# From Hesitation to Intuition: Internalizing Test-Time Adaptation into LLM Agents via On-Policy Self-Distillation

**Anonymous authors**
Paper under double-blind review

## Abstract

Memory-augmented large language model (LLM) agents adapt at test time by retrieving past experience into their context, thereby avoiding the prohibitive cost of updating parameters online. Existing approaches, however, often suffer from two fundamental limitations, *i.e.*, the decision of when to refresh the strategy relies on the agent's own self-evaluation, which is vulnerable to cognitive hallucinations, and the acquired experience is kept entirely outside the model parameters, leading to repeated memory retrieval and strategy reconstruction. To address these issues, in this paper, we propose TMA, a **T**hinking **M**ode-**A**ware agentic memory framework that consolidates test-time adaptation into the policy parameters. Specifically, we learn a linear discriminant probe on the hidden states of confident and hesitant chain-of-thought (CoT) segments, where the hesitant ones exhibit typical failure patterns, and project the agent's hidden state onto the learned direction as a hesitation score. Reflection is triggered only at hesitant states, adjusting the global strategy with retrieved trajectory memory. We further develop an on-policy self-distillation scheme, in which a teacher conditioned on privileged hindsight, *i.e.*, successful trajectories and high-quality reflections, supervises the student on its own rollouts to jointly optimize task execution and reflection generation. In this way, slow reflection is progressively absorbed into fast single-pass decisions, reducing subsequent reflection triggering. Empirical evaluations on ALFWorld, WebShop and Jericho demonstrate that TMA outperforms the state-of-the-art memory baselines with a relative success-rate gain of up to 8.7%. After five rounds of self-distillation on ALFWorld, TMA further improves the success rate by 6.4%, while reducing the reflections and token consumption per task by 92% and 28%, respectively.

## 1 Introduction

LLM agents are increasingly deployed on long-horizon tasks, such as web navigation (Qi et al., 2025; Zhou et al., 2024), automated software engineering (Jimenez et al., 2024; Zou et al., 2025), autonomous scientific discovery (Chen et al., 2025; Jansen et al., 2024; Zhang et al., 2026b;c), and embodied navigation (Shridhar et al., 2020). These agents exhibit remarkable proficiency in handling such complex tasks through targeted training, which introduces prohibitive costs of updating parameters online. Therefore, recent studies instead explore *test-time adaptation* via retrieving past experiences, adjusting strategies in dynamic environments during the LLM inferences (Suzgun et al., 2026; Hu et al., 2025).

As illustrated in Fig. 1, existing test-time adaptation paradigms have progressed from static experience retrieval toward dynamic strategy refreshing. The first paradigm, *i.e.*, static strategy retrieval, pre-constructs high-level strategies offline and retrieves them at task initiation (Ouyang et al., 2026), which are, however, often not specific to the current task and thus introduce redundant noise. The strategy synthesis on the fly synthesizes tailored strategies from the retrieved raw trajectories on the fly (Zheng et al., 2024; Wang et al., 2026), but confines retrieval and synthesis to the beginning of a task, so that the strategy becomes increasingly outdated as the task progresses, *i.e.*, experience drift under partial observability (Kaelbling et al., 1998). To mitigate such drift, the LLM-based self-evaluation paradigm dynamically refreshes the strategy during task execution (Zhang et al.,



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

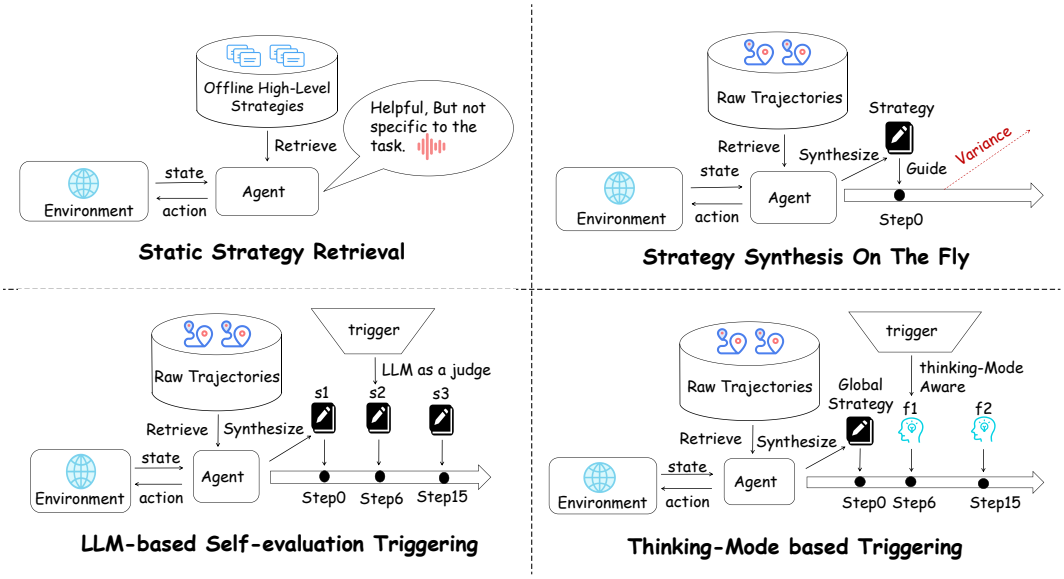



Figure 1: Comparison of existing paradigms and the proposed method TMA.

2026c), yet unfortunately relies on the agent's own self-evaluation to decide when to refresh, which is vulnerable to cognitive hallucinations (Ji et al., 2023). For instance, the agent may firmly attribute its failure to insufficient exploration while remaining unaware that its strategy has become stale, thereby skipping a necessary refresh (see Appendix E for a case study). Overall, these approaches share a common issue: when encountering analogous tasks, *agents still need to repeatedly retrieve memories and reconstruct strategies from scratch, rather than internalizing prior experience for more efficient adaptation.* In contrast, humans gradually turn repeated deliberation into intuition, and thus respond to familiar situations with less and less effort. An adaptive agent should likewise internalize its accumulated experience into fast, intuitive decisions, which motivate two questions:

*Q1: How to decide when to refresh without being misled by cognitive hallucinations? Q2: How to internalize test-time adaptation into reusable capabilities?*

To address these two questions, we propose TMA, a **T**hinking **M**ode-**A**ware agentic memory framework that triggers reflection from the agent's internal reasoning states and consolidates the resulting experience into the policy parameters. For Q1, instead of relying on the agent's self-evaluation, we fit a linear discriminant probe on the hidden states of confident and hesitant chain-of-thought segments, and project the agent's hidden state at each step onto the learned direction as a hesitation score. Only when hesitation is detected does the agent retrieve relevant trajectories from memory to synthesize a targeted reflection and refresh its global strategy. For Q2, we develop an on-policy self-distillation scheme, where a teacher conditioned on privileged hindsight, *i.e.*, successful trajectories and high-quality reflections collected in past episodes, provides dense token-level supervision on the student's own rollouts, so as to jointly optimize task execution and reflection generation. Empirically, on test-time adaptation across ALFWorld, WebShop, and Jericho, TMA outperforms representative memory baselines with a relative success-rate gain of +8.7% while achieving superior generalization on unseen tasks. Under the five-round continual learning protocol, TMA further improves the success rate on ALFWorld by 6.4%, while reducing the reflections and token consumption per task by 92% and 28%, respectively. Our contributions can be summarized as follows:

- We propose a thinking mode-aware reflection trigger that detects hesitant reasoning patterns from the agent's hidden states using a lightweight linear probe. By reusing hidden states from the agent's normal forward pass, the probe requires only a single dot product per step without additional LLM inference, enabling efficient reflection triggering while mitigating the cognitive hallucinations of LLM-based self-evaluation.
- We develop an on-policy self-distillation scheme that internalizes test-time adaptation into the policy parameters, where a hindsight-conditioned teacher jointly distills task execution

and reflection generation into the student. During the test-time adaptation, slow reflection is progressively absorbed into fast single-pass decisions, which in turn reduces reflection triggering in subsequent interactions.

- Extensive experiments on three benchmarks, including a five-round continual learning protocol and comprehensive ablations, demonstrate both the effectiveness and efficiency of the proposed TMA.

## 2 Related Work

**Memory-Augmented Test-Time Adaptation.** Test-time adaptation (TTA) optimizes inference-time decision-making by retrieving past experiences during task execution (Zhao et al., 2024). Existing agent memory systems fall into two paradigms (Park et al., 2023), as shown in Fig. 1. The first pre-constructs high-level experiences for retrieval (Ouyang et al., 2026). Although fast, the retrieved experiences are tied to out-of-date states and introduce redundant noise under distribution shifts (Sumers et al., 2023). The second retrieves raw trajectories and synthesizes tailored skills (Zheng et al., 2024; Wang et al., 2026), but relies on statically synthesized skills throughout execution, which drift from evolving task requirements under POMDP constraints (Kaelbling et al., 1998). AdaMEM mitigates this with dynamic policy refreshing, yet it depends on LLM-based self-evaluation to decide when to trigger updates (Zhang et al., 2026c), rendering it prone to the cognitive hallucinations inherent in LLMs (Ji et al., 2023). We instead propose a thinking mode-aware adaptive reflection trigger, enhancing the robustness of test-time adaptation.

