to $\Delta\mathcal{K}_t^{\text{sit}}$, $\Delta\mathcal{K}_t^{\text{traj}}$, and $\Delta\mathcal{K}_t^{\text{fail}}$.

Table 4: Implementation configuration for retrieval, trajectory construction, and Candidate Assessment.

Component	Configuration	Value
Situational retrieval	Minimum relation confidence θ_{edge}	0.15
Situational retrieval	Maximum returned facts/transitions B_{sit}	10
Situational retrieval	$(w_{\text{ent}}, w_{\text{conf}}, w_{\text{state}})$	$(0.45, 0.35, 0.20)$
Trajectory retrieval	$(w_{\text{act}}, w^{\tau}_{\text{state}}, w_{\text{trans}}, w_{\text{task}}, w_{\text{traj}})$	$(0.20, 0.40, 0.15, 0.10, 0.15)$
Trajectory retrieval	Runtime query cap `MAX_QUERY`	3
Trajectory construction	Look-ahead horizon H_{traj}	5
Trajectory record filtering	Minimum record / partial scores	$30/20$
Trajectory-context validation	Validation threshold	0.65
Applicability context	Context horizon H_{app}	3
Candidate Assessment	Sentence encoder	`all-MiniLM-L6-v2`
Candidate Assessment	Prediction head	$4d \to 512 \to 128 \to 1$
Candidate Assessment	Hidden-layer dropout	0.10
Candidate Assessment	Ranking weight λ	0.25
Candidate Assessment	Ranking margin γ	0.20
Applicability training	Optimizer	AdamW
Applicability training	Epochs / batch size	20 / 128
Applicability training	Learning rate / weight decay	2×10^{-4} / 10^{-4}
Applicability training	Train/validation split	$80/20$ by complete episode
Applicability threshold	Search range / step	$[0.20, 0.90]$ / 0.01
Applicability threshold	Selection criterion	F1; precision tie-break
Applicability threshold	Evaluation threshold	Validation-selected and stored with checkpoint

Table 5: Implementation configuration for Decision Routing.

Component	Configuration	Value
Routing modes	DIRECTREUSE	No additional LLM call
Routing modes	VERIFIEDREUSE verifier	Qwen3.5-4B
Routing modes	FRESHREASONING	Full reasoning procedure
Router features	Feature dimension	10
Router features	Default reuse reliability ρ_0	0.5
Router features	Situational support encoding	Positive / unavailable / negative $= 1/0.5/0$
Router features	`steps_since_gain` normalization	$\text{clip}(n/10, 0, 1)$
Router features	Applicability-margin range	Clipped to $[0, 1]$
Router	Network architecture	$10 \to 64 \to 32 \to 3$
Router	Dropout	0.10
Counterfactual matching	Feature-vector equality tolerance	10^{-6}
Router training	Optimizer / loss	AdamW / class-weighted cross-entropy
Router training	Class weighting	$N_{\text{train}}/\max(1, 3N_z)$
Router training	Epochs / batch size	30 / 128
Router training	Learning rate / weight decay	3×10^{-4} / 10^{-4}
Router training	Train/validation split	$80/20$ by episode group

Table 6: Implementation configuration for online Knowledge Evolution.

Component	**Configuration**	**Value**
Trajectory evolution	Merge threshold θ_{merge}	0.72
Trajectory confidence	Partial-outcome weight ω_{partial}	0.5
Trajectory confidence	Beta prior (α_0, β_0)	$(1, 1)$
Trajectory confidence	Reuse blend ω_{reuse}	0.40
Trajectory retention	Maximum writable records per source bucket	5
Trajectory retention	$(w_C, w_s, w_{\text{status}}, w_{\text{outcome}}, w_{\text{reuse}}, w_{\text{recent}})$	$(0.36, 0.24, 0.16, 0.12, 0.07, 0.05)$
Trajectory retention	MMR quality/diversity coefficient $\alpha_{\text{MMR}}/(1-\alpha_{\text{MMR}})$	$0.60/0.40$
Failure-Aware	Failure-score prior $(\alpha_{\text{fail}}, \beta_{\text{fail}})$	$(1, 1)$
Failure-Aware	Minimum failures n_{fail}	2
Failure-Aware	Hard-block threshold θ_{fail}	0.67
Failure-Aware	Maximum writable online records	5000

B Additional Experimental Details

B.1 Dataset and Task Details

ScienceWorld. ScienceWorld provides language-based interactive scientific environments in which an agent reasons about entities, physical states, and state-changing actions over multi-step trajectories. The benchmark contains 30 task families.

