

World-Model Evolving Agents

Leonardo Ranaldi

ILCC, School of Informatics, University of Edinburgh, United Kingdom

OMNIA Lab & University of Rome Tor Vergata, Italy

Idiap Research Institute, Switzerland

{first_name.last_name}@ed.ac.uk

Abstract

Agents learn from experience, distilling it into memories, skills, or abstractions that improve which action they propose. However, the consequence of that action is discovered only after execution, when a long-horizon error may already be irreversible. We introduce WEVA (**W**orld-model **E**volving **A**gents), a framework that makes experience predictive. WEVA learns from interaction traces what information is predictive and how actions transform the resulting state, using a *quasi-symbolic representation* over entities, preconditions, and outcomes. Specifically, before acting, WEVA predicts the states induced by candidate action sequences, discards those that violate learned preconditions, and executes the first action of the highest-scoring surviving branch. The agreement between predicted and realised states delivers a model-trust signal that regulates when lookahead is reliable and when further interaction is more informative. WEVA improves task success while executing fewer environment actions, reduces trajectories that terminate in verified constraint violations, and detects changes in environmental dynamics before they are reflected in task success. These findings position WEVA as a step towards agents capable of predicting the consequences before acting.¹

1 Introduction

A central aim for (language) agents is to become more competent as they act, and a growing line of research has been working on making that experience reusable, structuring reasoning trajectories into abstract representations (Ranaldi et al., 2025; Ranaldi and Pucci, 2025), scaling and distilling into reasoning memories (Zanzotto et al., 2025; Ouyang et al., 2026; Yu et al., 2026; Xia et al., 2026), or operating under meta-control over the reasoning cycle (Wu et al., 2025; Dupoux et al., 2026; Zhang et al., 2026c; Ranaldi and Ranaldi, 2026).

These mechanisms shape which action the agent proposes by retrieving, conditioning, or re-ranking decisions against previous experience. The consequence of a candidate action, however, is observed only after execution. Consequently, the agent learns from what has happened, leaving the stage at which a proposal is evaluated unchanged.

We argue that most current agent frameworks learn and act in a *model-free* way: accumulated experience informs which action should be proposed, but does not provide an explicit mechanism to evaluate the state that a candidate action would produce before execution. Over extended horizons, the omission becomes critical, as an error may surface only after the state in which it could have been prevented is no longer reachable. For example, Ranaldi et al. (2026b) show that an agent reaches the goal and completes the requested task, yet also modifies a non-target file, so task-specific success hides a failure to preserve the workspace.

To mitigate this issue, recent approaches (Zhu et al., 2025; Zhang et al., 2026a; Wei et al., 2026) have focused on accumulating experience to make observed failures less likely to recur. Yet, the underlying mechanism producing them is in place. This limitation extends beyond a single trajectory, proving restrictive in both single-agent and multi-agent environments; as agents increasingly interact, an unverified belief or action becomes part of the shared context upon which other agents act.

We propose WEVA (**W**orld-model **E**volving **A**gents), a framework in which the agent maintains an explicit world model and evaluates candidate actions against this world model before acting. Specifically, WEVA learns what information constitutes a predictive state, how actions transform that state, and uses the learned dynamics to anticipate consequences before committing to the environment (Figure 1). WEVA grounds these operations in quasi-symbolic representations (Ranaldi et al., 2025; Ranaldi, 2026). These representations sep-

¹Code:  [OmniAILab/world-model-evolving-agents](https://github.com/OmniAILab/world-model-evolving-agents)

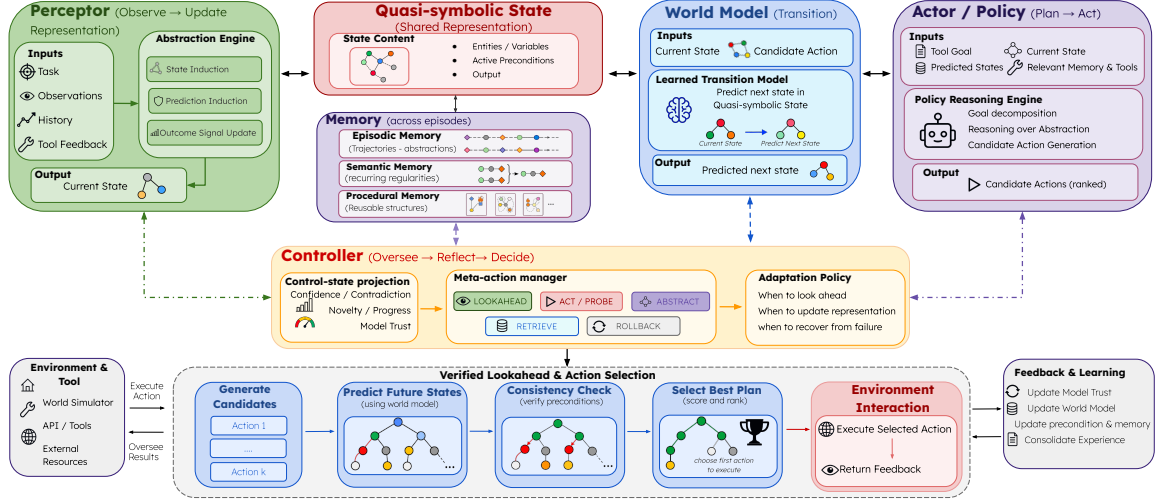


Figure 1: Overview of WEVA. The architecture orchestrates agentic reasoning, prediction, and self-evolution via four components (*Perceptor*, *World Model*, *Actor*, and *Controller*) grounded in a shared *Quasi-symbolic State*.

arate entities, preconditions, and outcomes, while allowing their contents to be induced from interaction, making predicted states directly comparable and supporting explicit consistency checks. Action selection can then be formulated as search over predicted states: discard a candidate sequence when one of its predicted transitions contradicts an active precondition before the corresponding action reaches the environment.

We evaluate WEVA across interactive planning environments and under a controlled domain shift. These settings test three aspects of the same mechanism: (i) anticipating consequences over extended horizons, (ii) verifying preconditions before commitment, (iii) revising the world model when environmental dynamics change. Together, they assess whether the agent’s internal predictions remain faithful to the environment and whether those predictions can prevent errors before execution.

Beyond task outcomes, we use *transition fidelity* as a proxy for predictive world-model quality and *detection lead* to measure how far prediction error precedes a deterioration in task success. We report the two measures separately because an agent may still complete a task even after its internal model has become stale, or fail for reasons unrelated to predictive quality. We therefore evaluate both the accuracy of the learned dynamics and their effect on downstream behaviour. WEVA improves task success across these settings, and reduces trajectories that terminate in a verified constraint violation.

Our contributions are the following:

- **Experience made predictive.** We extend

quasi-symbolic abstractions from representations of past interaction into predictive states with transitions, learned self-supervised from the agent’s own traces. The same traces determine which information the state should retain and how that information changes under an action, making predictability part of the representation-learning objective.

- **Lookahead verified before acting.** We formulate agentic decision-making as search over predicted states. Candidate action sequences are rolled forward through the learned transition operator and discarded when a predicted transition contradicts an active precondition, allowing consequences to be evaluated before they are incurred in the environment.
- **Diagnostic evidence.** We report transition fidelity as a proxy for world-model quality, separate from the external error rate that measures its effect, and show that its degradation precedes the degradation of task success under distribution shift, providing a signal that does not require the episode to fail first.

More broadly, WEVA changes how experience enters the decision process. Experience is no longer used only to improve the next action, but to construct a model of what that action may cause. This gives the agent a chance to reject a consequence before experiencing, to revise its expectations when the world behaves differently. These findings position WEVA as a step towards agents that use experience to anticipate how the world may change before acting makes that future irreversible.

2 WEVA: World-Model Evolving Agents

To predict before acting, WEVA learns what information should constitute the current state and how candidate actions transform it. The resulting predictive state provides a common interface across the framework: the Perceptor induces it from interaction, the World Model learns its dynamics, the Actor looks ahead over predicted successors, and the Controller regulates prediction and correction through model trust.