**On-Policy Distillation for Hindsight Learning.** Historical trajectories expose reusable strategies and root causes of failure that are essential for online decision-making (Park et al., 2023). Prior research exploits such hindsight information through relabeling (Zhang et al., 2023), return decomposition (Arjona-Medina et al., 2019), process supervision (Luo et al., 2024), verbal reflection (Zhang et al., 2024), and experience memory (Zhong et al., 2024). However, these approaches use hindsight merely as an auxiliary in-context repository, incurring high inference overhead without a closed-loop learning paradigm (Ye et al., 2026b). On-Policy Distillation (OPD) addresses the train-inference mismatch by training a student on its own visited states with dense teacher supervision (Agarwal et al., 2024), and has been applied to mathematical reasoning (Zhao et al., 2026), knowledge QA (Ye et al., 2026a), and tool-use tasks, yet typically optimizes only a single capability dimension. In contrast, our approach simultaneously internalizes experience memories and deep deliberative reasoning into model parameters, improving performance while lowering inference costs.

## 3 Method

As depicted in Fig. 2, the TMA framework consists of three stages. In the first stage, the thinking mode-aware memory construction phase, the agent constructs a memory bank from interaction trajectories and analyzes the underlying thinking modes of chain-of-thought within these trajectories to fit a linear discriminant probe. Second, the fitted probe monitors the agent's internal reasoning states and adaptively triggers memory retrieval and reflection synthesis from the memory bank when necessary. Finally, in the third stage, an on-policy self-distillation scheme closes the loop by jointly distilling successful execution trajectories and high quality reflections collected during online interactions. This closed-loop process allows the agent to continuously leverage past experiences to refine its performance, progressively transforming costly thinking into intuitive responses.

### 3.1 Stage 1: Thinking Mode-Aware Memory Construction

**Trajectory Memory Construction.** We first construct a memory bank $\mathcal{M}$. We define a trajectory as $\tau = \{(s_t, a_t, r_t)\}_{t=1}^{T}$, where $s_t$, $a_t$, and $r_t$ denote the state, action, and reward at step $t$. We preserve the complete raw interaction trajectories, both successes and failures. To accurately retrieve relevant memories based on the agent's current state, we index the memory $\mathcal{M}$ using state embedding vectors. For every step $t$, we store a key-value pair: $k_i = \phi(s_t), v_i = (s_t, a_t, \tau_{t+1:T})$, where $i$ indexes each memory entry and $\phi(\cdot)$ is a pretrained embedding model. The value $v_i$ contains the current state $s_t$, the immediate action $a_t$, and the remaining trajectory $\tau_{t+1:T}$ that leads to the final outcome. To enable the agent to recognize when memory retrieval and reflection are necessary, we further analyze its underlying CoT reasoning patterns.

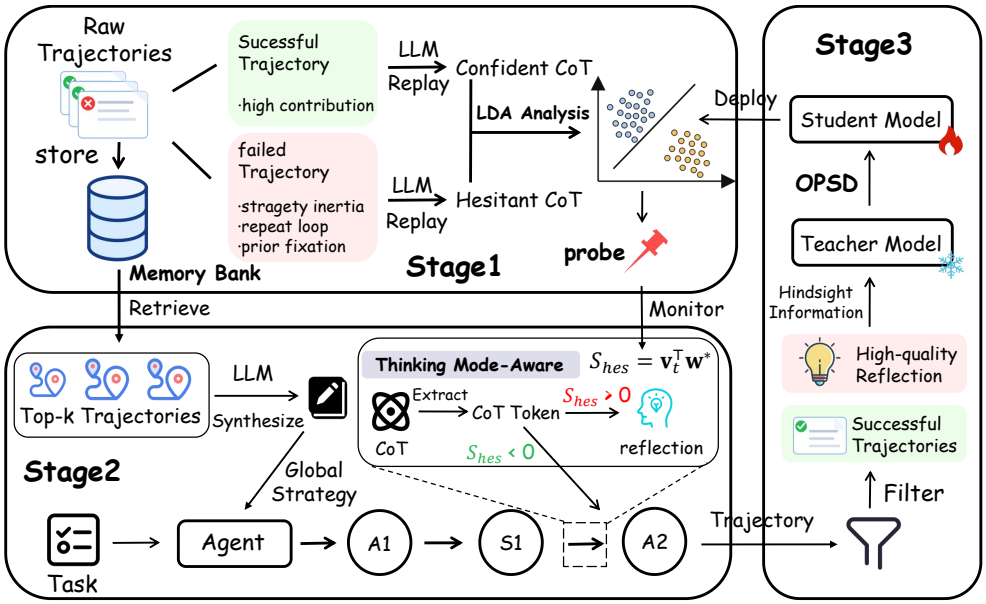



Figure 2: The main framework of the proposed method TMA.

**CoT Thinking-Mode Analysis.** We characterize each reasoning step by one of two operational thinking modes. A CoT is *confident* if it grounds the next action in newly observed evidence and advances the task. A CoT is *hesitant* if the reasoning stalls: the agent hedges (e.g., "might", "check again") or re-justifies a previously unproductive action instead of revising its plan. Hesitation is thus defined behaviorally, as stalled reasoning that typically precedes task failure, rather than as the model's subjective or token-level uncertainty. And a hesitant CoT may still produce a low-entropy action (please see Appendix B for more details). We instantiate hesitant CoTs with three canonical failure patterns: (1) *strategy inertia*, where the agent attributes stagnation to insufficient exploration while its strategy has become stale; (2) *action-repetition loops*, where the agent keeps retrying an invalid action; and (3) *commonsense-prior fixation*, where the agent searches only statistically probable locations and never explores less probable receptacles. The strong model locates the steps where these patterns first manifest and extracts the accompanying CoT rationales as hesitant samples.

We then fit a linear discriminant probe using the curated confident and hesitant CoT samples. Specifically, we use a frozen copy of the agent's backbone as the feature extractor and take the final-layer hidden state of the last CoT token as the representation $\mathbf{x} \in \mathbb{R}^d$, where $d$ is the hidden-state dimension of the corresponding backbone. This yields two sets of hidden-state representations, $C_{\text{conf}}$ and $C_{\text{hes}}$, corresponding to confident and hesitant reasoning patterns, respectively. We apply Linear Discriminant Analysis (LDA) to these representations to learn a single discriminative direction $\mathbf{w}^*$ that maximizes the separation between the two reasoning patterns relative to their within-class variance. The resulting direction serves as a lightweight probe for monitoring the agent's internal reasoning states (the detailed LDA formulation is provided in Appendix A.1).

During online interaction, we extract the final-layer hidden state $\mathbf{x}_t \in \mathbb{R}^d$ of the last CoT token directly from the agent's normal forward pass using the same backbone. We center this raw representation at the midpoint of the confident and hesitant class-conditional means, denoted as $\boldsymbol{\mu}_{\text{mid}}$, to obtain the centered hidden state $\mathbf{v}_t = \mathbf{x}_t - \boldsymbol{\mu}_{\text{mid}}$. The fitted probe projects $\mathbf{v}_t$ onto the learned direction $\mathbf{w}^*$ to obtain a scalar *Hesitation Score*, which measures how closely the current reasoning state resembles the hesitant (failure-prone) mode:

$$S_{\text{hes}}(\mathbf{v}_t) = \mathbf{v}_t^\top \mathbf{w}^*. \tag{1}$$

The probe reuses hidden states from the agent's regular forward pass and requires only a single dot product for hesitation score computation, introducing negligible additional computational overhead.

We orient the discriminant direction toward the hesitant class, such that positive scores lie on the hesitant side of the zero-centered decision boundary. This score provides an internal signal for

detecting reasoning hesitation, enabling the subsequent online adaptation phase to adaptively trigger memory retrieval and reflection when necessary.