For controlled ScienceWorld experiments, we select five representative task families: `change-the-state-of-matter-of`, `chemistry-mix`, `grow-plant`, `inclined-plane-friction-unnamed-surfaces`, and `use-thermometer`. The continual world model evolution experiment follows a fixed stream of five interaction cycles, with one episode from each task family per cycle, yielding 25 episodes per seed. In contrast, ablation, routing, and sensitivity experiments follow a task-family-grouped protocol, executing all scheduled variations of one task family before proceeding to the next. World knowledge is retained across episodes in both protocols and across cycles in the continual evolution experiment, except in variants that explicitly reset or freeze knowledge. Within each protocol and seed, compared variants share the same initial world knowledge, task instances, and execution order. For the continual evolution experiment, episode membership and cycle boundaries are also matched across variants to ensure directly comparable within-cycle results. The exact schedules are provided with the experiment configurations.

ALFWorld. ALFWorld contains language-guided embodied tasks involving navigation, object manipulation, and multi-stage completion. We evaluate six task families:

01 `pick_and_place_simple`

02 `look_at_obj_in_light`

03 `pick_clean_then_place_in_recep`

04 `pick_heat_then_place_in_recep`

05 `pick_cool_then_place_in_recep`

06 `pick_two_obj_and_place`

Results are reported separately for the official seen and unseen settings.

B.2 Baseline Details

The evaluated baselines cover complementary interactive-agent paradigms.

- **SayCan** (Ichter et al., 2023) grounds language-model decisions with executable action affordances.
- **ReAct** (Yao et al., 2023) interleaves reasoning and environment interaction.
- **Reflexion** (Shinn et al., 2023) uses interaction feedback to construct reflective information.
- **RAP** (Hao et al., 2023) evaluates prospective reasoning trajectories with model-based planning.
- **AgentGym** (Xi et al., 2025) supports interactive-agent training and evaluation.
- **PatchWorld** (Bai et al., 2026) induces executable world representations from interaction trajectories.
- **ExpeL** (Zhao et al., 2024) extracts reusable knowledge from previous task-solving experience.
- **CLIN** (Majumder et al., 2024) continually accumulates interaction-derived knowledge.
- **MemP** (Fang et al., 2026) maintains persistent reusable task-solving memory.
- **MemRL** (Zhang et al., 2026a) combines memory with feedback-driven decision making.

Table 7: ScienceWorld task families and their corresponding IDs used in the overall evaluation.

ID	Task Family
01	boil
02	change-the-state-of-matter-of
03	chemistry-mix
04	chemistry-mix-paint-secondary-color
05	chemistry-mix-paint-tertiary-color
06	find-animal
07	find-living-thing
08	find-non-living-thing
09	find-plant
10	freeze
11	grow-fruit
12	grow-plant
13	identify-life-stages-1
14	identify-life-stages-2
15	inclined-plane-determine-angle
16	inclined-plane-friction-named-surfaces
17	inclined-plane-friction-unnamed-surfaces
18	lifespan-longest-lived
19	lifespan-longest-lived-then-shortest-lived
20	lifespan-shortest-lived
21	measure-melting-point-known-substance
22	measure-melting-point-unknown-substance
23	melt
24	mendelian-genetics-known-plant
25	mendelian-genetics-unknown-plant
26	power-component
27	power-component-renewable-vs-nonrenewable-energy
28	test-conductivity
29	test-conductivity-of-unknown-substances
30	use-thermometer

Controlled Comparison Protocol. Each baseline follows the setting of its original paper or released implementation, including the method-specific prompt structure, reasoning or planning procedure, reflection policy, memory/retrieval mechanism, and method-specific hyperparameters. The only backbone substitution is to Qwen3.5-9B for the LLM reasoning model. Shared benchmark-level quantities are harmonized across methods within the same evaluation: all methods receive the same dataset split and episode schedule, and key budget parameters such as the maximum interaction/training steps and evaluation limits are unified. Methods that natively maintain cross-episode memory are evaluated on the same sequential deployment stream; methods whose original design has no cross-episode adaptation are left without it rather than being augmented with additional memory. Consequently, the comparison preserves the algorithms' original mechanisms while controlling the backbone and shared interaction budget.

WorldEvolver is discussed as closely related work but is not included in the quantitative comparison because a directly reproducible implementation compatible with the evaluated benchmarks was not available.

B.3 Implementation Details

CWME uses Qwen3.5-9B for FreshReasoning and Qwen3.5-4B for VerifiedReuse. DirectReuse executes the selected grounded action without an additional LLM call. The appli-

cability module uses the frozen SentenceTransformer `all-MiniLM-L6-v2` and trains only the lightweight prediction head. During evaluation, all language-model parameters, the sentence encoder, Candidate Assessment, and decision router remain frozen. The fixed CWME architecture, training, retrieval, routing, and knowledge-maintenance configurations are summarized in Tables 4–6.