2.1 Learning a Predictive Representation

What a world model can learn is determined before learning begins by how experience is represented. Natural-language representations, including reasoning traces, distilled memories, and skill documents, leave both their vocabulary and organisation open. This flexibility makes them expressive, but provides no fixed interface for comparing a stored claim with a subsequently realised state. At the other extreme, fully symbolic representations make such comparisons explicit at the cost of defining the representation in advance. WEVA settles an intermediate point through *quasi-symbolic abstractions* (Ranaldi et al., 2025): semi-structured representations in which a state is organised around entities, preconditions, and outcomes, while the contents operating those roles are learned from interaction (§2.2). The quasi-symbolic state functions as the shared interface: the fields make transitions learnable and predictions comparable, while active preconditions make them verifiable before acting.

2.2 Predictive Quasi-Symbolic Abstractions

To predict the consequences of an action, WEVA needs a state that keeps the information governing how the interaction may change. We build this state from quasi-symbolic abstractions (Ranaldi et al., 2025), which organise interaction around variables, transitions, preconditions, and outcomes:

$$\mathcal{A} = (\mathcal{V}, \mathcal{T}, \mathcal{C}, \mathcal{E}), \quad (1)$$

where $\mathcal{V}$ denotes variables, entities, tools, or states; $\mathcal{T}$ records observed transitions; $\mathcal{C}$ captures the preconditions governing those transitions; and $\mathcal{E}$ represents outcome and progress signals.

Given an objective g , an observation o_t , and an interaction window $\tau_t^{(k)}$, the Perceptor induces $\mathcal{A}_t = \mathcal{A}_{\theta_P}(g, o_t, \tau_t^{(k)})$. From this abstraction, we define the *predictive state*

$$\sigma_t = (\mathcal{V}_t, \mathcal{C}_t, \mathcal{E}_t), \quad (2)$$

which includes the entities currently relevant to the task, the active preconditions, and the signals describing progress. The complete abstraction $\mathcal{A}_t$ is available to memory, while σ_t is the state over which the world model learns action-conditioned change.

WEVA learns a transition model $\mathcal{F}_{\theta_W}$ that maps the current state and a candidate action into the state predicted to follow:

$$\hat{\sigma}_{t+1} = \mathcal{F}_{\theta_W}(\sigma_t, a), \quad (3)$$

where $p_{\theta_W}(\cdot \mid \sigma_t, a)$ denotes the corresponding distribution over successor states. The roles of $\mathcal{V}$, $\mathcal{C}$, and $\mathcal{E}$ remain fixed across environments, while their contents are induced from interaction. This gives WEVA a common state for learning transitions, comparing predictions with realised outcomes, and checking candidate actions before execution. We define the *world model* as the combination of the learned transition model $\mathcal{F}_{\theta_W}$ and the set of active preconditions $\mathcal{C}_t$ used to verify its predictions. $\mathcal{C}_t$ contains preconditions induced by the Perceptor and preconditions promoted from procedural memory (§2.5). Consequently, updates to θ_W optimise $\mathcal{F}_{\theta_W}$, while memory-based correction revises the preconditions available in next episodes without altering θ_W .

2.3 Learning the Dynamics

Every interaction provides supervision for learning how actions change the predictive state. After executing a_t , the next observation allows the Perceptor to induce the realised successor σ_{t+1} , yielding a transition $(\sigma_t, a_t, \sigma_{t+1})$ without additional annotation. We learn $\mathcal{F}_{\theta_W}$ by minimising $\mathcal{L}_W(\theta_W)$

$$-\mathbb{E}_{(\sigma_t, a_t, \sigma_{t+1})} [\log p_{\theta_W}(\sigma_{t+1} \mid \sigma_t, a_t)]. \quad (4)$$

Transition learning also shapes what the Perceptor controls. A predictive state is useful to the extent that its contents support action-conditioned prediction; we therefore feed the likelihood assigned to the realised successor back to the Perceptor as a learning signal:

$$r_t^P = \log p_{\theta_W}(\sigma_{t+1} \mid \sigma_t, a_t), \quad (5)$$

where θ_W is frozen during the Perceptor update and θ_P during the transition update. The two components are updated alternately, allowing state induction and transition learning to improve through the same realised experience (Qiu et al., 2026).

Structural consistency and novelty, inherited from Ranaldi (2026), provide complementary signals.

We measure the resulting predictive agreement through transition fidelity (§4) and use verified error to assess how far that agreement reflects the environment.

2.4 Verified Lookahead

Predictions become useful for decision-making when they can rule out, before execution, actions inconsistent with the agent’s current preconditions. WEVA therefore performs *verified lookahead*: given the current state σ_t and horizon H , the Actor proposes B candidate actions at each predicted state, keeps the K best partial branches, and rolls the learned dynamics $\mathcal{F}_{\theta_W}$ forward (Appendix D). A branch $b = (a_1, \dots, a_h)$ of depth $1 \leq h \leq H$ produces predicted states $\hat{\sigma}_{t+1}, \dots, \hat{\sigma}_{t+h}$, with $\hat{\sigma}_t = \sigma_t$.

We check each predicted transition against the preconditions active in the source state. The transition operator keeps these preconditions unless the action changes them. We count violations as

$$\text{viol}(\sigma, a, \hat{\sigma}') = \sum_{c \in \mathcal{C}(\sigma)} \chi(c, \sigma, a, \hat{\sigma}'), \quad (6)$$

where $\chi = 1$ when taking a from σ or reaching $\hat{\sigma}'$ contradicts an applicable precondition c . The same transition operation produces the predicted successor and the violation judgement (Appendix I). This is an internal consistency check; Avg.LER measures externally verified violations after execution.

A branch is consistent only when every predicted transition satisfies the active preconditions:

$$\text{cons}(b) = \prod_{i=1}^h \mathbb{I}[\text{viol}(\hat{\sigma}_{t+i-1}, a_i, \hat{\sigma}_{t+i}) = 0]. \quad (7)$$

Among surviving branches, the Actor selects the branch with the highest predicted progress $\ell(\hat{\sigma}_{t+h})$ in $\mathcal{E}$ and executes the first action of the selected branch. A violation at a later predicted step can prune a branch even when the first transition is consistent. When no branch survives, the Actor falls back to $\pi_{\theta_A}(\cdot \mid g, o_t, \sigma_t)$ without lookahead verification. Model trust determines when verified lookahead guides action selection (§2.6).

2.5 World-Model Correction

Prediction error becomes corrective when it changes the world model used for subsequent decisions. A recurring prediction–realisation disagreement under the same action and structurally similar

conditions becomes a candidate precondition in $\mathcal{C}$. We propose the candidate from the difference between the predicted and realised successor, given the executed action and similar transitions retrieved from memory. Subsequent transitions increase or decrease its evidence, and an admitted precondition can prune future branches matching the same failure pattern.

Recurrence alone can introduce spurious preconditions. We therefore admit a candidate only when it *extends without regressing*: it must reject at least one observed failure without contradicting an earlier successful transition.

Memory supports this correction at three levels: realised transitions, recurring regularities, and preconditions used during action selection. Appendix C details their promotion and retrieval. This gives WEVA a non-gradient route to correct the precondition component of the world model through subsequent interaction.

2.6 Model Trust and Meta-Control

Lookahead is useful only as long as the learned dynamics remain predictive under the current environmental regime. WEVA monitors this reliability through a model-trust signal computed over a window W of executed steps,

$$\mu_t = \frac{1}{|W|} \sum_{i \in W} \text{sim}_{\psi}(\hat{\sigma}_i, \sigma_i), \quad (8)$$

which compares the predicted successor of each committed action with the subsequently realised state. Predictions from unexecuted branches have no realised counterpart and do not contribute. We add μ_t to the Controller state, allowing meta-control to depend on the reliability of the current world model (Appendix B).

The Controller uses μ_t to decide when to predict consequences internally and when further interaction is more informative. High μ_t supports LOOKAHEAD. As agreement declines, direct interaction becomes preferable because the current world model no longer reliably predicts realised states. Under a sustained decline, the Controller often selects ABSTRACT (Appendix E), re-inducing the state over a broader trajectory to support world-model correction.

Field-level components of sim_{ψ} identify whether prediction–realisation agreement is lowest for $\mathcal{V}$, $\mathcal{C}$, or $\mathcal{E}$. The correction procedure uses the identified field to guide entity re-induction, precondition revision, or progress correction (Appendix B).

Prediction error can emerge before an entire plan fails. A decline in μ_t may therefore expose changing environmental dynamics before task success deteriorates; we call the resulting gap in interaction steps *detection lead* (Appendix F).