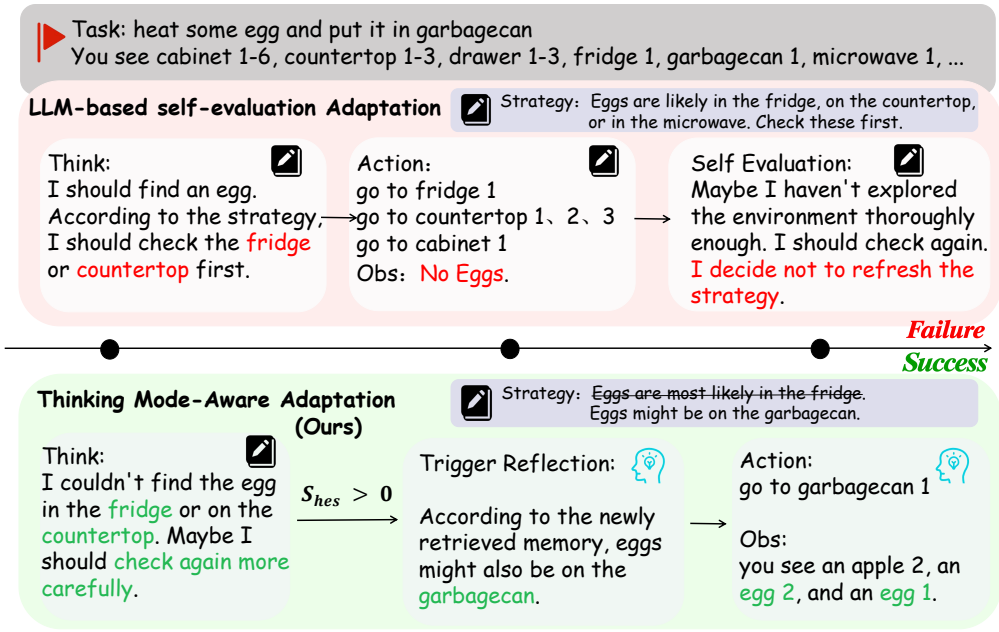



Figure 3: Comparison of LLM-based self-evaluation adaptation and thinking mode-aware adaptation.

### 3.2 Stage 2: Online Interactive Test-Time Adaptation

**Thinking Mode-Aware Strategy Adaptation.** At task initialization, the agent retrieves the top-$k$ relevant trajectories from $\mathcal{M}$ based on the initial state $s_0$ and synthesizes a global strategy $z_0$, which guides overall task execution by providing high-level plans, general principles, and critical caveats.

During execution, the agent adaptively generates a localized reflection $f_t$ when the probe detects a hesitant reasoning state. Synthesized from relevant historical trajectories retrieved from $\mathcal{M}$ and the current task context, this reflection focuses on immediate challenges and provides targeted guidance for resolving them. It is further used to revise the global strategy, allowing the agent to adapt its planning to evolving task states and mitigate experience drift. Specifically, at each step $t$, after the agent completes its CoT reasoning, the fitted probe extracts the final-layer hidden state $\mathbf{v}_t$ of the last CoT token and projects it onto the learned discriminant direction $\mathbf{w}^*$ to compute the Hesitation Score $S_{\text{hes}}^{(t)}$ via Eq. (1). Reflection is triggered when the score becomes positive, *i.e.*,

$$d_t = \mathbb{I}\left(S_{\text{hes}}^{(t)} > 0\right), \tag{2}$$

where $\mathbb{I}(\cdot)$ denotes the indicator function and $d_t \in \{0, 1\}$ determines whether reflection is needed.

When $d_t = 1$, the agent retrieves relevant trajectories from $\mathcal{M}$ and synthesizes a new reflection $f_t$ based on the task context and retrieved experiences. The new reflection replaces the previous one and is used to revise the global strategy. Otherwise, the agent retains the previous reflection $f_{t-1}$ and continues execution without additional memory retrieval. By triggering reflection only upon hesitation, TMA maintains effective adaptation while avoiding unnecessary memory retrieval.

**Comparison with LLM-Based Self-Evaluation.** Existing adaptation methods rely on the agent's self-evaluation to decide when reflection is needed, making them vulnerable to cognitive hallucinations. Since LLMs are primarily trained to predict statistically plausible text rather than verify the correctness of their own reasoning, their self-evaluation may overlook less probable alternatives. Fig. 3, For instance, when searching for an egg, an agent may repeatedly check common locations like the fridge while overlooking less likely ones, attributing its failure to insufficient exploration rather than recognizing strategy errors. In contrast, TMA detects potential failure patterns in CoT

reasoning, such as incorrectly attributing failure to insufficient exploration, triggering reflection to retrieve relevant experiences that better match its current task state and revise its strategy accordingly.

### 3.3 Stage 3: Unified On-Policy Self-Distillation

Although adaptive reflection handles changing task states, repeated deliberation across similar tasks incurs high computational costs. To internalize experiences into policy parameters, we introduce an on-policy self-distillation scheme that jointly optimizes two complementary capabilities: task execution and reflection. A teacher model initialized from the same parameters provides hindsight supervision on the student's generated prefixes. The distillation process comprises two objectives:

*i) Task Execution Distillation* aims to internalize effective task behaviors, enabling the agent to make intuitive decisions without repeated memory retrieval and deliberate reflection. Both the student and teacher receive the public input $p_1 = (x)$, where $x$ denotes the task description. The teacher additionally receives privileged hindsight $h_1 = (\tau^+, z_0, f_t)$, comprising a filtered successful trajectory $\tau^+$, its associated global strategy $z_0$, and step-level reflections $f_t$.

*ii) Reflection Distillation* aims to improve the agent's ability to generate effective reflections when hesitation is detected. Following existing work (Zhang et al., 2026c), we select high-quality reflections from successful trajectories that alter subsequent actions, denoted as $f_t^+$. Both models receive the public input $p_2 = (x, \tau_{<t})$, consisting of task description and preceding interaction history, while the teacher additionally receives the corresponding reflection as privileged hindsight $h_2 = (f_t^+)$.

We then jointly optimize these two objectives under a unified on-policy self-distillation framework. Let $k \in \{1, 2\}$ index task execution and reflection distillation, respectively, with $p_k$ and $h_k$ denoting the corresponding public input and privileged hindsight. For each objective, the student samples an output sequence $\hat{y}_k$ conditioned only on the public input $\hat{y}_k \sim p_\theta^S(\cdot \mid p_k)$, where $p_\theta^S$ denotes the student policy parameterized by trainable weights $\theta$. The teacher then leverages the corresponding privileged hindsight to provide token-level supervision on the same student-generated prefixes:

$$p_{\theta,n}^S = p_\theta^S(\cdot \mid p_k, \hat{y}_{k,<n}), \quad p_{\bar{\theta},n}^T = p_{\bar{\theta}}^T(\cdot \mid p_k, h_k, \hat{y}_{k,<n}), \tag{3}$$

where $p_{\bar{\theta}}^T$ denotes the teacher policy with stop-gradient parameters $\bar{\theta}$, and $\hat{y}_{k,<n}$ represents the student-generated prefix preceding token $n$. The unified objective jointly minimizes the expected token-level divergence over student-generated sequences:

$$\mathcal{L}_{\text{distill}}(\theta) = \sum_{k=1}^{2} \mathbb{E}_{(p_k,h_k)\sim d_\pi^k}\left[\mathbb{E}_{\hat{y}_k \sim p_\theta^S(\cdot|p_k)}\left[\frac{1}{L_k}\sum_{n=1}^{L_k}\delta_{k,n}\right]\right], \ \delta_{k,n} = \mathcal{D}_{\text{KL}}\left(\text{sg}[p_{\bar{\theta},n}^T] \,\|\, p_{\theta,n}^S\right), \tag{4}$$

where $d_\pi^k$ denotes the training data distribution for objective $k$, and $L_k$ is the length of $\hat{y}_k$. And $\delta_{k,n}$ represents the token-level distillation loss that measures the divergence between the teacher's privileged hindsight-informed distribution and the student's own distribution. The stop-gradient operator $\text{sg}[\cdot]$ ensures that gradients are not backpropagated through the teacher, stabilizing the distillation process. The updated student can be re-deployed for online interaction, where newly accumulated experiences support subsequent distillation rounds, finishing the continual adaptation.

## 4 EXPERIMENTS

### 4.1 Experimental Setup

**Agent Benchmarks.** We evaluate **TMA** on three benchmarks spanning embodied navigation, web shopping, and game exploration. **ALFWorld** (Shridhar et al., 2020) features embodied household tasks across six categories. We evaluate the agent's generalization and memory utilization capabilities using the seen and unseen splits. Performance is measured by the **Success Rate**. **WebShop** (Yao et al., 2022a) simulates an online shopping environment with 1.1 million products, where agents execute natural language instructions through web interactions. We measure performance using the dense **Task Score**. **Jericho** (Hausknecht et al., 2020) is a suite of interactive fiction games that provides a controlled testbed for evaluating continual learning under non-stationary dynamics.

**Backbone Models.** We adopt **Qwen3-4B-Instruct2507** (Yang et al., 2025a) for **ALFWorld**, and an **Qwen2.5-7B-Instruct-RL** (Yang et al., 2025b) for **WebShop**.[1] Our primary evaluation employs an on-policy setup where the same backbone constructs memory, synthesizes strategies, and generates actions. This strictly tests for self-improvement rather than distillation. **Qwen3-Embedding-4B** (Zhang et al., 2025) provides embedding for all retrieval operations.