Local Language-Model Inference. The reported implementation runs local language models through the HuggingFace `text-generation` pipeline with automatic device mapping. Qwen3-family models are loaded in `bfloat16`; FlashAttention-2 is used when available and otherwise the implementation falls back to PyTorch SDPA. The Qwen chat template is invoked with internal thinking disabled. All formal calls use temperature zero, yielding greedy decoding rather than stochastic sampling. The implementation caps the encoded prompt at 8192 tokens after reserving the requested output budget.

Hardware and Runtime Environment. All experiments were conducted on NVIDIA H200 GPUs with approximately 140 GB of GPU memory. Each run was allocated 10 CPU cores and 100 GB of system memory on compute nodes equipped with Intel Xeon Platinum 8568Y+ processors. The environment uses Ubuntu 22.04.3 LTS, NVIDIA driver 575.57.08, and CUDA 12.9.

Training-Split Knowledge Initialization and Artifact Separation. The initial world knowledge used in the reported experiments is constructed exclusively from ordinary agent–environment interaction trajectories collected on the benchmark training split. Observed task contexts, environment states, executed actions, state transitions, successful outcomes, and failed or non-progressing outcomes are compiled offline into the initial Situational Knowledge, Action Trajectory Experience, and Failure-Aware Knowledge artifacts, denoted $\mathcal{K}^{\mathrm{sit}}_{\mathrm{train}}$, $\mathcal{K}^{\mathrm{traj}}_{\mathrm{train}}$, and $\mathcal{K}^{\mathrm{fail}}_{\mathrm{train}}$, respectively. No evaluation-split trajectory or test-time interaction is used to construct the initial knowledge used in the reported results.

Training artifacts and online knowledge are separated as

$$\mathcal{K}^{\mathrm{sit}}_t = \mathcal{K}^{\mathrm{sit}}_{\mathrm{train}} \oplus \Delta\mathcal{K}^{\mathrm{sit}}_t, \qquad \mathcal{K}^{\mathrm{traj}}_t = \mathcal{K}^{\mathrm{traj}}_{\mathrm{train}} \oplus \Delta\mathcal{K}^{\mathrm{traj}}_t, \tag{42}$$

$$\mathcal{K}^{\mathrm{fail}}_t = \mathcal{K}^{\mathrm{fail}}_{\mathrm{train}} \oplus \Delta\mathcal{K}^{\mathrm{fail}}_t. \tag{43}$$

Only the corresponding online deltas are writable and evolve from interaction outcomes.

Benchmark-Specific Runtime Limits. The reported ScienceWorld formal controlled protocols use at most 100 interaction steps per episode, while ALFWorld uses at most 50. All remaining fixed CWME runtime and training hyperparameters are reported in Tables 4–6. All controlled variants within a seed use the same initial training artifacts and episode schedule. Across baselines, the shared maximum-step and evaluation budgets are standardized as described in Section B.2.

B.4 Assessment and Routing Supervision

Assessment and routing supervision is constructed from training-split interactions only. For Assessment, the development portion is divided into disjoint training and validation partitions by whole episode using an 80/20 split, while a separate episode-disjoint test split is reserved exclusively for final Assessment evaluation. Router examples are split by episode group for training and validation. Neither the Assessment test split nor benchmark evaluation interactions are used to train either learned module.

For Assessment, only executed DirectReuse/VerifiedReuse outcomes or explicit hard semantic conflicts receive labels; ambiguous or merely unselected candidates are skipped. The resulting label represents current-decision Assessment rather than eventual success of the complete retrieved trajectory. The classification threshold is chosen using validation episodes and stored with the checkpoint. Final Assessment metrics are then computed once on the independent test split with this threshold fixed.

For routing, forced Direct/Verify/Full counterfactuals are collected under the same frozen world model state and matched decision context. Among routes that achieve valid task progress, the route with the lowest predefined reasoning level provides the classification target. States in which all three

routes fail are logged and excluded from route-classification supervision without being removed from end-to-end evaluation.

B.5 Evaluation Metrics

ScienceWorld uses environment task score, and ALFWorld uses task success rate. Candidate Assessment is evaluated using precision, recall, F1, AUROC, and AUPRC. Routing quality is evaluated using macro-F1 and class-wise precision, recall, and F1. Reasoning cost is measured using Fresh Calls, Other Calls, token usage, and wall-clock time. Fresh Calls count invocations of the full reasoning model, regardless of model size. Other Calls count lightweight verification and auxiliary language-model invocations, including pattern verification when enabled. Direct reuse, retrieval, alignment, grounding, and environment interactions are excluded from both call counts. Token usage is measured using the tokenizer of each corresponding language model. For each invocation, we record the actual number of input tokens after applying the chat template and prompt truncation, together with the number of generated output tokens. Total token usage is obtained by summing input and output token counts across all logged full-reasoning, verification, and auxiliary language-model calls. Token counts are reported in thousands. Wall-clock time includes retrieval, alignment, grounding, environment interaction, and model execution. Knowledge evolution is summarized using world-knowledge sizes and update events.