3 Learning to Predict and Adapt

Learning in WEVA operates at two timescales (trainable WEVA configuration). Before interaction, the Controller learns when prediction, action, and correction are useful; during interaction, the Perceptor, Actor, and World Model continue learning from realised trajectories while the calibrated Controller is fixed.

Predictability-driven learning. The Perceptor and World Model are updated alternately from the same realised transitions. With θ_W fixed, the Perceptor uses the predictability signal in Equation 5 together with structural consistency and novelty. The updated Perceptor then re-induces the source and successor states. With θ_P fixed, the World Model is trained on these states using $\mathcal{L}_W$. Alternating the updates links state induction to transition learning (Qiu et al., 2026).

We fix the number of rounds in advance. Since both states are induced by the Perceptor, transition likelihood alone does not show whether the representation faithfully captures the environment. We therefore interpret transition fidelity alongside externally verified error and task success.

Memory-based correction. Realised transitions enter *Episodic Memory*, and regularities across transitions enter *Semantic Memory*. Recurring errors concerning action applicability can provide candidate preconditions for *Procedural Memory*. Supported preconditions enter $\mathcal{C}$ and can constrain subsequent lookahead; contradictory evidence can weaken or remove them. Promotion and revision occur between episodes (Appendix C).

Phase I: prior meta-control calibration. The Controller parameters ϕ are calibrated on environments sampled from $\mathcal{D}_{\text{env}}$, disjoint from the evaluation tasks, over the extended meta-action space. During calibration, the Perceptor, World Model, and Actor compute a fixed number of inner-loop updates. Post-adaptation performance provides the Controller reward. The objective penalises LOOKAHEAD when search leaves the committed action unchanged.

Phase II: interaction-time adaptation. After calibration, ϕ^* is fixed. GRPO updates the Actor from task outcomes and the Perceptor from structural consistency, novelty, and the predictability signal in Equation 5. The World Model learns from realised transitions through $\mathcal{L}_W$. The control-state projection, structural similarity sim_ψ , and state partition are fixed; the contents of the predictive state continue to be induced from interaction.

Protocol. For each instance and adaptation round, rollouts begin from the same initial context. Executed environment actions count as interaction steps. Model trust and the Controller state change after every executed action, whereas parameter updates and precondition promotion occur between episodes, without branching or restoring the active environment. Appendix H specifies the rollout group size.

4 Experiments

We evaluate WEVA across long-horizon interaction, constraints, and non-stationarity using ALFWorld (Shridhar et al., 2021), WebShop (Yao et al., 2023a), AppWorld (Trivedi et al., 2024), ScienceWorld (Wang et al., 2022) and propose a use case on WorldBench (Ranaldi et al., 2026b). We report success for ALFWorld and ScienceWorld, WebShop score, and Task Goal Completion (TGC) and Scenario Goal Completion (SGC) for AppWorld. ScienceWorld-Shift (SW-Shift) introduces an unannounced change in task-relevant environment dynamics at a predetermined point during interaction, allowing us to examine how model trust and subsequent behaviour respond to the change.

The benchmarks differ in the separation between the interaction used to learn the transition operation and the interaction used to evaluate it (Zhang et al., 2026d). The two coincide in the static environments; in SW-Shift, model trust can detect the change within the episode in which it occurs, while the world model is revised in next episodes. The environments also differ in how many preconditions can be externally verified.

Backbones. We use Qwen3.5-9B for the main experiments and evaluate frozen configurations on Gemini-3-Flash and GPT-5.4-mini to assess robustness across backbones.

Metrics. We report two complementary measures. Avg.LER captures trajectories terminating in externally verified violations of environment rules,

Method	ALFWORLD $\uparrow$	SciWORLD $\uparrow$	SW-SHIFT $\uparrow$	WEBSHOP $\uparrow$	APPW.-TGC $\uparrow$	APPW.-SGC $\uparrow$	Avg.LER $\downarrow$
Direct	46.8 (-46.0)	64.5 (-16.8)	31.2 (-36.9)	63.1 (-27.8)	—	—	—
ReAct [†] (Yao et al., 2023b)	61.2 (-31.6)	69.4 (-11.9)	41.7 (-26.4)	73.8 (-17.1)	21.6 (-37.7)	11.9 (-31.6)	23.7%
WorldEvolver [†] (Zhang et al., 2026c)	89.0 (-3.8)	78.8 (-2.5)	62.1 (-6.0)	88.5 (-2.4)	52.9 (-6.4)	37.7 (-5.8)	6.4%
EVA-Doc (frozen)	87.7 (-5.1)	78.3 (-3.0)	57.6 (-10.5)	87.7 (-3.2)	49.9 (-9.4)	35.4 (-8.1)	7.9%
EVA	89.9 (-2.9)	79.4 (-1.9)	62.8 (-5.3)	89.2 (-1.7)	54.1 (-5.2)	39.0 (-4.5)	5.8%
EVA-M (multi-agent)	91.0 (-1.8)	79.8 (-1.5)	64.2 (-3.9)	89.8 (-1.1)	60.6 (+1.3)	45.7 (+2.2)	5.5%
WEVA-H0 (no lookahead)	89.7 (-3.1)	79.2 (-2.1)	62.4 (-5.7)	89.0 (-1.9)	53.9 (-5.4)	38.7 (-4.8)	5.9%
WEVA-Doc	90.1 (-2.7)	79.6 (-1.7)	63.6 (-4.5)	89.5 (-1.4)	54.9 (-4.4)	39.6 (-3.9)	5.2%
WEVA	92.8	81.3	68.1	90.9	59.3	43.5	3.2%

Table 1: Main results on Qwen3.5-9B. Parentheses report the difference from WEVA. [†] denotes methods reproduced under our evaluation protocol using the same backbone, task splits, and interaction budget.

task constraints, or tool preconditions:

$$\text{LER} = \frac{1}{N} \sum_{n=1}^N \mathbb{I}[v^{(n)} = 1], \quad (9)$$

where $v^{(n)}=1$ when $\tau^{(n)}$ terminates in a verified violation, macro-averaged across environments supporting external verification.

Transition fidelity (TF) measures agreement between predicted and realised states:

$$\text{TF} = \frac{1}{|\mathcal{D}|} \sum_{(\sigma_t, a_t, \sigma_{t+1}) \in \mathcal{D}} \text{sim}_{\psi}(\hat{\sigma}_{t+1}, \sigma_{t+1}). \quad (10)$$

We report TF overall and over $\mathcal{V}$, $\mathcal{C}$, and $\mathcal{E}$. Avg.LER captures the environmental effect of verification, while TF isolates predictive agreement within the learned state.

Baselines. We compare with Direct, ReAct (Yao et al., 2023b) and WorldEvolver (Zhang et al., 2026c), all reproduced under the same backbone, task, settings, and interaction budget used for WEVA. We further compare with the model-free abstraction framework on which WEVA is based (Ranaldi, 2026) and a *free-text state prediction* configuration using the same horizon, scoring rule, and interaction budget. Free-text predictions are parsed into fields only for evaluation.

WEVA Configurations. We isolate the contributions of verified lookahead, constraint verification, model trust, and world-model correction under a shared representation. Full WEVA uses $H=2$ with verification and model-trust gating. **WEVA-H0** removes lookahead while retaining transition monitoring; **WEVA-U** disables verification; and **WEVA-NT** performs lookahead without model-trust gating. We retain EVA as the model-free reference. **WEVA-Doc** uses prompted Perceptor, World

Model, Actor, and Controller components (Appendix I), and adaptation occurs through memory-based correction. **WEVA-F** fixes the world model, preconditions, and retrieval set after calibration.

Implementation. The main experiments use Qwen3.5-9B, with frozen-weight comparisons on Gemini-3-Flash and GPT-5.4-mini. All methods share the same task splits, interaction budget, and backbone within each comparison. Appendix H reports implementation details, while Appendix G provides the per-backbone results.

5 Results & Discussion

5.1 Main Results

Overall performance. Table 1 reports the principal comparison on Qwen3.5-9B. WEVA improves over EVA on ALFWORLD (89.9%→92.8%), SW-Shift (62.8%→68.1%), and AppWorld TGC (54.1→59.3). WorldEvolver provides the closest predictive comparison, since both frameworks use deployment-time experience to correct future predictions, while WEVA additionally searches over predicted states and verifies each transition against active preconditions. This yields gains of +3.8 on ALFWORLD, +2.5 on ScienceWorld, and +6.0 on SW-Shift. The multi-agent EVA-M, which uses multi-agent decomposition, leads on AppWorld TGC and SGC, but we leave this to future work.