**Baselines.** We compare **TMA** against representative memory-augmented agent frameworks: *1)* **No Memory**, the base LLM agent without memory augmentation; *2)* **Synapse** (Zheng et al., 2024), which retrieves full historical trajectories as in-context demonstrations; *3)* **ReasoningBank** (Ouyang et al., 2026), which retrieves offline synthesized high-level strategies at task initialization; *4)* **AdaMEM** (Zhang et al., 2026c), which dynamically refreshes memory-guided policies via LLM-based self-evaluation; *5)* **JitRL** (Li et al., 2026), which performs reward-driven policy adjustment through prompting without parameter updates; *6)* **EvoTest** (He et al., 2026), which evolves prompts, memory, and tool-use strategies across episodes without weight updates; *7)* **OPD-Evolver** (Zhang et al., 2026a), which combines hierarchical memory with on-policy self-distillation to internalize experience utilization into model parameters.

**Implementation Details.** We implement all agents using the **ReAct** (Yao et al., 2022b) framework. To strictly control for information access, all compared memories are constructed based on an identical pool of raw trajectories that include both successful and failed episodes. Specifically, for **ALFWorld**, when the memory bank is constructed from seen test tasks, we collect 856 raw trajectories containing the seen split. When constructed from training tasks, the memory bank comprises 1,229 raw trajectories from the training split. The former setting evaluates experience reuse and internalization on repeated tasks, examining whether the agent can progressively improve task performance on recurring tasks while reducing its reliance on costly online reflection through OPSD. The latter assesses adaptation to tasks not used for memory construction. These two settings serve complementary evaluation objectives for the memory framework. For **WebShop**, the memory repository is constructed from 4,600 raw trajectories collected from the training set. For the thinking-mode probe, we curate 493 CoT samples from raw executions, comprising 289 confident samples and 204 hesitant samples. Unless otherwise specified, all tables and figures use top-8 retrieval by default. More implementation details are provided in Appendix A.

## 4.2 Main Results

### 4.2.1 Test-Time Adaptation Performance

**Effectiveness of Thinking Mode-Aware Adaptation.** The main results are presented in Table 1, which compares different memory-augmented agent frameworks under identical experimental conditions. To isolate the contribution of the thinking mode-aware adaptation mechanism, we evaluate both TMA without the third-stage OPSD process (**TMA w/o OPSD**) and the complete TMA framework after one round of OPSD. We consider two memory construction settings: *(1)* memory banks built from seen test tasks, which evaluate each framework's ability to schedule and exploit previously stored experiences, and *(2)* memory banks built from training tasks, which assess how well the acquired memory generalizes to test tasks. Across both settings, TMA w/o OPSD consistently outperforms AdaMEM, demonstrating the effectiveness of leveraging internal reasoning states for adaptive reflection triggering rather than relying on LLM-based self-evaluation. Furthermore, the complete TMA framework achieves additional performance gains through OPSD, highlighting the complementary benefits of adaptive reflection and experience internalization.

**Retrieval Scale and Memory Composition.** We investigate the effects of retrieval scale and memory composition on test-time adaptation. We vary the retrieval top-$k$ from 1 to 16 and report the average success rate on ALFWorld. As shown in Fig. 4a, performance improves steadily with increasing k and saturates at larger values, indicating that the framework robustly benefits from richer retrieved memory contexts. We further compare success-only, failure-only, and joint retrieval using a memory bank built from training tasks. As shown in Fig. 4b, joint retrieval achieves the best performance, indicating that failed trajectories provide complementary information beyond successful experiences. In contrast, failure-only retrieval provides limited guidance when used alone. Together, these results

[1] We select the 7B model for WebShop due to the insufficient success rate of 4B base model on this task, which hinders the collection of successful trajectories required to populate the long-term memory.

Table 1: Main results on embodied navigation and web shopping benchmarks. The memory bank is constructed from seen test tasks to evaluate each framework's ability to schedule and exploit stored experiences, and from training tasks to assess the generalization of stored experiences to test tasks. The best performance in each column is highlighted in bold.

| Methods | AlfWorld | | | | | | | WebShop |
|---|---|---|---|---|---|---|---|---|
| | Pick | Look | Clean | Heat | Cool | Pick2 | Avg.SR | Score |
| *Qwen3-4B-Instruct (AlfWorld); Qwen2.5-7B-Instruct-RL (WebShop)* | | | | | | | | |
| **Memory bank constructed on seen test tasks** | | | | | | | | |
| No Memory | 68.9 | 0 | 59.3 | 18.8 | 24.0 | 20.8 | 43.6 | 70.4 |
| Synapse | 94.3 | 23.1 | 55.6 | 43.8 | 28.0 | 25.0 | 52.1 | 69.0 |
| ReasoningBank | 91.4 | 7.69 | 70.4 | 50.0 | 32.0 | 33.3 | 54.3 | 71.6 |
| JitRL | 94.3 | 15.4 | 63.0 | **75.0** | 36.0 | 33.3 | 57.9 | 73.6 |
| AdaMEM | **100** | 23.1 | 66.7 | **75.0** | 72.0 | 20.8 | 65.0 | 75.6 |
| TMA(w/o OPSD) | **100** | 23.1 | 74.1 | 62.5 | **84.0** | **41.7** | 70.7 | **78.2** |
| TMA | **100** | **30.8** | **81.5** | 68.8 | 76.0 | **41.7** | **72.1** | — |
| **Memory bank constructed on training tasks** | | | | | | | | |
| No Memory | 68.9 | 0 | **59.3** | 18.8 | 24.0 | 20.8 | 43.6 | 70.4 |
| Synapse | 85.7 | 23.1 | 48.1 | 25.0 | 24.0 | 16.7 | 42.9 | 64.1 |
| ReasoningBank | 80.0 | **30.8** | 51.9 | 43.8 | 44.0 | 29.2 | 50.7 | 68.8 |
| JitRL | 71.4 | 7.69 | 48.1 | 31.3 | 20.0 | 25.0 | 40.7 | 60.3 |
| AdaMEM | 94.3 | 23.1 | 44.4 | **56.3** | 64.0 | 29.2 | 57.1 | 72.3 |
| TMA(w/o OPSD) | 94.3 | 15.4 | 51.9 | 43.8 | 72.0 | 33.3 | 59.3 | **75.7** |
| TMA | **100** | 23.1 | 55.6 | 37.5 | **76.0** | **37.5** | **62.9** | — |

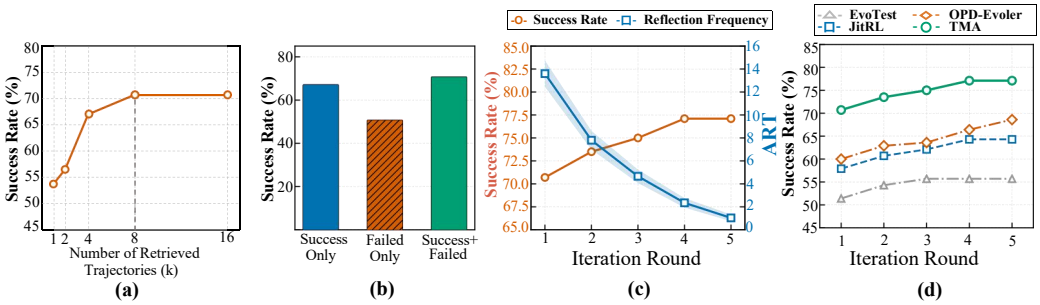



Figure 4: (a) Retrieval top-$k$ scaling analysis on ALFWorld. (b) Improvement of joint retrieval of successful and failed trajectories. (c) Success rate and average reflections per task (ART) across 5 OPSD iterative rounds on ALFWorld. (d) Learning trajectories of TMA vs. baselines over 5 rounds.

demonstrate that TMA can benefit from both successful and failed experiences in supporting effective memory-based adaptation.

### 4.2.2 Experience Internalization

**Effectiveness of the OPSD Experience Internalization.** To evaluate the effectiveness of experience internalization, we conduct five iterative rounds of the TMA closed-loop framework on ALFWorld, tracking the task success rate (SR) and average number of reflection triggers per task across successive rounds. The shaded regions around the reflection frequency curve indicate the standard error of the mean (SEM). As shown in Fig. 4c, self-distillation improves the success rate by 6.4%, while the average reflection frequency are reduced by 92%. This trend suggests that the agent progressively internalizes previously acquired deliberative experience into its policy parameters, reducing its reliance on repeated memory retrieval and costly online reflection.