C Task-Wise Results

C.1 ScienceWorld

Table 8 reports task-wise ScienceWorld performance. Results vary substantially across task families, indicating that the aggregate comparison is not driven by a single category. CWME performs strongly on several state-transition and entity-reasoning tasks, while some multi-stage tasks remain challenging.

C.2 ALFWorld Seen

Table 9 reports task-wise results on the seen split. Performance varies across embodied task types. CWME is particularly effective on tasks involving state-changing interaction, while some object-placement tasks remain more competitive.

C.3 ALFWorld Unseen

Table 10 reports results on unseen tasks. CWME remains strong when task instances differ from those used during training, especially when success depends on adapting actions to changed object states. Performance nevertheless remains heterogeneous across task families.

D Additional world model Evolution Analysis

D.1 Cycle-Wise Performance

Table 11 reports complete cycle-wise results. Every cycle contains one episode from each of the five task families, so the task-family composition is identical across cycles. All variants use exactly the same task instances, episode order, and cycle boundaries within each seed. Because individual instances and environment states can differ between cycles, we do not interpret the absolute trajectory across Cycle 1–5 as a monotonic learning curve. Instead, the matched within-cycle comparison evaluates how the different knowledge-evolution mechanisms behave under the same continual interaction stream.

D.2 Knowledge Evolution

Table 12 characterizes how structured world knowledge changes during interaction. CWME performs additions together with merges, revisions, suppressions, and evictions. These operations show

Table 8: Task-wise performance on ScienceWorld. Task IDs correspond to Table 7. Results are mean ± standard deviation over three runs. The best and second-best overall results are highlighted in bold and underlined, respectively.

ID	ReAct	RAP	ExpeL	CLIN	MemP	MemRL	CWME
01	0.8 ± 1.1	0.6 ± 0.8	2.5 ± 0.8	16.6 ± 18.7	15.0 ± 18.7	0.0 ± 0.0	57.4 ± 0.0
02	0.2 ± 0.3	2.4 ± 0.8	0.7 ± 0.3	15.8 ± 17.6	15.0 ± 18.7	0.0 ± 0.0	71.8 ± 0.0
03	40.6 ± 22.7	32.0 ± 3.1	32.0 ± 14.0	24.0 ± 6.5	14.2 ± 12.4	86.6 ± 17.7	26.9 ± 0.9
04	32.0 ± 7.5	58.0 ± 37.0	21.0 ± 11.6	16.0 ± 1.7	26.0 ± 5.3	100.0 ± 0.0	26.0 ± 6.9
05	12.0 ± 2.3	14.0 ± 2.2	9.2 ± 5.8	5.2 ± 6.9	10.4 ± 2.5	80.0 ± 26.5	7.8 ± 2.7
06	20.2 ± 1.4	55.0 ± 39.7	12.5 ± 0.0	25.0 ± 25.0	5.0 ± 6.6	30.2 ± 17.5	100.0 ± 0.0
07	20.2 ± 1.4	48.4 ± 24.0	17.1 ± 3.8	40.0 ± 36.1	18.4 ± 8.2	35.2 ± 21.2	100.0 ± 0.0
08	81.6 ± 5.1	80.0 ± 6.6	13.7 ± 6.9	88.2 ± 8.1	44.8 ± 34.3	100.0 ± 0.0	75.0 ± 0.0
09	36.8 ± 27.4	15.2 ± 6.9	18.4 ± 2.1	0.0 ± 0.0	18.6 ± 2.1	25.0 ± 33.1	69.8 ± 17.5
10	0.0 ± 0.0	0.0 ± 0.0	3.5 ± 2.9	41.4 ± 22.3	16.4 ± 21.7	0.0 ± 0.0	68.8 ± 18.7
11	26.2 ± 4.0	23.0 ± 1.8	7.1 ± 2.6	30.0 ± 20.4	8.6 ± 6.4	17.6 ± 12.2	15.8 ± 0.3
12	9.6 ± 0.9	30.8 ± 9.9	1.8 ± 0.5	18.6 ± 10.6	2.6 ± 1.7	63.2 ± 10.5	34.2 ± 0.5
13	43.4 ± 4.2	39.2 ± 3.8	2.2 ± 0.6	1.0 ± 1.3	9.2 ± 8.1	47.6 ± 4.5	28.2 ± 10.1
14	8.0 ± 0.0	8.0 ± 0.0	4.6 ± 1.4	2.4 ± 3.2	12.8 ± 5.9	25.0 ± 25.0	18.2 ± 8.2
15	23.0 ± 7.2	30.0 ± 0.0	4.8 ± 3.0	60.0 ± 17.3	11.0 ± 5.4	40.0 ± 0.0	46.0 ± 15.6
16	22.0 ± 1.8	29.0 ± 1.7	8.0 ± 2.0	100.0 ± 0.0	9.0 ± 4.9	36.0 ± 5.3	31.0 ± 29.9
17	21.0 ± 5.3	7.0 ± 2.6	12.0 ± 3.4	54.0 ± 15.5	11.0 ± 5.4	40.0 ± 0.0	40.0 ± 0.0
18	60.0 ± 13.2	70.0 ± 18.0	46.2 ± 1.1	50.0 ± 0.0	30.0 ± 18.9	40.0 ± 36.1	70.0 ± 8.7
19	43.0 ± 13.2	56.4 ± 22.6	31.4 ± 2.2	20.0 ± 26.5	10.0 ± 13.2	40.0 ± 36.1	33.2 ± 24.8
20	70.0 ± 26.5	80.0 ± 8.7	42.5 ± 7.8	60.0 ± 13.2	30.0 ± 26.5	60.0 ± 17.3	60.0 ± 17.3
21	6.8 ± 0.3	38.8 ± 0.3	20.0 ± 26.5	74.6 ± 21.3	1.0 ± 1.3	0.0 ± 0.0	39.0 ± 0.5
22	19.4 ± 17.7	24.8 ± 24.9	13.1 ± 20.5	35.0 ± 30.4	0.0 ± 0.0	20.0 ± 26.5	95.0 ± 6.6
23	0.6 ± 0.8	0.0 ± 0.0	1.4 ± 0.7	16.6 ± 18.2	8.2 ± 9.8	0.0 ± 0.0	56.8 ± 11.9
24	17.0 ± 0.0	17.0 ± 0.0	8.6 ± 9.3	80.0 ± 34.6	9.2 ± 3.2	12.6 ± 4.1	100.0 ± 0.0
25	17.0 ± 0.0	18.4 ± 1.4	6.7 ± 8.8	80.0 ± 26.5	9.8 ± 2.9	10.0 ± 7.9	13.8 ± 4.2
26	39.4 ± 17.6	100.0 ± 0.0	2.0 ± 0.9	68.6 ± 5.0	40.6 ± 9.7	88.6 ± 2.9	70.4 ± 9.8
27	10.6 ± 14.0	26.8 ± 16.7	1.2 ± 1.1	23.2 ± 15.3	20.2 ± 13.7	59.4 ± 17.5	42.0 ± 1.3
28	74.8 ± 5.2	75.8 ± 3.1	6.0 ± 3.8	6.0 ± 1.3	2.0 ± 2.6	50.6 ± 18.2	47.0 ± 17.2
29	89.6 ± 4.5	72.0 ± 2.6	2.5 ± 2.2	17.2 ± 4.6	0.0 ± 0.0	71.2 ± 9.5	62.4 ± 0.7
30	9.0 ± 1.5	8.0 ± 2.6	0.0 ± 0.0	3.8 ± 5.0	0.0 ± 0.0	100.0 ± 0.0	54.1 ± 3.6
Overall	28.49 ± 0.57	35.35 ± 0.45	11.75 ± 0.33	35.77 ± 0.64	13.63 ± 2.53	$\underline{42.63 \pm 2.92}$	$\mathbf{52.02 \pm 0.90}$