The ablations separate prospective search from constraint verification. Moving from WEVA-H0 to WEVA-U adds lookahead without verification, improving ALFWORLD (89.7%→91.1%) and AppWorld TGC from (53.9→56.3). Adding verification gives a further +1.7 and +3.0, while Avg.LER falls from 5.9% (WEVA-H0) and 5.4% (WEVA-U) to 3.2%. Search accounts for part of the gain in task success, and checking predicted transitions against active preconditions accounts for most of the reduction in verified violations.

When the world model is fixed. WEVA-F keeps structured prediction but fixes the world model, preconditions, and retrieval set. Compared with free-text prediction, the gain is +0.9 on ALFWorld and +1.4 on AppWorld TGC. Restoring interaction-time correction produces the larger difference, with +4.6 on ALFWorld (88.2%→92.8%) and +8.1 on AppWorld TGC (51.2→59.3). SW-Shift shows the same separation: WEVA-F reaches 59.0%, WorldEvolver 62.1%, WEVA-Doc 63.6%, and the full model 68.1%. A fixed predictive state supports verification, but the larger gain emerges when subsequent interaction can correct the model used to evaluate future actions.

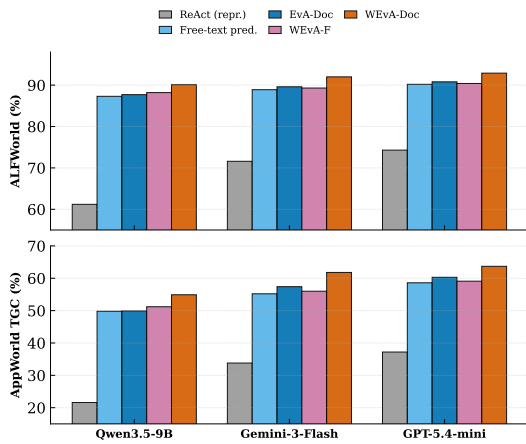


Figure 2: Performance across three backbones.

Across backbones. The same pattern holds on Gemini-3-Flash and GPT-5.4-mini (Figure 2 and Table 7). WEVA-Doc improves over EVA-Doc by +2.4 and +2.1 on ALFWorld and by +4.4 and +3.4 on AppWorld TGC, respectively. Free-text prediction is below WEVA-F on all three backbones; on Gemini-3.5 and GPT-5.4, WEVA-F is slightly below -Doc. Across all three backbones, WEVA-Doc outperforms WEVA-F, and the world-model configurations outperform ReAct. The margin contracts most on AppWorld TGC as backbone performance improves, while SW-Shift is comparatively stable (+5.3, +5.4, and +4.3 over the corresponding model-free reference). Stronger base policies leave fewer precondition errors for verification to remove, while detecting changes in environmental dynamics is useful.

Interaction cost. WEVA converges using 33% of the actions executed by ReAct and 79% of those required by WEVA-H0, while lookahead adds internal steps, not executed in the environment, equal

to 26% of the ReAct budget.

Verification and trust. Removing verification raises Avg.LER from 3.2% to 5.4%, indicating that prediction alone recovers less of the reduction in verified violations. Removing model-trust gating has limited effect on static environments but reduces SW-Shift by 4.1 (68.1%→64.0%), as stale predictions continue to guide action selection after the dynamics change. Verification therefore filters predicted branches, whilst model trust determines when those predictions remain reliable.

Structure makes prediction checkable. Free-text prediction reaches 87.3% on ALFWorld with Avg.LER at 8.3%, compared with 92.8% and 3.2% for WEVA. The key difference is that free text provides no fixed field for checking an active precondition. WEVA-F further separates structure from correction: structure alone adds +.9 on ALFWorld, interaction-time correction adds a further +4.6.

Method	TGC↑		CTS↑	
	En	Avg.Multi	En	Avg.Multi
Direct	55.4	50.3	38.6	34.6
ReAct	57.1	52.0	40.2	35.9
EVA-Doc	60.3	55.4	44.8	40.1
WEVA-Doc	62.0	58.1	51.6	48.3

Table 2: WEVA on WORLDBENCH. TGC all task-specific evaluation functions succeed, CTS requires all non-target files to remain unchanged. En averages on English; Avg. is the average over all languages.

Use case: Multilingual Grounded Workspaces.

To test WEVA in file-based workflows, we evaluate WEVA-Doc on WORLDBENCH (Ranaldi et al., 2026b). Agents act for a persona in a file-based sandbox and success requires both task completion and preservation of non-target files (Table 2). In order to make reproducible tests, we use WEVA (specifically WEVA-Doc) with Gemini-3.5 as the backbone model. WEVA-Doc reaches 48.3 CTS, +8.2 over EVA-Doc and +13.7 over Direct. The improvement comes from preservation, since predicted transitions that modify a non-target file, such as a superseded copy of the target register, are rejected before execution as reported in Figure 7. Locale conventions stored as preconditions also mitigate the gap between English and the multilingual average (3.3 against 4.7 CTS for EVA-Doc).

Variant	ALFWORLD	APPWORLD
Full WEVA	92.8	59.3
<i>World-model components:</i>		
w/o Constraint Verification	91.1 (-1.7)	56.3 (-3.0)
w/o Model Trust	91.6 (-1.2)	57.0 (-2.3)
w/o Transition Learning (<i>prompt</i>)	90.3 (-2.5)	55.1 (-4.2)
w/o Lookahead ($H=0$)	89.7 (-3.1)	53.9 (-5.4)
<i>Learning and correction:</i>		
w/o Reciprocal Perceptor	91.3 (-1.5)	56.9 (-2.4)
w/o Constraint Acquisition	91.5 (-1.3)	56.8 (-2.5)
w/o Memory Levels (<i>flat episodic</i>)	90.8 (-2.0)	55.7 (-3.6)
w/o Retrieval Policy (<i>fixed</i>)	92.0 (-0.8)	58.0 (-1.3)
w/o Field Diagnosis (<i>scalar</i>)	92.1 (-0.7)	58.2 (-1.1)
w/o Revision (<i>fixed context</i>)	88.2 (-4.6)	51.2 (-8.1)
<i>Representation:</i>		
w/o State Structure (<i>free-text</i>)	87.3 (-5.5)	49.8 (-9.5)
w/o Quasi-Symbolic Abstractions	66.8 (-26.0)	29.4 (-29.9)
<i>Inherited components:</i>		
w/o Controller	78.2 (-14.6)	40.1 (-19.2)
w/o Memory	80.3 (-12.5)	42.5 (-16.8)
w/o Online Adaptation	71.9 (-20.9)	34.2 (-25.1)

Table 3: Ablation components and representations. ALFWORLD reports success (%) and AppWorld TGC.

5.2 Ablation Study

Table 3 isolates the component contribution. Among the predictive components, removing **lookahead** produces the largest loss, followed by replacing the learned dynamics with prompted prediction. Removing **constraint verification** has a smaller effect on task success but raises LER, separating prediction from the pruning of inconsistent branches. **Model trust** matters most under change: removing the gate reduces SW-Shift by 4.1, as stale predictions continue to guide selection.

The **Perceptor**, **precondition acquisition**, and memory promotion each help keep the predictive state useful across interaction. Removing memory produces a notable than removing either learning mechanism, while disabling correction gives the largest drop in this group, with WEVA-F losing -4.6 on ALFWorld and -8.1 on AppWorld TGC. Retrieval policy and field diagnosis contribute less, suggesting that they regulate a predictive model.

Representation produces the clearest separation. Replacing the quasi-symbolic state with free-text prediction decreasing performance, while removing quasi-symbolic abstractions produces a larger loss. Free-text prediction preserves prediction but removes the structured fields used for verification; removing the abstraction removes the predictive state itself. The components inherited from EVA are important, with the largest degradation after re-

moving **Online Adaptation**, followed by the **Controller** and **Memory**.

5.3 Analysis

Lookahead and transition fidelity. Performance improves up to $H=2$ and decreases at $H=3$; since one-step transition fidelity is 0.79, accumulating prediction error across the rollout may explain this. Structured prediction improves overall fidelity, with the largest separation on $\mathcal{C}$ ($0.45 \rightarrow 0.75$), the field used for consistency checking. Appendix D reports the horizon and search budget; Appendix G reports transition fidelity by field.