**Comparison with Continual Learning Frameworks.** We compare TMA with evolutionary frameworks on Jericho and ALFWorld over five rounds to evaluate closed-loop experience internalization. Table 2 reports the Average Score (Avg) across all rounds and the Final Score (Final) after the last round, with green values indicating improvements over the initial round. Avg measures overall performance throughout continual learning, while Final reflects the performance achieved after experience internalization. As shown in Figure 4d, TMA consistently outperforms baselines on ALFWorld and improves across rounds, demonstrating effective experience internalization. Nevertheless, despite substantially reducing online reflection, TMA achieves relatively modest gains in each successive round, suggesting further improvements are needed for more efficient experience internalization.

Table 2: Closed-loop continual learning performance across five iterative rounds on Jericho games and ALFWorld.

| | Library | | Zork1 | | ALFWorld | |
|---|---|---|---|---|---|---|
| Method | Avg | Final | Avg | Final | Avg | Final |
| EvoTest | 16.2 | 18 (+3) | 36.1 | 44 (+11) | 54.6 | 55.7 (+4.3) |
| JitRL | 22.4 | 26 (+6) | 49.5 | 58 (+15) | 61.9 | 64.3 (+6.4) |
| OPD-Evolver | 30.2 | 37 (+10) | 53.0 | 75 (+35) | 64.3 | 68.6 (+8.6) |
| TMA (Ours) | **32.6** | **41** (+14) | **56.6** | **75** (+28) | **74.9** | **77.1** (+6.4) |

### 4.3 Ablation Study

To investigate the contribution of each core component, we conduct ablations on ALFWorld with the memory bank constructed from training tasks. All variants are evaluated under the same settings, reporting success rate (SR) and average inference tokens per step, as shown in Table 3.

Table 3: Ablation results on ALFWorld with the memory bank constructed from training tasks. TED and RD denote Task Execution Distillation and Reflection Distillation, respectively.

| Variant | Success Rate | Inference Tokens Per Step |
|---|---|---|
| TMA-4B | 62.9 | 3.3K |
| w/o Global Strategy $z$ | 49.3 | 4.0K |
| w/o Dynamic Reflection $f$ | 55.0 | 2.8K |
| w/ Self-Evaluation | 57.1 | 3.7K |
| w/o TED | 57.9 | 5.6K |
| w/o RD | 59.3 | 3.5K |

**Necessity of Global Strategy and Adaptive Reflection.** Removing the global strategy reduces SR from 62.9% to 49.3% and increases tokens per step from 3.3K to 4.0K. Disabling dynamic reflection lowers SR to 55.0% while reducing tokens to 2.8K. These results show their complementary roles: global strategy guides high-level planning, while reflection enables targeted correction. Removing global strategy causes a larger performance drop.

**Effectiveness of Thinking Mode-Aware Reflection Triggering.** Replacing our trigger with LLM-based self-evaluation (**w/ Self-Evaluation**) decreases the success rate from 62.9% to 57.1%, demonstrating the benefit of internal reasoning state monitoring for adaptive reflection triggering.

**Complementary Effects of Dual-Dimensional Distillation.** We ablate the two OPSD objectives by removing Task Execution Distillation (w/o TED) and Reflection Distillation (w/o RD). Without TED, SR drops from 62.9% to 57.9% and token consumption increases from 3.3K to 5.6K, indicating higher reliance on online deliberation. Removing RD further reduces SR to 59.3%. Both objectives improve performance, while TED plays a more prominent role in reducing inference overhead.

## 5 Conclusion

In this work, we presented TMA, a thinking mode-aware memory framework that internalizes test-time adaptation into LLM agents. Instead of relying on LLM-based self-evaluation, TMA applies a linear discriminant probe to the agent's hidden states to detect hesitant, failure-prone reasoning, and triggers memory retrieval and reflection. An on-policy self-distillation scheme then distills successful executions and effective reflections into the policy, gradually absorbing slow reflection into fast single-pass decisions. Experiments on ALFWorld, WebShop, and Jericho show that TMA outperforms existing memory and continual learning baselines, and that iterative self-distillation further improves success rates while substantially reducing reflections and token consumption.

### AI USE STATEMENT

In this work, we used generative AI tools to assist with language polishing, literature retrieval and discovery, and drafting selected sections of the paper. We did not use generative AI tools for research ideation or execution, generating synthetic datasets, or proving mathematical claims. Other AI-assisted activities requiring disclosure are not applicable to this work. All AI-assisted content was carefully reviewed, revised, and verified by the authors. In particular, we manually checked the relevance and accuracy of AI-assisted literature retrieval, reviewed and refined AI-generated text to ensure scientific accuracy and clarity, and independently verified the technical claims and experimental results presented in the paper. We take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the assistance of generative AI.

### ETHICS STATEMENT

This work investigates memory-augmented language model agents and their ability to adapt and reuse experiences across sequential decision-making tasks. Our experiments are conducted using publicly available benchmarks and simulated environments, without involving human subjects or collecting personally identifiable information. While the proposed framework aims to improve agent adaptability and efficiency, memory retention and autonomous decision-making may introduce potential risks related to privacy, error propagation, and unintended agent behaviors when deployed in real-world applications. We encourage appropriate safeguards, including memory management, access control, and human oversight, in practical deployments.

### REPRODUCIBILITY STATEMENT

To ensure the reproducibility of this work, detailed implementations of TMA are provided in Section 3 and the Implementation Details of Section 4. Additionally, we plan to release our source code in the future to further support reproducibility. These measures are intended to facilitate the verification and replication of our results by other researchers in the field.

## REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024.

Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. *Advances in Neural Information Processing Systems*, 32, 2019.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, et al. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*, volume 2025, pp. 96934–96990, 2025.

Matthew Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. Interactive fiction games: A colossal adventure. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7903–7910, 2020.

Yufei He, Juncheng Liu, Yue Liu, Yibo Li, Tri Cao, Zhiyuan Hu, Xinxing Xu, and Bryan Hooi. Evotest: Evolutionary test-time learning for self-improving agentic systems. In *International Conference on Learning Representations*, volume 2026, pp. 104166–104202, 2026.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, et al. Memory in the age of ai agents. *arXiv preprint arXiv:2512.13564*, 2025.

Peter Jansen, Marc-Alexandre Côté, Tushar Khot, Erin Bransom, Bhavana Dalvi Mishra, Bodhisattwa Prasad Majumder, Oyvind Tafjord, and Peter Clark. Discoveryworld: A virtual environment for developing and evaluating automated scientific discovery agents. *Advances in neural information processing systems*, 37:10088–10116, 2024.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38, 2023.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? In *International Conference on Learning Representations*, volume 2024, pp. 54107–54157, 2024.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

Yibo Li, Zijie Lin, Ailin Deng, Xuan Zhang, Yufei He, Shuo Ji, Tri Cao, and Bryan Hooi. Just-in-time reinforcement learning: Continual learning in llm agents without gradient updates. *arXiv preprint arXiv:2601.18510*, 2026.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, et al. Improve mathematical reasoning in language models by automated process supervision, 2024. *URL https://arxiv. org/abs/2406.06592*, 2, 2024.

Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836, 2018.

Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, et al. ReasoningBank: Scaling agent self-evolving with reasoning memory. In *International Conference on Learning Representations*, volume 2026, pp. 94327–94354, 2026.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, et al. WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 79791–79821, 2025.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

Theodore R Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L Griffiths. Cognitive architectures for language agents. *arXiv preprint arXiv:2309.02427*, 2023.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7080–7106, 2026.

Jingxing Wang, Chenyu Zhou, Zhihui Fu, Jun Wang, Weiwen Liu, Weinan Zhang, and Jianghao Lin. Skills on the fly: Test-time adaptive skill synthesis for llm agents. *arXiv preprint arXiv:2605.16986*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang

Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a.

Qwen: An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025b.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022b.

Qinglin Ye, Zhiyuan Gu, Jingjie Xia, Yiheng Zhang, Kaiyan Zhao, Shunchao Zheng, Yuhang Mu, Wenchao Du, and Yiming Wang. Opdsearch+: On-policy distillation with rl refinement for search-augmented reasoning, 2026a.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-policy context distillation for language models. *arXiv preprint arXiv:2602.12275*, 2026b.

Guibin Zhang, Xun Xu, Yanwei Yue, Zikun Su, Wangchunshu Zhou, Xiaobin Hu, and Shuicheng Yan. Opd-evolver: Cultivating holistic agent evolver via on-policy distillation. *arXiv preprint arXiv:2606.17628*, 2026a.

Tianjun Zhang, Fangchen Liu, Justin Wong, Pieter Abbeel, and Joseph E Gonzalez. The wisdom of hindsight makes language models better instruction followers. In *International Conference on Machine Learning*, pp. 41414–41428. PMLR, 2023.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Yunxiang Zhang, Muhammad Khalifa, Shitanshu Bhushan, Grant Murphy, Lajanugen Logeswaran, Jaekyeom Kim, Moontae Lee, Honglak Lee, and Lu Wang. MLRC-Bench: Can language agents solve machine learning research challenges? *Advances in Neural Information Processing Systems*, 38, 2026b.