Table 9: Task-wise success rates on the ALFWorld seen split. Task IDs correspond to the six task families listed in the ALFWorld benchmark description. Results are mean ± standard deviation over three runs. The best and second-best overall results are highlighted in bold and underlined, respectively.

ID	ReAct	ExpeL	CLIN	MemP	MemRL	CWME
01	0.0 ± 0.0	0.0 ± 0.0	100.0 ± 0.0	90.5 ± 4.4	80.0 ± 0.0	62.9 ± 2.9
02	69.2 ± 7.7	46.2 ± 7.7	69.2 ± 7.7	92.3 ± 0.0	84.6 ± 7.7	76.9 ± 0.0
03	0.0 ± 0.0	3.7 ± 0.0	0.0 ± 0.0	77.8 ± 9.8	63.0 ± 3.7	51.9 ± 0.0
04	6.2 ± 0.0	6.2 ± 0.0	0.0 ± 0.0	18.8 ± 0.0	37.5 ± 0.0	68.8 ± 6.2
05	8.0 ± 0.0	8.0 ± 0.0	40.0 ± 0.0	54.7 ± 6.1	56.0 ± 6.9	52.0 ± 0.0
06	0.0 ± 0.0	0.0 ± 0.0	79.2 ± 4.2	12.5 ± 0.0	33.3 ± 0.0	50.0 ± 4.2
Overall	13.91 ± 1.28	10.68 ± 1.28	48.07 ± 1.74	57.75 ± 1.90	$\underline{59.07 \pm 2.22}$	$\mathbf{60.40 \pm 1.96}$

that world model evolution modifies existing knowledge rather than simply enlarging an append-only memory.