Detection under shift. On SW-Shift, model trust falls below its alert level before task success degrades, and the residual concentrates on $\mathcal{C}$; Appendix F reports the detection lead.

Where verification helps. Among the static benchmarks, the gain over EVA is largest on AppWorld, where preconditions are explicit and chained, and smallest on WebShop, which exposes few verifiable preconditions (Table 1). The benefit also narrows with stronger backbones, where fewer branches remain for verification to reject.

6 Related Work

Reasoning with Abstractions. CoT remains open-vocabulary and vulnerable to bias and hallucination (Turpin et al., 2023), while neuro-symbolic approaches rely on expressed symbolic inventories (Freitas et al., 2025; Ranaldi et al., 2026a). Related work has transferred or refined reasoning across model scales through CoT supervision and self-refinement (Ranaldi and Freitas, 2024a,b), and across languages via program-aided intermediate representations (Ranaldi et al., 2024). QuaSAR (Ranaldi et al., 2025) introduced quasi-symbolic abstractions. WEVA extends this line from realised to predicted states, making the representation verifiable across interactions.

World Models in LLMs and VLMs. We use *model-free at decision time* to denote experience-driven agents without explicit predictive models. Recent work learns environment dynamics and successor states (Qiu et al., 2026; Wang et al., 2025; team et al., 2025), generally evaluating predictive accuracy. WEVA uses these predictions for prospective search and verifies each transition against explicit preconditions before acting.

System	Learns from experience	Predicts next state	Searches futures	Checks constraints	Acts in environment
EVA	✓	–	–	–	✓
SWIRL (Qiu et al., 2026)	dynamics	✓	–	–	–
Env. synthesis (Dong et al., 2026)	✓	external	–	external	✓
WorldEvolver (Zhang et al., 2026c)	context	✓	–	confidence	✓
AEL (Xu et al., 2026)	config.	–	–	–	✓
DarwinX (Zhang et al., 2026d)	config.	–	variants	–	✓
AEvo (Zhang et al., 2026a)	procedure	–	edits	external	✓
WEVA	✓	✓	✓	✓	✓

Table 4: WEVA position. *External* marks environment-supplied models; *context*, *config.*, *procedure*, and *dynamics* identify what is adapted; *Variants/edits* indicate search over agent; *confidence* indicates filtering by model certainty.

Agentic Self-Evolution. Self-evolving agents reuse experience through memories, skills, or evolving procedures to improve future action selection (Yao et al., 2023b; Xia et al., 2026; Zhang et al., 2026b). Related work adapts memory and retrieval (Xu et al., 2026), explores agent variants under an extends-without-regressing criterion (Zhang et al., 2026d), and optimises the evolution procedure itself (Zhang et al., 2026a). Zhang et al. (2026c) is the closest comparison, updating deployment-time world-model context while keeping model parameters frozen. WEVA differs by explicitly predicting successor states, verifying transitions against active preconditions, and using prediction–realisation disagreement to correct the world model.

Model-Based RL and Cognitive Architectures. Model-based RL plans over learned environment dynamics, while cognitive architectures separate observation, action, and meta-control (Dupoux et al., 2026; Ranaldi and Ranaldi, 2026). We treat these traditions eclectically: WEVA combines prospective planning with explicit meta-control, grounding both in a shared quasi-symbolic state.

Positioning. Table 4 summarises these distinctions. The central question is whether a model consults the world before acting and whether it can reject a predicted transition. SWIRL predicts dynamics without acting; WorldEvolver predicts and acts but gates predictions based on model confidence. WEVA combines prediction, prospective search, verification, and action.

7 Conclusion

We propose WEVA (**W**orld-model **E**volving **A**gents), which learns action-conditioned dynamics from experience, turning quasi-symbolic abstractions into a predictive model of interaction. WEVA

checks candidate branches against active preconditions, executes the first action of a surviving branch, and falls back to the Actor’s policy. Perception, prediction, action selection, and meta-control share a predictive state; agreement between predicted and realised states measures model trust, which regulates lookahead and correction. Across long-horizon and changing environments, WEVA improves task success, reduces violations, and detects environmental changes before task deterioration. These results position WEVA as a step towards agents that learn to anticipate the consequences of their decisions before acting.

Limitations & Future Work

Our results apply when the predictive state captures relevant consequences and explicit preconditions can check them. Lookahead adds computation, while model trust and acquired preconditions are approximate signals from interaction. Future work will calibrate trust and extend the predictive state beyond its current structure. In multi-agent settings, disagreement between reported and predicted states may help assess if information is trustworthy.

Ethics Statement

This work uses publicly available benchmarks and does not involve human subjects or private data. During the preparation of this work, we used an AI assistant to improve the paper’s readability, and we revised the content and take full responsibility.

Acknowledgements

I thank my supervisors, mentors, and collaborators across the School of Informatics, University of Edinburgh, the Huawei Edinburgh Joint Lab, the Idiap Research Institute, and the University of Rome Tor Vergata for feedback that supported this research.

References

- Guanting Dong, Junting Lu, Junjie Huang, Wanjun Zhong, Longxiang Liu, Shijue Huang, Zhenyu Li, Yang Zhao, Xiaoshuai Song, Xiaoxi Li, Jiajie Jin, Yutao Zhu, Hanbin Wang, Fangyu Lei, Qinyu Luo, Mingyang Chen, Zehui Chen, Jiazhan Feng, Ji-Rong Wen, and Zhicheng Dou. 2026. [Agent-world: Scaling real-world environment synthesis for evolving general agent intelligence](#). *Preprint*, arXiv:2604.18292.
- Emmanuel Dupoux, Yann LeCun, and Jitendra Malik. 2026. [Why ai systems don't learn and what to do about it: Lessons on autonomous learning from cognitive science](#). *Preprint*, arXiv:2603.15381.
- André Freitas, Marco Valentino, and Danilo Silva de Carvalho. 2025. [Neuro-symbolic natural language processing](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 14–15, Suzhou, China. Association for Computational Linguistics.
- Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T. Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. 2026. [Reasoningbank: Scaling agent self-evolving with reasoning memory](#). *Preprint*, arXiv:2509.25140.
- Yifu Qiu, Zheng Zhao, Waylon Li, Yftah Ziser, Anna Korhonen, Shay B. Cohen, and Edoardo M. Ponti. 2026. [Self-improving world modelling with latent actions](#). *Preprint*, arXiv:2602.06130.
- Federico Ranaldi, Leonardo Ranaldi, Fabio Massimo Zanzotto, and Shay B Cohen. 2026a. [Thinking in schemas: Robust syllogistic reasoning in LLMs](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 40891–40909, San Diego, California, United States. Association for Computational Linguistics.
- Leonardo Ranaldi. 2026. [Evolving agents](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16561–16569, San Diego, California, United States. Association for Computational Linguistics.
- Leonardo Ranaldi and Andre Freitas. 2024a. [Aligning large and small language models via chain-of-thought reasoning](#). In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1812–1827, St. Julian's, Malta. Association for Computational Linguistics.
- Leonardo Ranaldi and Andre Freitas. 2024b. [Self-refine instruction-tuning for aligning reasoning in language models](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2325–2347, Miami, Florida, USA. Association for Computational Linguistics.
- Leonardo Ranaldi and Giulia Pucci. 2025. [Multilingual reasoning via self-training](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11566–11582, Albuquerque, New Mexico. Association for Computational Linguistics.
- Leonardo Ranaldi, Giulia Pucci, Barry Haddow, and Alexandra Birch. 2024. [Empowering multi-step reasoning across languages via program-aided language models](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12171–12187, Miami, Florida, USA. Association for Computational Linguistics.
- Leonardo Ranaldi and Federico Ranaldi. 2026. [Agentic oversight via dialectic reasoning](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 24924–24944, San Diego, California, United States. Association for Computational Linguistics.
- Leonardo Ranaldi, Sherrie Shen, Jushi Kai, and Alexandra Birch. 2026b. [Worldbench: Culturally grounded benchmark for multilingual agents](#). *Preprint*, arXiv:2609.01056.
- Leonardo Ranaldi, Marco Valentino, and Andre Freitas. 2025. [Improving chain-of-thought reasoning via quasi-symbolic abstractions](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 17222–17240, Vienna, Austria. Association for Computational Linguistics.
- Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. [Alfworld: Aligning text and embodied environments for interactive learning](#). *Preprint*, arXiv:2010.03768.
- FAIR CodeGen team, Jade Copet, Quentin Carbonneaux, Gal Cohen, Jonas Gehring, Jacob Kahn, Jan-nik Kossen, Felix Kreuk, Emily McMilin, Michel Meyer, Yuxiang Wei, David Zhang, Kunhao Zheng, Jordi Armengol-Estapé, Pedram Bashiri, Maximilian Beck, Pierre Chambon, Abhishek Charnalia, Chris Cummins, and 32 others. 2025. [Cwm: An open-weights llm for research on code generation with world models](#). *Preprint*, arXiv:2510.02387.
- Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. 2024. [AppWorld: A controllable world of apps and people for benchmarking interactive coding agents](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16022–16076, Bangkok, Thailand. Association for Computational Linguistics.
- Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. 2023. [Language models](#)