Yunxiang Zhang, Yiheng Li, Ali Payani, and Lu Wang. Adamem: Test-time adaptive memory for language agents. *arXiv preprint arXiv:2606.05684*, 2026c.

Zhihan Zhang, Tao Ge, Zhenwen Liang, Wenhao Yu, Dian Yu, Mengzhao Jia, Dong Yu, and Meng Jiang. Learn beyond the answer: Training language models with reflection for mathematical reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 14720–14738, 2024.

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.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *International Conference on Learning Representations*, volume 2024, pp. 19036–19066, 2024.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 19724–19731, 2024.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, volume 2024, pp. 15585–15606, 2024.

Kaijian Zou, Aaron Xiong, Yunxiang Zhang, Frederick Zhang, Yueqi Ren, Jirong Yang, Ayoung Lee, Shitanshu Bhushan, and Lu Wang. Liveoibench: Can large language models outperform human contestants in informatics olympiads? *arXiv preprint arXiv:2510.09595*, 2025.

# Appendix

## A Implementation Details

**Backbone-Matched Probe Extraction.** For each agent backbone, we use a frozen copy of that same backbone to extract CoT hidden states and fit its corresponding LDA probe. Thus, the ALFWorld experiment use Qwen3-4B for probe feature extraction, while the WebShop experiments use Qwen2.5-7B. The 2560-dimensional formulation in Appendix A.1 describes the Qwen3-4B configuration, for other backbones, the raw feature and projection dimensions follow their respective hidden-state sizes.

### A.1 Linear Discriminant Formulation of the Thinking-Mode Probe

As described in Section 3, we fit a linear discriminant probe on confident and hesitant chain-of-thought segments. Taking Qwen3-4B as an example, we first extract a raw hidden representation $\mathbf{x} \in \mathbb{R}^{2560}$ from each CoT sample. Before fitting LDA, we apply Principal Component Analysis (PCA) to reduce the feature dimension from 2560 to 128. Specifically, PCA is fitted on the training representations to obtain the training mean $\boldsymbol{\mu}_{\text{PCA}} \in \mathbb{R}^{2560}$ and the matrix $\mathbf{P} \in \mathbb{R}^{2560\times 128}$ of the leading 128 principal directions. Each representation is transformed as

$$\widetilde{\mathbf{x}} = \mathbf{P}^{\top}(\mathbf{x} - \boldsymbol{\mu}_{\text{PCA}}) \in \mathbb{R}^{128}. \tag{5}$$

Let $C_{\text{conf}}$ and $C_{\text{hes}}$ denote the sets of PCA-reduced confident and hesitant representations, respectively. Their class-conditional means are

$$\boldsymbol{\mu}_{\text{conf}} = \frac{1}{N_{\text{conf}}} \sum_{\widetilde{\mathbf{x}} \in C_{\text{conf}}} \widetilde{\mathbf{x}}, \quad \boldsymbol{\mu}_{\text{hes}} = \frac{1}{N_{\text{hes}}} \sum_{\widetilde{\mathbf{x}} \in C_{\text{hes}}} \widetilde{\mathbf{x}}. \tag{6}$$

The within-class and between-class scatter matrices in the 128-dimensional PCA space are

$$\begin{aligned} \mathbf{S}_W = &\sum_{\widetilde{\mathbf{x}} \in C_{\text{conf}}} (\widetilde{\mathbf{x}} - \boldsymbol{\mu}_{\text{conf}})(\widetilde{\mathbf{x}} - \boldsymbol{\mu}_{\text{conf}})^{\top} \\ &+ \sum_{\widetilde{\mathbf{x}} \in C_{\text{hes}}} (\widetilde{\mathbf{x}} - \boldsymbol{\mu}_{\text{hes}})(\widetilde{\mathbf{x}} - \boldsymbol{\mu}_{\text{hes}})^{\top}, \end{aligned} \tag{7}$$

$$\mathbf{S}_B = (\boldsymbol{\mu}_{\text{hes}} - \boldsymbol{\mu}_{\text{conf}})(\boldsymbol{\mu}_{\text{hes}} - \boldsymbol{\mu}_{\text{conf}})^{\top}. \tag{8}$$

The LDA direction $\widetilde{\mathbf{w}}^* \in \mathbb{R}^{128}$ maximizes the Fisher criterion, equivalently solving the generalized eigenvalue problem:

$$J(\widetilde{\mathbf{w}}) = \frac{\widetilde{\mathbf{w}}^{\top}\mathbf{S}_B\widetilde{\mathbf{w}}}{\widetilde{\mathbf{w}}^{\top}\mathbf{S}_W\widetilde{\mathbf{w}}}, \qquad \mathbf{S}_B\widetilde{\mathbf{w}} = \lambda\mathbf{S}_W\widetilde{\mathbf{w}}. \tag{9}$$

For consistency with the main-text projection of the original hidden state, we map the resulting direction back to the raw feature space as $\mathbf{w}^* = \mathbf{P}\widetilde{\mathbf{w}}^* \in \mathbb{R}^{2560}$. Let $\boldsymbol{\mu}^{\text{raw}}_{\text{conf}}$ and $\boldsymbol{\mu}^{\text{raw}}_{\text{hes}}$ be the class means before PCA. We orient $\mathbf{w}^*$ toward the hesitant class and define the raw-space midpoint

$$\boldsymbol{\mu}_{\text{mid}} = \frac{\boldsymbol{\mu}^{\text{raw}}_{\text{conf}} + \boldsymbol{\mu}^{\text{raw}}_{\text{hes}}}{2}. \tag{10}$$

During online inference, the raw hidden state $\mathbf{x}_t$ is centered at the training-set midpoint to obtain $\mathbf{v}_t = \mathbf{x}_t - \boldsymbol{\mu}_{\text{mid}}$. The Hesitation Score is then computed by projecting the centered representation onto the learned discriminant direction:

$$S_{\text{hes}}(\mathbf{v}_t) = \mathbf{v}_t^{\top}\mathbf{w}^*. \tag{11}$$

Equivalently, the centering and projection can be combined into a single affine operation on the raw hidden state:

$$S^{(t)}_{\text{hes}} = \mathbf{x}_t^{\top}\mathbf{w}^* + b, \qquad b = -\boldsymbol{\mu}_{\text{mid}}^{\top}\mathbf{w}^*. \tag{12}$$

The bias $b$ can be precomputed after fitting the probe, allowing the Hesitation Score to be evaluated online using a single dot product and a scalar bias addition. This is equivalent to projecting the PCA-reduced representation onto $\widetilde{\mathbf{w}}^*$ after centering it at the midpoint of the reduced class means.

This is equivalent to projecting the PCA-reduced representation onto $\widetilde{\mathbf{w}}^*$ after centering it at the midpoint of the reduced class means. The centered class-conditional mean scores are

$$\begin{aligned}\mathbb{E}[S_{\text{hes}} \mid \text{hes}] &= \tfrac{1}{2}\mathbf{w}^{*\top}(\boldsymbol{\mu}^{\text{raw}}_{\text{hes}} - \boldsymbol{\mu}^{\text{raw}}_{\text{conf}}) > 0,\\ \mathbb{E}[S_{\text{hes}} \mid \text{conf}] &= -\tfrac{1}{2}\mathbf{w}^{*\top}(\boldsymbol{\mu}^{\text{raw}}_{\text{hes}} - \boldsymbol{\mu}^{\text{raw}}_{\text{conf}}) < 0.\end{aligned} \tag{13}$$

Thus, zero is the midpoint of the two projected class means. Under the standard LDA shared-covariance model with equal class priors and equal misclassification costs, this midpoint is the classification decision boundary; the zero threshold follows from centering the discriminant score rather than from the Fisher criterion alone. A higher $S_{\text{hes}}$ indicates a stronger shift toward the hesitant prototype; when the centered score exceeds zero, the system triggers the deep CoT reasoning module to retrieve relevant memory fragments and resolve the cognitive uncertainty, consistent with the trigger rule in the main text (Eq. (2)). This classification boundary is adopted as the default reflection threshold; it is not claimed to be the threshold that optimizes downstream task success or reflection cost.

### A.2 Inference Configurations

**Retrieval Configurations.** We adopt a fixed top-$k = 8$ retrieval setting throughout all experiments. Embeddings are produced by `Qwen/Qwen3-Embedding-4B`, yielding 2560-dimensional vectors that are L2-normalized before indexing. The HNSW index (Malkov & Yashunin, 2018) is built in cosine space with $M = 32$, *ef_construction* = 200, and *ef_search* = 100.