E Additional Assessment and Routing Analysis

E.1 Applicability Evaluation Scope

The applicability results in Table 2 are computed on the independent episode-disjoint test split described in Section B.4. The threshold is selected only on validation episodes and is fixed before

Table 10: Task-wise success rates on the ALFWorld unseen split. Task IDs correspond to the six task families listed in the ALFWorld benchmark description. Results are mean ± standard deviation over three runs. The best and second-best overall results are highlighted in bold and underlined, respectively.

ID	ReAct	ExpeL	CLIN	MemP	MemRL	CWME
01	0.0 ± 0.0	0.0 ± 0.0	58.3 ± 0.0	54.2 ± 4.2	45.8 ± 4.2	41.7 ± 4.2
02	63.0 ± 12.8	88.9 ± 0.0	72.2 ± 0.0	63.0 ± 12.8	83.3 ± 0.0	77.8 ± 0.0
03	3.2 ± 0.0	3.2 ± 0.0	51.6 ± 3.2	63.4 ± 4.9	54.8 ± 11.2	54.8 ± 8.5
04	4.3 ± 0.0	4.3 ± 0.0	30.4 ± 0.0	26.1 ± 4.3	52.2 ± 11.5	91.3 ± 0.0
05	4.8 ± 0.0	4.8 ± 0.0	28.6 ± 4.8	28.6 ± 0.0	52.4 ± 0.0	90.5 ± 0.0
06	5.9 ± 0.0	0.0 ± 0.0	88.2 ± 0.0	29.4 ± 0.0	17.6 ± 0.0	52.9 ± 0.0
Overall	13.53 ± 2.14	16.87 ± 0.00	$\underline{54.90 \pm 1.33}$	44.11 ± 2.52	51.03 ± 1.06	$\mathbf{68.17 \pm 1.70}$

Table 11: Cycle-wise continual world model evolution on ScienceWorld. Each cycle contains one episode from each of the five controlled task families, yielding 25 episodes per seed. Results are averaged over three seeds. The best and second-best results are highlighted in bold and underlined, respectively.

Setting	Cycle 1	Cycle 2	Cycle 3	Cycle 4	Cycle 5	Avg.
Frozen World Model	45.93	41.80	37.60	35.47	35.00	39.16
Reset-per-Episode	45.93	42.40	42.13	36.67	35.60	40.55
Append-Only Evolution	45.93	<u>43.00</u>	<u>43.73</u>	<u>38.80</u>	35.20	<u>41.33</u>
Trajectory Evolution	45.93	41.80	39.80	35.47	35.00	39.60
Situational Evolution	45.93	<u>43.00</u>	40.13	38.67	<u>38.53</u>	41.25
CWME (ours)	45.93	**43.23**	**43.90**	**42.03**	**42.50**	**43.52**

test evaluation. Because plausible but unexecuted candidates are intentionally left unlabeled, these metrics evaluate outcome-grounded decision applicability on the labeled candidate population, not retrieval quality over every generated candidate. We use the same test protocol and logs for all applicability baselines so that similarity, historical reliability, and the learned scorer are compared on identical labeled examples.

E.2 Router Prediction

Table 13 reports prediction performance of the learned router on matched held-out states with a valid successful-route label. States for which all three routes fail have no classification target and are therefore excluded from router-classification metrics, while remaining in the end-to-end task evaluation.

E.3 End-to-End Routing Behavior

Table 14 compares complete routing strategies under the task-family-grouped protocol. Unlike router-classification metrics, end-to-end task evaluation retains all encountered decision states, including states for which none of the three counterfactual routes produces valid progress. Fixed policies form the two extremes, while learned routing uses all three computation paths. The verification route handles cases in which retrieved experience remains useful but requires additional confirmation before execution.

F Additional Ablation Analysis

F.1 Ablation Definitions

w/o Situational Knowledge. Removes Situational Knowledge while retaining the remaining components.

Table 12: World model evolution statistics on the ScienceWorld continual stream, averaged over three seeds.

Statistic	Mean
Initial Situational Knowledge relations	320.00
Final Situational Knowledge relations	412.00
Initial Action Trajectory Experience patterns	25.00
Final Action Trajectory Experience patterns	41.00
Trajectory additions	20.00
Trajectory merges	77.00
Trajectory evictions	4.00
Situational nodes added	46.00
Situational edges added	92.00
Failure-Aware additions	13.00
Failure-Aware revisions	7.00
Failure-Aware suppressions	5.33

Table 13: Decision-routing prediction on labeled held-out states. Results are averaged over three seeds. Macro denotes the unweighted average across the three routing classes.