- don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Preprint*, arXiv:2305.04388.
- Kangrui Wang, Pingyue Zhang, Zihan Wang, Yaning Gao, Linjie Li, Qineng Wang, Hanyang Chen, Chi Wan, Yiping Lu, Zhengyuan Yang, Lijuan Wang, Ranjay Krishna, Jiajun Wu, Li Fei-Fei, Yejin Choi, and Manling Li. 2025. [Vagen: Reinforcing world model reasoning for multi-turn vlm agents](#). *Preprint*, arXiv:2510.16907.
- Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. ScienceWorld: Is your agent smarter than a 5th grader? In *Empirical Methods in Natural Language Processing (EMNLP)*.
- Tianxin Wei, Ting-Wei Li, Zhining Liu, Xuying Ning, Ze Yang, Jiaru Zou, Zhichen Zeng, Ruizhong Qiu, Xiao Lin, Dongqi Fu, Zihao Li, Mengting Ai, Duo Zhou, Wenxuan Bao, Yunzhe Li, Gaotang Li, Cheng Qian, Yu Wang, Xiangru Tang, and 10 others. 2026. [A survey of agentic reasoning for large language models: Towards recursively self-improving and collective agents](#). *Preprint*, arXiv:2601.12538.
- Chunlong Wu, Ye Luo, Zhibo Qu, and Min Wang. 2025. [Meta-policy reflexion: Reusable reflective memory and rule admissibility for resource-efficient llm agent](#). *Preprint*, arXiv:2509.03990.
- Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. 2026. [Skillrl: Evolving agents via recursive skill-augmented reinforcement learning](#). *Preprint*, arXiv:2602.08234.
- Wujiang Xu, Jiaojiao Han, Minghao Guo, Kai Mei, Xi Zhu, Han Zhang, and Dimitris N. Metaxas. 2026. [Ael: Evolving agent harness in open-ended environments](#). *Preprint*, arXiv:2604.21725.
- Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2023a. [Webshop: Towards scalable real-world web interaction with grounded language agents](#). *Preprint*, arXiv:2207.01206.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023b. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.
- Hongzhuo Yu, Fei Zhu, Guo-Sen Xie, and Ling Shao. 2026. [Self-consolidation for self-evolving agents](#). *Preprint*, arXiv:2602.01966.
- Fabio Massimo Zanzotto, Elena Sofia Ruzzetti, Giancarlo A. Xompero, Leonardo Ranaldi, Davide Venditti, Federico Ranaldi, Cristina Giannone, Andrea Favalli, and Raniero Romagnoli. 2025. [Position paper: MeMo: Towards language models with associative memory mechanisms](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 15169–15180, Vienna, Austria. Association for Computational Linguistics.
- Jiayi Zhang, Yongfeng Gu, Jianhao Ruan, Maojia Song, Yiran Peng, Zhiguang Han, Jinyu Xiang, Zhi-tao Wang, Caiyin Yang, Yixi Ouyang, Bang Liu, Chenglin Wu, and Yuyu Luo. 2026a. [Harnessing agentic evolution](#). *Preprint*, arXiv:2605.13821.
- Wentao Zhang, Zhe Zhao, Haibin Wen, Yingcheng Wu, Ming Yin, Bo An, and Mengdi Wang. 2026b. [Autogenesis: A self-evolving agent protocol](#). *Preprint*, arXiv:2604.15034.
- Xuan Zhang, Wenxuan Zhang, See-Kiong Ng, and Yang Deng. 2026c. [Self-evolving world models for llm agent planning](#). *Preprint*, arXiv:2606.30639.
- Yifan Zhang, Yutong Dai, Juntao Tan, Luyu Yang, Rishi Mullur, Thai Hoang, Zhiyuan Hu, James Zhu, Phil Mui, Silvio Savarese, Ran Xu, and Zeyuan Chen. 2026d. [Darwinx: Evolving agent harnesses through natural selection](#). *Preprint*, arXiv:2608.07545.
- Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, Xiaoteng Ma, Xiaodong Yu, Gowtham Ramesh, Jialian Wu, Zicheng Liu, Pan Lu, James Zou, and Jiaxuan You. 2025. [Where llm agents fail and how they can learn from failures](#). *Preprint*, arXiv:2509.25370.

A Shared-State Operations

The quasi-symbolic state σ_t provides the shared interface across the four components of WEVA. The Perceptor induces the current state, the World Model predicts successor states, the Actor selects an environmental action, and the Controller selects a meta-action. Table 5 reports their inputs, outputs, and parameters. WEVA operates over a textual

Component	Param.	Reads	Writes
Perceptor	θ_P	$g, o_t, \tau_t^{(k)}$	σ_t
World Model	θ_W	σ_t, a	$\hat{\sigma}_{t+1}$
Actor	θ_A	σ_t , branch	a_t
Controller	ϕ	s_t^m	a_t^m

Table 5: WEVA components.

representation, with component-specific adapters communicating through σ_t . The free-text ablation of §5.2 measures the contribution of this shared quasi-symbolic representation.

B Control-State Components

The Controller reads

$$s_t^m = [c_t, \delta_t, \nu_t, \ell_t, \mu_t]. \quad (11)$$

The confidence proxy c_t , contradiction ratio δ_t , novelty score ν_t , and progress signal ℓ_t follow the model-free framework (Ranaldi, 2026); model trust μ_t is defined in §2.6. Structural similarity is

$$\text{sim}_\psi(\sigma, \sigma') = \frac{\sum_X w_X J(\text{plain}(X), \text{plain}(X'))}{\sum_X w_X}, \quad (12)$$

where J is Jaccard similarity, plain normalises a field, and $\mathcal{E}^\circ$ excludes terminal outcome. Retrieval ranges over $X \in \{\mathcal{V}, \mathcal{T}, \mathcal{C}, \mathcal{E}^\circ\}$, while prediction ranges over $X \in \{\mathcal{V}, \mathcal{C}, \mathcal{E}^\circ\}$. Restricting the comparison to a single field yields the field-level agreement μ_t^X used for diagnosis.

Meta-action space. The Controller selects among five meta-actions:

- LOOKAHEAD: rolls candidate branches through the transition operation and commits the first action of a consistent branch.
- ACT: proceeds with the Actor’s policy under σ_t .
- ABSTRACT: re-induces the abstraction over an extended trajectory.
- RETRIEVE: retrieves structurally compatible experience.
- ROLLBACK: restores the most recent internally consistent abstraction and reasoning context.

Field-level diagnosis. Low values of μ_t^X localise prediction error within the state: disagreement on $\mathcal{V}$ triggers entity re-induction, on $\mathcal{C}$ precondition acquisition (§2.5), and on $\mathcal{E}$ progress correction. The correction step uses this location to select the corresponding update; Appendix E relates these decisions to model trust.

What counts as the same entry. State agreement depends on plain , which normalises surface form through lowercasing, removal of determiners and punctuation, fixed entry ordering, and morphological normalisation. It uses no permitted vocabulary, canonical identifiers, or rejection rules. Equivalent facts expressed differently may count as disagreement, while lexical overlap between distinct facts may inflate agreement. Transition fidelity is therefore a noisy measure of semantic agreement.

C Memory Levels and Retrieval Policies

Levels. The *episodic* level stores realised transitions $(\sigma_t, a_t, \sigma_{t+1})$ and retrieves through the structural similarity defined in Appendix B. The *semantic* level stores regularities distilled across episodes, recording what a given action tends to change. The *procedural* level stores preconditions consulted at every step via $\mathcal{C}_t$. The three levels move from realised transitions to recurring regularities and preconditions used directly during action selection.