**Decoding Parameters.** We use a decoding temperature of 0.7 for all models, with a maximum generation length of 2048 tokens. When the policy backbone is `Qwen3-4B-Instruct-2507`, the maximum generation length is extended to 8192 tokens to accommodate long-CoT outputs.

### A.3 OPSD Training Configurations

On-Policy Self-Distillation (OPSD) is implemented on top of the `verl-agent` framework, fine-tuning the Qwen3-4B backbone with Fully Sharded Data Parallelism (FSDP) and LoRA (Hu et al., 2021) (rank = 32). Optimization uses AdamW with a learning rate of $1 \times 10^{-6}$. The global training batch size is set to 16, the mini-batch size is also 16, and the per-device micro-batch size is 4. The validation batch size is 32. The maximum prompt and response lengths are limited to 2048 and 512 tokens, respectively, with over-long prompts filtered out and a truncation strategy set to `error`. Training runs for at most 150 steps, with validation performed every 5 steps. We disable intermediate checkpoint saving (*save_freq* = -1). All OPSD experiments are conducted on 4 NVIDIA A100 GPUs.

**Iterative OPSD Protocol.** We use a fixed set of evaluation tasks across successive OPSD rounds to study experience reuse on repeated tasks. After each round, the agent's successful execution trajectories and corresponding reflections, collected on this same task set, provide privileged hindsight for the next round of distillation. These trajectories and reflections contain task-specific information about successful completion, which the student learns to internalize through task execution and reflection distillation. Although the task set remains unchanged, each round uses the experiences collected with the updated agent in the preceding round. To remain aligned with the evolving student, the LDA probe is also updated at each OPSD round: after the student parameters are updated, we extract CoT representations using the current backbone and refit the probe before the next round of online interaction.

To complement the prose description of the framework, we now provide the full algorithmic specification of TMA's three integrated stages. Algorithm 1 summarizes Stage 1 (Thinking Mode-Aware Memory Construction). Algorithm 2 formalizes Stage 2 (Online Interactive Test-Time Adaptation). Algorithm 3 details Stage 3 (Unified On-Policy Self-Distillation). All symbols are aligned with Section 3 and the equations therein.

**Algorithm 1: Stage 1: Thinking Mode-Aware Memory Construction**

**Input:** Raw interaction trajectory set $\mathcal{D} = \{\tau_j\}_{j=1}^{N}$; pretrained state encoder $\phi(\cdot)$; LLM backbone $f_{\text{LLM}}$; annotated CoT corpus $\{C_{\text{conf}}, C_{\text{hes}}\}$
**Output:** Factual memory $\mathcal{M}$ indexed by HNSW; cognitive orientation axis $\mathbf{w}^*$; class midpoint $\boldsymbol{\mu}_{\text{mid}}$

$\mathcal{M} \leftarrow \emptyset$
**for** each trajectory $\tau_j \in \mathcal{D}$ **do**
  **for** each step $(s_t, a_t, r_t) \in \tau_j$ **do**
    $k_i \leftarrow \phi(s_t), \quad v_i \leftarrow (s_t,\ a_t,\ \tau_{j,\,t+1:T})$
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{(k_i, v_i)\}$
  **end for**
**end for**
Index $\mathcal{M}$ by $\{k_i\}$ via HNSW
$\mathcal{X}_{\text{conf}}, \mathcal{X}_{\text{hes}} \leftarrow \emptyset$
**for** each thinking mode $c \in \{\text{conf}, \text{hes}\}$ **do**
  **for** each CoT snippet $\mathbf{c}_m \in C_c$ **do**
    $\mathbf{x}_m \leftarrow$ last-layer hidden state of $f_{\text{LLM}}(\mathbf{c}_m)$ at the final CoT token
    $\mathcal{X}_c \leftarrow \mathcal{X}_c \cup \{\mathbf{x}_m\}$
  **end for**
**end for**
$\boldsymbol{\mu}_{\text{conf}}^{\text{raw}} \leftarrow \frac{1}{N_{\text{conf}}} \sum_{\mathbf{x} \in \mathcal{X}_{\text{conf}}} \mathbf{x}$
$\boldsymbol{\mu}_{\text{hes}}^{\text{raw}} \leftarrow \frac{1}{N_{\text{hes}}} \sum_{\mathbf{x} \in \mathcal{X}_{\text{hes}}} \mathbf{x}$
$\boldsymbol{\mu}_{\text{mid}} \leftarrow \frac{1}{2}\left(\boldsymbol{\mu}_{\text{conf}}^{\text{raw}} + \boldsymbol{\mu}_{\text{hes}}^{\text{raw}}\right)$
Fit PCA on $\mathcal{X}_{\text{conf}} \cup \mathcal{X}_{\text{hes}}$ to obtain the training mean $\boldsymbol{\mu}_{\text{PCA}}$ and the projection matrix $\mathbf{P} \in \mathbb{R}^{d \times 128}$
**for** each thinking mode $c \in \{\text{conf}, \text{hes}\}$ **do**
  $\widetilde{\mathcal{X}}_c \leftarrow \{\mathbf{P}^\top(\mathbf{x} - \boldsymbol{\mu}_{\text{PCA}}) \mid \mathbf{x} \in \mathcal{X}_c\}$
**end for**
$\boldsymbol{\mu}_{\text{conf}} \leftarrow \frac{1}{N_{\text{conf}}} \sum_{\widetilde{\mathbf{x}} \in \widetilde{\mathcal{X}}_{\text{conf}}} \widetilde{\mathbf{x}}$
$\boldsymbol{\mu}_{\text{hes}} \leftarrow \frac{1}{N_{\text{hes}}} \sum_{\widetilde{\mathbf{x}} \in \widetilde{\mathcal{X}}_{\text{hes}}} \widetilde{\mathbf{x}}$
$\mathbf{S}_W \leftarrow \sum_{\widetilde{\mathbf{x}} \in \widetilde{\mathcal{X}}_{\text{conf}}} (\widetilde{\mathbf{x}} - \boldsymbol{\mu}_{\text{conf}})(\widetilde{\mathbf{x}} - \boldsymbol{\mu}_{\text{conf}})^\top + \sum_{\widetilde{\mathbf{x}} \in \widetilde{\mathcal{X}}_{\text{hes}}} (\widetilde{\mathbf{x}} - \boldsymbol{\mu}_{\text{hes}})(\widetilde{\mathbf{x}} - \boldsymbol{\mu}_{\text{hes}})^\top$
$\mathbf{S}_B \leftarrow (\boldsymbol{\mu}_{\text{hes}} - \boldsymbol{\mu}_{\text{conf}})(\boldsymbol{\mu}_{\text{hes}} - \boldsymbol{\mu}_{\text{conf}})^\top$
$\widetilde{\mathbf{w}}^* \leftarrow \arg\max_{\widetilde{\mathbf{w}} \neq \mathbf{0}} \dfrac{\widetilde{\mathbf{w}}^\top \mathbf{S}_B \widetilde{\mathbf{w}}}{\widetilde{\mathbf{w}}^\top \mathbf{S}_W \widetilde{\mathbf{w}}}$
$\mathbf{w}^* \leftarrow \mathbf{P}\widetilde{\mathbf{w}}^*$
**if** $(\boldsymbol{\mu}_{\text{hes}}^{\text{raw}} - \boldsymbol{\mu}_{\text{conf}}^{\text{raw}})^\top \mathbf{w}^* < 0$ **then**
  $\mathbf{w}^* \leftarrow -\mathbf{w}^*$
**end if**
**return** $\mathcal{M}$, $\mathbf{w}^*$, $\boldsymbol{\mu}_{\text{mid}}$

**Algorithm 2: Stage 2: Online Interactive Test-Time Adaptation**

**Input:** Factual memory $\mathcal{M}$ with HNSW index; cognitive axis $\mathbf{w}^*$; class midpoint $\boldsymbol{\mu}_{\text{mid}}$; current task $x$; initial state $s_0$; LLM backbone $f_{\text{LLM}}$
**Output:** Executed trajectory $\tau = \{(s_t, a_t)\}_{t=0}^{T_{\text{end}}-1}$
$\mathcal{T}_0 \leftarrow \text{TopK}_{\text{HNSW}}(\mathcal{M},\ \phi(s_0),\ k)$
$z_0 \leftarrow f_{\text{LLM}}\big(\Pi_{\text{global}}(x,\ s_0,\ \mathcal{T}_0)\big)$
$\mathcal{F}_{-1} \leftarrow \emptyset$
$b \leftarrow -\boldsymbol{\mu}_{\text{mid}}^\top \mathbf{w}^*$
**for** $t = 0, 1, \ldots, T_{\text{end}} - 1$ **do**
  Generate CoT reasoning for step $t$ using $f_{\text{LLM}}$ with the current task context
  $\mathbf{x}_t \leftarrow$ last-layer hidden state at the final CoT token, reused from the CoT generation forward pass
  $S_{\text{hes}}^{(t)} \leftarrow \mathbf{x}_t^\top \mathbf{w}^* + b$
  $d_t \leftarrow \mathbb{I}\Big(S_{\text{hes}}^{(t)} > 0\Big)$
  **if** $d_t = 1$ **then**
    $\mathcal{T}_t \leftarrow \text{TopK}_{\text{HNSW}}(\mathcal{M},\ \phi(s_t),\ k)$
    $f_t \leftarrow f_{\text{LLM}}\big(\Pi_{\text{slow}}(x,\ s_t,\ \mathcal{T}_t,\ \text{history}_{<t})\big)$
    $\mathcal{F}_t \leftarrow f_t$
    $z_0 \leftarrow \text{Revise}(z_0,\ f_t)$
  **else**