Metric	Direct	Verify	Fresh	Macro
Precision	0.840	0.720	0.870	0.810
Recall	0.810	0.660	0.910	0.793
F1	0.825	0.689	0.890	0.801

w/o Action Trajectory Experience. Removes reusable historical trajectory structure.

w/o Failure-Aware Knowledge. Removes failure-derived constraints and suppression information.

w/o Candidate Assessment. Removes the learned state-conditioned decision-applicability filtering while retaining trajectory retrieval, grounding, the remaining hard feasibility checks, and the downstream routing mechanism.

w/o Decision Routing. Replaces the learned decision router with a fixed rule-based DirectReuse/VerifiedReuse/FreshReasoning policy while retaining the remaining world model, Candidate Assessment, and hard feasibility checks. This variant isolates the contribution of learned reasoning allocation from the availability of the three reasoning routes.

w/o Online Evolution. Keeps the initial world knowledge fixed throughout evaluation.

F.2 Complete ScienceWorld Ablation Results

Table 15 complements the main-paper ablations with detailed reasoning-cost statistics. Removing major knowledge or decision components often increases dependence on expensive reasoning. The results therefore connect task performance with the efficiency of experience reuse.

F.3 ALFWorld Ablation

Table 16 evaluates the same mechanism-level ablations on ALFWorld. The trends remain consistent with ScienceWorld and are generally more evident on unseen tasks, where previous experience must be adapted to new states.

Table 14: End-to-end routing strategies on ScienceWorld under the task-family-grouped protocol. Results are averaged over three seeds. The best and second-best scores are highlighted in bold and underlined, respectively.

Strategy	Score ↑	Direct (%)	Verify (%)	Fresh (%)	Time (s) ↓
Always Fresh	<u>44.69</u>	0.00	0.00	100.00	404.3
Always Direct	37.15	100.00	0.00	0.00	171.0
Rule-based D/F	42.08	42.15	0.00	57.85	328.6
Rule-based D/V/F	42.95	43.67	16.00	40.33	320.3
Learned D/F	44.37	52.28	0.00	47.72	315.7
CWME D/V/F (ours)	**45.39**	49.11	15.44	35.44	301.4

Table 15: Complete ablation results on ScienceWorld under the task-family-grouped protocol, averaged over three seeds. The best and second-best scores are highlighted in bold and underlined, respectively.

Method	Score ↑	Fresh Calls ↓	Other Calls	Tokens (K) ↓	Direct (%)	Time (s) ↓
CWME (ours)	**45.39**	16.30	7.10	56.00	49.11	301.4
w/o Situational Knowledge	42.93	23.77	6.07	77.37	50.11	380.1
w/o Action Trajectory Experience	38.58	35.00	0.00	127.75	0.00	446.5
w/o Failure-Aware Knowledge	<u>43.97</u>	15.70	8.27	55.37	52.47	305.7
w/o Candidate Assessment	43.06	15.70	8.43	55.53	55.13	309.4
w/o Decision Routing	42.95	18.64	7.28	62.37	43.67	320.3
w/o Online Evolution	41.16	19.83	7.03	66.53	38.50	350.9

F.4 Failure-Aware Behavior

Table 17 isolates the effect of Failure-Aware Knowledge. The non-progress rate is computed as the fraction of reuse attempts whose telemetry outcome is `neutral` or `stale`. The recovery-event counter records pattern-origin stages marked failed, stale, or negative in task effect that activate the recovery path; it is a diagnostic event count, not a recovery success rate. These statistics characterize how negative interaction outcomes alter future reuse behavior.

G Additional Efficiency and Sensitivity Analysis

G.1 Reasoning Cost

Table 18 reports complete reasoning-cost statistics. Repeated large-model reasoning does not consistently yield the strongest task performance. CWME instead combines reuse, Candidate Assessment, and routing to replace unnecessary fresh reasoning with cheaper alternatives when possible.

G.2 Effect of Reasoning-Model Scale

We further examine the effect of the language model used for FreshReasoning on ScienceWorld. We replace the reasoning backbone with Qwen3.5-4B, Qwen3.5-9B, and Qwen3.5-27B while keeping the remaining CWME configuration unchanged. Increasing the reasoning-model scale substantially improves performance from 4B to 9B, whereas scaling from 9B to 27B yields only a marginal additional gain at considerably higher computational cost. These results indicate that Qwen3.5-9B provides a favorable balance between task performance and reasoning efficiency, and we therefore use it as the default FreshReasoning model in the main experiments.

G.3 Sensitivity Analysis

We examine the sensitivity of CWME to three implementation parameters on the same controlled ScienceWorld subset used in the ablation experiments. We vary one parameter at a time while keeping the remaining configuration, episode schedule, and interaction budget fixed. The dagger

Table 16: Ablation results on ALFWorld, averaged over three seeds. The best and second-best results are highlighted in bold and underlined, respectively.

Method	Seen	Unseen
CWME (ours)	**60.40**	**68.17**
w/o Candidate Assessment	57.53	61.73
w/o Decision Routing	56.57	60.17
w/o Failure-Aware Knowledge	<u>58.63</u>	<u>63.97</u>
w/o Online Evolution	57.10	61.00

Table 17: Effect of Failure-Aware Knowledge on experience reuse.