Promotion and demotion. A candidate precondition carries an evidence score, increasing with supporting transitions and decreasing with contradictory evidence. Candidates exceeding the threshold enter $\mathcal{C}$ only if they pass the test of §2.5; the others are discarded. Promotion and demotion occur between episodes without gradient updates, allowing the frozen configuration to improve.

Retrieval policies. The Controller selects which levels are visible, how many entries are returned, and whether retrieval returns full transitions or compressed summaries. Policies range from no retrieval, through episodic recall alone, to combined episodic and semantic retrieval. Selection conditions on s_t^m , allowing retrieval depth to respond to current model trust.

D Search Budget and Interaction Cost

At each predicted state, the Actor proposes B candidate actions and the search keeps the K best partial branches.

Lookahead horizon. Table 6 reports the effect of H : $H=2$ performs best, and the small decrease at $H=3$ is consistent with error accumulating across the rollout.

Horizon	ALFW.	APPW.	Avg.LER ↓	TF	R.CONV.
$H=0$	89.7	53.9	5.9%	0.79	$3.6\times$
$H=1$	91.4	56.8	4.4%	0.79	$4.2\times$
$H=2$	92.8	59.3	3.2%	0.79	$4.6\times$
$H=3$	92.5	58.6	3.3%	0.79	$4.4\times$

Table 6: Effect of the lookahead horizon. $H=0$ has transition monitoring without prospective search.

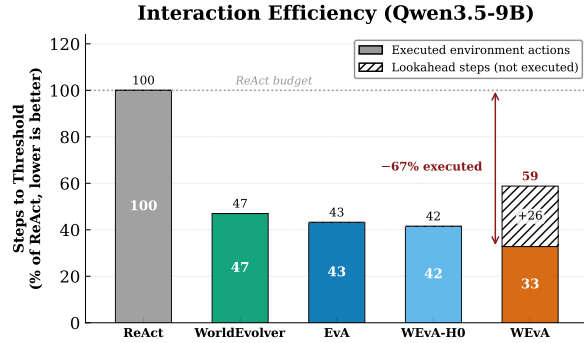


Figure 3: Executed environment actions and internal lookahead required to reach the convergence threshold, normalised to ReAct.

Interaction cost. Figure 3 separates executed environment actions from internal lookahead: lookahead exchanges environment interaction for internal computation, with model trust limiting search when the transition operation becomes unreliable.

Where lookahead adds little. The benefit is smallest when verification has little to reject: on WebShop, which has the fewest verifiable preconditions, and on short ALFWorld tasks, where lookahead preserves the Actor’s original action.

E Meta-Control against Model Trust

Figure 4 reports Controller decisions across model-trust bands. LOOKAHEAD falls from 79.6% of selections in the highest trust band to 4.1% in the lowest, while ABSTRACT increases from 0.6% to 37.2%. ACT is concentrated in the intermediate bands, where direct interaction provides evidence without requiring immediate correction, and ROLLBACK is rare. The distribution shows that the Controller uses μ_t to move from prediction towards interaction and correction as model trust decreases.

Meta-Action Selection against Model Trust (Qwen3.5-9B)

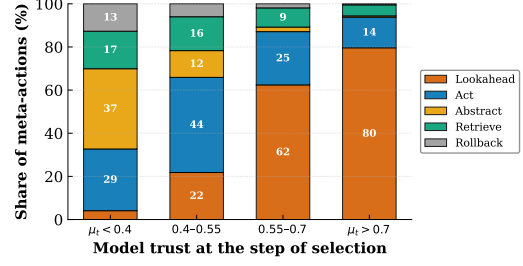


Figure 4: Share of each meta-action by model-trust band.

F Detection under Distribution Shift

Figure 5 reports model trust, field-level agreement, and task success around the SW-Shift change. Under the thresholds shown in Figure 5, the aggregate model-trust curve crosses its alert level about 70 cumulative interaction steps before task success crosses its degradation threshold. This descriptive lead depends on the chosen thresholds; agreement declines most on $\mathcal{C}$. The lower SW-Shift success of WEvA-F (59.0% vs 68.1%) is consistent with a benefit from interaction-time adaptation, without isolating its components.

Detection, Diagnosis and Recovery under Shift (SW-Shift, Qwen3.5-9B)

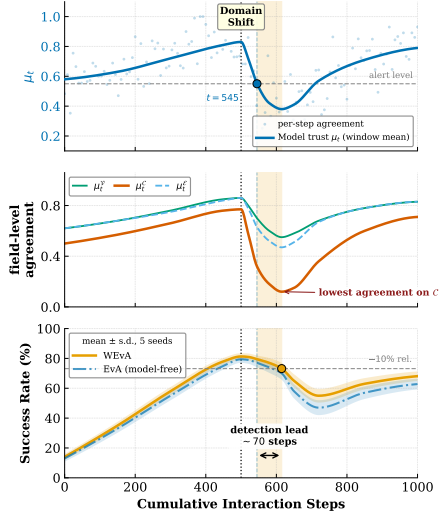


Figure 5: Detection and recovery on SW-Shift. **Top:** per-step prediction–realisation agreement and model trust μ_t ; **middle:** field-level agreement μ_t^X ; **bottom:** task success (mean $\pm$ s.d.). The shaded interval marks the detection lead.

G Backbones and Per-Backbone Fidelity

Qwen3.5-9B supports interaction-time parameter updates, while Gemini-3-Flash and GPT-5.4-mini are evaluated through frozen configurations.

Method	ALFW.	SciW.	APPW	SW-SHIFT
<i>Qwen3.5-9B (open weights)</i>				
ReAct	61.2	69.4	21.6	41.7
Free-text state pred.	87.3	77.4	49.8	57.8
EVA-Doc	87.7	78.3	49.9	57.6
WEVA-F	88.2	78.5	51.2	59.0
WEVA-Doc	90.1	79.6	54.9	63.6
WEVA (updated)	92.8	81.3	59.3	68.1
<i>Gemini-3-Flash (frozen weights)</i>				
ReAct	71.6	74.2	33.8	50.4
Free-text state pred.	88.9	79.5	55.2	62.1
EVA-Doc	89.6	80.2	57.4	63.5
WEVA-F	89.3	79.8	56.0	62.4
WEVA-Doc	92.0	81.6	61.8	68.9
<i>GPT-5.4-mini (frozen weights)</i>				
ReAct	74.3	76.1	37.2	53.8
Free-text state pred.	90.2	80.8	58.6	64.7
EVA-Doc	90.8	81.4	60.3	65.9
WEVA-F	90.4	81.0	59.1	64.9
WEVA-Doc	92.9	82.5	63.7	70.2

Table 7: Comparison across three backbones.

Across backbones, the task splits, interaction budget, $H=2$, $B=3$, $K=2$, and decoding settings is fixed. The agent and world model use the same backbone in each comparison.

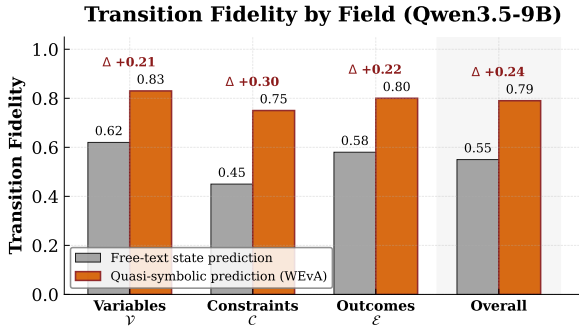


Figure 6: Transition fidelity per field.

Transition fidelity per field. Figure 6 compares structured and free-text prediction at matched scale. Overall fidelity increases with the largest separation on $\mathcal{C}$ (0.45→0.75) and the smallest on $\mathcal{V}$ (0.62→0.83). The improvement in $\mathcal{C}$ matters because the same field determines whether a predicted transition survives verification.

H Model Versions and Decoding

We use Qwen3.5-9B for the main experiments and Gemini-3-Flash and GPT-5.4-mini for frozen-weight evaluation. Table 8 reports the checkpoints.