Method	Score ↑	Non-progress (%) ↓	Recovery Events
w/o Failure-Aware Knowledge	43.97	7.78	15.33
CWME (ours)	45.39	4.35	34.33

denotes the value used in the formal experiments. CWME remains relatively stable around the default configuration. A moderate trajectory horizon performs best, while increasing retrieval or storage capacity yields only marginal performance gains with additional computational cost.

H Error Analysis

We categorize unsuccessful ScienceWorld episodes by the stage at which the decision process fails. Structured run logs are post-processed so that each unsuccessful episode contributes to one mutually exclusive category; the ratios in Table 21 are therefore normalized over unsuccessful episodes. The table shows that failures arise from multiple stages, including retrieval, trajectory grounding, Candidate Assessment, routing, and stale world knowledge.

Retrieval and grounding errors suggest that useful experience may either fail to enter the candidate set or fail to transfer correctly to the current state. Assessment and routing errors indicate that uncertainty in experience reuse remains another important limitation. These observations motivate more robust trajectory grounding and uncertainty handling in future work.

I Logging

For each formal run, we store structured episode- and step-level logs, including task performance, execution time, model calls, token volume, retrieved trajectories, applicability decisions, routing decisions, world-knowledge sizes, and knowledge-update operations. Model calls are recorded with the actual model identity and task role. The formal protocol snapshot additionally stores the experiment identifier and seed, benchmark split, cognition/perception model references, configuration hash, frozen-artifact hashes, schedule identity and episode count, and flags confirming that the initial artifacts are frozen while online deltas remain writable.

Table 18: Detailed reasoning-efficiency comparison on the full ScienceWorld benchmark. Results are mean ± standard deviation over three seeds. The best and second-best scores are highlighted in bold and underlined, respectively.

Method	Score ↑	Fresh Calls ↓	Other Calls	Tokens (K) ↓	Time (s) ↓
ReAct	28.49 ± 0.57	32.78 ± 1.46	0.00 ± 0.00	108.54 ± 3.01	191.8 ± 17.0
ExpeL	11.75 ± 0.33	38.06 ± 1.85	0.00 ± 0.00	132.97 ± 2.83	865.3 ± 12.5
MemP	13.63 ± 2.53	30.27 ± 1.23	0.00 ± 0.00	103.55 ± 2.78	203.1 ± 18.1
MemRL	42.63 ± 2.92	29.34 ± 1.12	0.00 ± 0.00	99.64 ± 2.87	419.1 ± 39.3
Always Fresh	$\underline{49.87 \pm 0.28}$	43.53 ± 0.83	0.00 ± 0.00	148.83 ± 3.55	335.2 ± 28.6
CWME (ours)	$\mathbf{52.02 \pm 0.90}$	26.28 ± 0.79	7.20 ± 0.22	89.95 ± 2.70	199.8 ± 13.4

Table 19: Effect of the model used for FRESHREASONING on ScienceWorld. All remaining CWME settings are kept fixed. Results are mean ± standard deviation over three seeds. The best and second-best scores are highlighted in bold and underlined, respectively.

Model	Score ↑	Fresh Calls ↓	Tokens (K) ↓	Time (s) ↓
Qwen3.5-4B	39.68 ± 1.68	61.96 ± 1.15	77.68 ± 5.30	146.7 ± 8.6
Qwen3.5-9B	$\underline{52.02 \pm 0.90}$	26.28 ± 0.79	89.95 ± 2.70	199.8 ± 13.4
Qwen3.5-27B	$\mathbf{52.97 \pm 0.75}$	30.76 ± 1.45	114.52 ± 20.50	435.9 ± 25.7

Table 20: Sensitivity analysis on the task-family-grouped ScienceWorld subset used in the ablation experiments. Only the listed parameter is varied, and † denotes the value used in the formal experiments.

Parameter	Value	Score ↑	Time (s) ↓
H_{traj}	3	<u>44.62</u>	306.1
H_{traj}	5†	**45.39**	301.4
H_{traj}	8	44.06	327.8
`MAX_QUERY`	2	43.98	297.5
`MAX_QUERY`	3†	<u>45.39</u>	301.4
`MAX_QUERY`	5	**45.62**	318.3
Writable records per source bucket	3	44.68	309.8
Writable records per source bucket	5†	<u>45.39</u>	301.4
Writable records per source bucket	8	**45.57**	311.2

Table 21: Distribution of error categories among unsuccessful ScienceWorld episodes.

Failure Type	Ratio (%)
Retrieval miss	18.00
Applicability false positive	14.00
Applicability false negative	10.00
Incorrect reasoning route	16.00
Trajectory grounding error	18.00
Stale world knowledge	9.00
Environment / action error	9.00
Other	6.00