Decoding. Decoding is kept deterministic whenever possible. The Perceptor, World Model, and

Model	Access	Checkpoint / Version
Qwen3.5-9B	open weights	Qwen/Qwen3.5-9B
Gemini-3-Flash	interface	gemini-3.5-flash
GPT-5.4-mini	interface	gpt-5.4-mini

Table 8: Models used in the experiments.

Controller use greedy decoding ($T=0$), while the Actor uses low-temperature sampling ($T=0.3$, top- $p=0.9$) to produce distinct candidates for lookahead. Maximum generation lengths are fixed per component and kept constant across methods. All baselines use the decoding configuration of their corresponding WEVA setting.

Implementation and Hyperparameters. We implement WEVA in VeRL using a shared backbone with operation-specific adapters. The Perceptor and Actor are updated with GRPO ($G=8$, learning rate 10^{-6} , KL coefficient 0.001, clipping $\epsilon=0.2$), while the World Model learns from realised transitions through successor-state likelihood. We calibrate the Controller before interaction via score-function policy gradient and keep it fixed thereafter. We use verified lookahead with $H=2$, branching factor $B=3$, and beam width $K=2$; we update memory promotion and retrieval between episodes without gradients. All experiments run on four NVIDIA H200 GPUs, with the remaining optimisation and memory settings held fixed across methods.

I Prompts

Transition operation. predicts the quasi-symbolic state following a candidate action.

Transition System Prompt

Predict the state that would follow the candidate action, keeping symbolic elements and natural language side by side.

CURRENT STATE

$\mathcal{V}$: {variables}

$\mathcal{C}$: {constraints}

$\mathcal{E}$: {outcome signals}

CANDIDATE ACTION: {action}

Return the state after the action, using the same fields and the same vocabulary. Change only what the action changes. Mark any active precondition violated by attempting the action in the current state or by reaching the predicted successor. Do not execute the action and do not describe the reasoning.

Return: $\mathcal{V}$., $\mathcal{C}$., $\mathcal{E}$., VIOLATED:.

Controller: reads the control state and selects one meta-action.

Controller System Prompt

SYSTEM: Select one meta-action from the current control state.
CONTROL STATE: confidence $c_t=\{\text{conf}\}$; contradiction ratio $\delta_t=\{\text{delta}\}$; novelty $\nu_t=\{\text{novelty}\}$; progress $\ell_t=\{\text{progress}\}$; model trust $\mu_t=\{\text{trust}\}$. All values lie in $[0, 1]$.
LOOKAHEAD: roll candidate actions forward over the transition model and commit along a consistent branch.
ACT: proceed under the current abstraction.
ABSTRACT: re-induce the abstraction over an extended trajectory window.
RETRIEVE: query memory for structurally compatible experience.
ROLLBACK: restore the last consistent internal abstraction and reasoning context.
Answer with one token: LOOKAHEAD, ACT, ABSTRACT, RETRIEVE, or ROLLBACK.

Precondition Proposal Prompt

A predicted state disagreed with the realised state.
ACTION: {action}
PREDICTED STATE: {predicted V, C, E}
REALISED STATE: {realised V, C, E}
SIMILAR TRANSITIONS: {retrieved transitions}
State one precondition that, if active, would have predicted the realised state. Use the vocabulary of the state fields.
Return: PRECONDITION:.

Perceptor: induces the predictive state from the goal, the current observation, and the recent interaction.

Perceptor System Prompt

Induce the current state, keeping symbolic elements and natural language side by side.
GOAL: {goal}
OBSERVATION: {observation}
RECENT INTERACTION: {window}
ACTIVE PRECONDITIONS: {preconditions from memory}
Keep only what governs how the interaction may change.
 $\mathcal{V}$: entities, tools, and their current states.
 $\mathcal{C}$: preconditions that hold and constrain the next actions.
 $\mathcal{E}$: progress towards the goal and observed outcomes.
Return: $\mathcal{V}$; $\mathcal{C}$; $\mathcal{E}$..

Actor: proposes candidate actions from the current state.

Actor System Prompt

Propose the next action towards the goal.
GOAL: {goal}
CURRENT STATE: $\mathcal{V}$: {variables}; $\mathcal{C}$: {constraints}; $\mathcal{E}$: {outcome signals}
RETRIEVED EXPERIENCE: {memory}
Propose {B} distinct actions that are admissible in the environment, ranked by expected progress. Do not execute them.
Return one action per line.

Precondition proposal. A recurring prediction–realisation disagreement is turned into a candidate precondition (§2.5).

J Example of a WORLDBENCH Task Execution

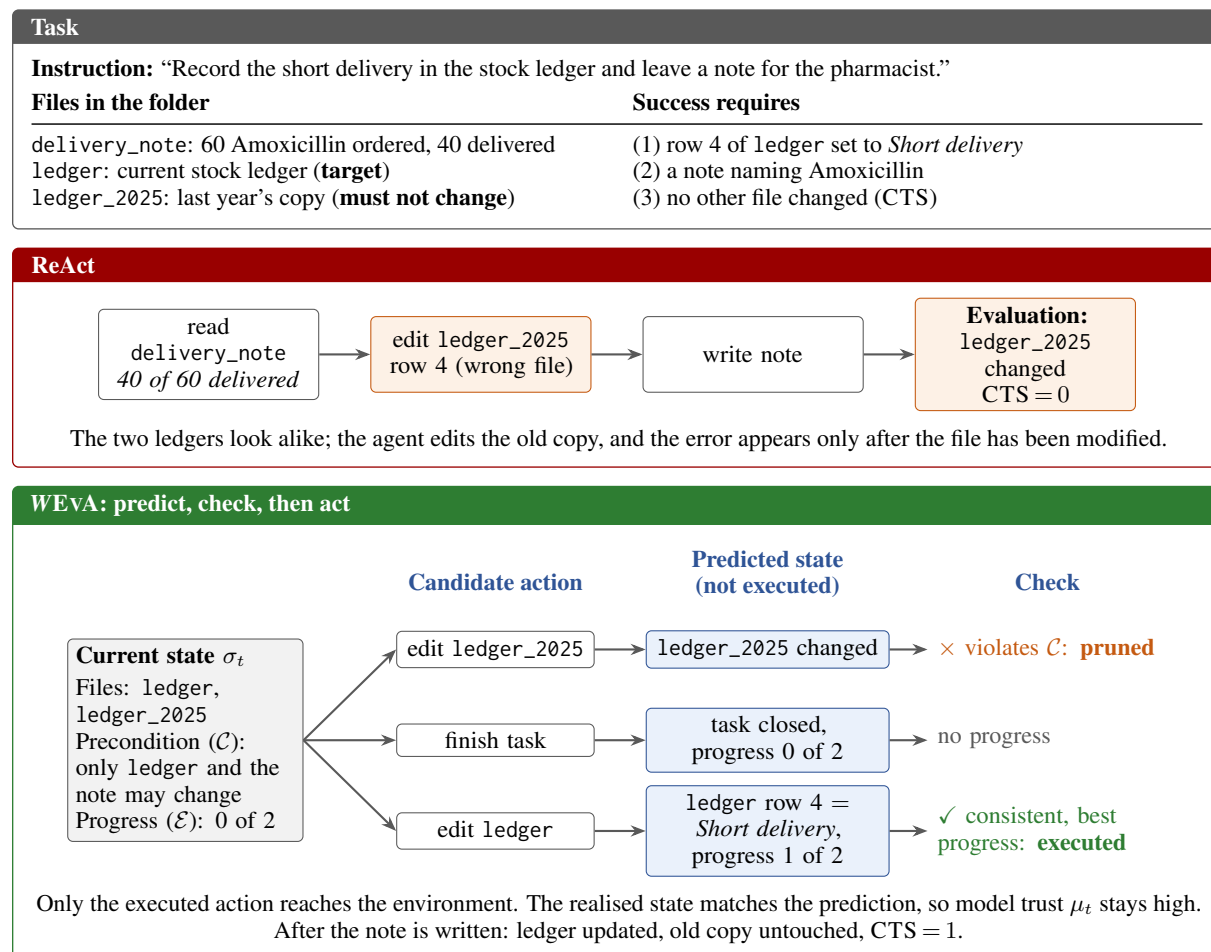


Figure 7: WEVA on a WORLDBENCH task. In the ReAct-style execution (red), the edit to the old copy of the ledger is detected only after execution. WEVA (green) predicts the outcome of each candidate, prunes the edit that violates $\mathcal{C}$, and executes only the consistent action.