$\mathcal{F}_t \leftarrow \mathcal{F}_{t-1}$
**end if**
$a_t \sim f_{\text{LLM}}\big(\Pi_{\text{basic}}\big(x,\, s_t,\, z_0,\, \mathcal{F}_t,\, \text{history}_{<t}\big)\big)$
Execute $a_t$ in the environment and observe $s_{t+1}$
**end for**
**return** $\tau$

**Algorithm 3: Stage 3: Unified On-Policy Self-Distillation**

**Input:** Successful trajectory buffer $\mathcal{B}^+ = \{(\tau_j^+,\, z_0^{(j)},\, \{f_t^{(j)}\})\}$; teacher $p_{\bar{\theta}}^T$ (stop-gradient copy of student $p_\theta^S$); data distributions $d_\pi^1, d_\pi^2$; token-level KL $\mathcal{D}_{\text{tok}}$; annotated CoT corpus $\{C_{\text{conf}}, C_{\text{hes}}\}$
**Output:** Updated student parameters $\theta$; updated probe direction $\mathbf{w}^*$; updated class midpoint $\boldsymbol{\mu}_{\text{mid}}$

$\mathcal{F}^+ \leftarrow \emptyset$
**for** each step $t$ in any $\tau_j^+ \in \mathcal{B}^+$ **do**
  **if** reflection $f_t^{(j)}$ changes the agent's subsequent action **then**
    $\mathcal{F}^+ \leftarrow \mathcal{F}^+ \cup \{(x^{(j)},\, \tau_{<t}^{(j)},\, f_t^{(j)})\}$
  **end if**
**end for**
**for** $k \in \{1, 2\}$ **do**
  **if** $k = 1$ **then**
    $p_k \leftarrow x^{(j)}, \quad h_k \leftarrow (\tau_j^+,\, z_0^{(j)},\, f_t^{(j)})$
  **else if** $k = 2$ **then**
    $p_k \leftarrow (x^{(j)},\, \tau_{<t}^{(j)}), \quad h_k \leftarrow f_t^{(j)}$
  **end if**
  $\hat{y}_k \sim p_\theta^S(\cdot \mid p_k)$
  **for** $n = 1, \ldots, L_k$ **do**
    $p_{\theta,n}^S \leftarrow p_\theta^S(\cdot \mid p_k,\, \hat{y}_{k,<n})$
    $p_{\bar{\theta},n}^T \leftarrow p_{\bar{\theta}}^T(\cdot \mid p_k,\, h_k,\, \hat{y}_{k,<n})$
    $\delta_{k,n} \leftarrow \mathcal{D}_{\text{tok}}\Big(\text{sg}[p_{\bar{\theta},n}^T] \,\|\, p_{\theta,n}^S\Big)$
  **end for**
  $\mathcal{L}_k \leftarrow \frac{1}{L_k} \sum_{n=1}^{L_k} \delta_{k,n}$
  Update $\theta$ via gradient descent on $\mathcal{L}_k$ for objective $k$
**end for**
$\bar{\theta} \leftarrow \theta$ $\boldsymbol{\mu}_{\text{mid}}$ using the updated representations
**return** $\theta$

# B CoT Examples

This section presents four representative training samples drawn from the 493-sample corpus used to fit the cognitive orientation axis described in Appendix A.1. Two confident and two hesitant reasoning samples are shown to illustrate the discriminative patterns that the LDA classifier learns to separate.

## B.1 confident Thinking Samples

**Example 1: Compact confident Reasoning.**

```
  put a cool plate in cabinet
· Action open cabinet 11
· env=20 step=6

 Current Situation Analysis:

 I am now at cabinet 11, which is closed. My goal is to
 "put a cool plate in cabinet."
 I have already:
```

```
 - Gone to the diningtable and picked up plate 1.
 - Taken the plate to the fridge and cooled
   it using fridge 1.
 - Verified that the plate is now cool
   (action: "cool plate 1 with fridge 1").

 Thus, the plate is now cool, as required by the task.

 The current admissible actions include:
 - examine cabinet 11
 - examine plate 1
 - go to various cabinets, countertops,
   appliances, etc.
 - open cabinet 11

 Since the task is to "put a cool plate in cabinet,"
 the logical next step is to place the cool plate
 into one of the cabinets.
 Cabinet 11 is the one I am currently at,
 and it is closed. To place the plate inside,
 I must first open the cabinet.

 Therefore, I should open cabinet 11 to
 place the cool plate inside.

 Strategy item usage:
 - "Go to diningtable 1 to pick up the plate 1"
 → Already completed.
 - "Go to fridge 1, open it, and cool the plate 1"
 → Already completed.
 - "Go to cabinet 11 and place the cool plate 1 inside"
 → This is partially correct, but to actually place the plate,
 I must first open the cabinet.
 So, I will use the "open cabinet 11" action as a necessary step
 before placing the plate.

 Thus, the correct next action is to open cabinet 11.
```

The agent considers what it has already accomplished, including cooling the plate, and checks the available actions against the retrieved strategy. It rules out unnecessary options and confidently selects as its next action.

**Example 2: Full Confident Reasoning.**

```
put a hot mug in cabinet
· Action open cabinet 1
· env=1 step=6

Current Situation Analysis:

- The task is to "put a hot mug in cabinet."
- We have already completed the following
  steps:
  a. Moved to countertop 1
  b. Picked up mug 1
  c. Moved to microwave 1
  d. Opened microwave 1
  e. Heated mug 1 using microwave 1
  f. Moved to cabinet 1 (the target cabinet)

Now, we are at cabinet 1, which is currently closed.
The mug has been heated and is now hot.
The next logical step in the task is to place the hot mug
into a cabinet. However, we cannot place an object into
```

```
a closed cabinet without first opening it.

Admissible actions available:
- examine cabinet 1
- examine mug 1
- go to other cabinets/drawers/objects
- open cabinet 1
- go back to countertop or microwave, etc.

Relevant Strategy Items:
1. "Go to countertop 1 to pick up a mug" - Not needed.
We already have the mug and heated it.
2. "Go to microwave 1, open it, and heat the mug"
- Already completed. Mug is now hot.
3. "Go to cabinet 1, open it, and place the hot mug inside"
- This is the exact next step required to complete the task.

Therefore, we must open cabinet 1 first,
then place the hot mug inside.

Action to take: Open cabinet 1, as it is currently closed
and we need to access it to place the mug.

Note: We cannot place the mug in a closed cabinet.
The action "place mug 1 in cabinet 1" is not available.
Only "open cabinet 1" is admissible. After opening,
we can then place it.

Thus, the correct and necessary next step is to open cabinet 1.
```

The agent reviews the steps completed so far and matches each available action to the retrieved strategy. It explains why the cabinet must be opened before placing items inside and rules out other options before selecting its next action.

### B.2 Hesitant Thinking Samples

**Example 3: Hesitant – Repeated Look.**

```
put two newspaper in drawer
· Action look
· env=30 step=3

I am currently at sidetable 1, which has a desklamp 1 on it.
The task is to put two newspapers into a drawer. So far,
I have not found any newspapers in the observations.

The strategy items suggest that:
- First, perform a 'look' to detect if any newspapers
are visible in the environment, particularly on
sidetables or other surfaces.
- If newspapers are found, pick them up
and move them to a drawer.

Since I am currently at sidetable 1 and the observation says
"On the sidetable 1, you see a desklamp 1", there is
no mention of any newspapers. However, the previous "look"
actions showed "nothing" - which might mean that
newspapers are not visible in open view.

But the strategy says to perform a 'look' again
to detect visibility. Even though the previous look
showed nothing, it's possible that a more thorough
or updated look might reveal something. Since I am
now at sidetable 1, and the environment hasn't revealed
any newspapers yet, I should perform another 'look'
```

```
to ensure I don't miss any newspapers that might be
on sidetables or nearby surfaces.

I will execute a 'look' to check again for any
visible newspapers, especially on sidetables
or other surfaces, as per the strategy.

No other strategy items are relevant at this stage,
as we are still in the detection phase.

Thus, the next logical step is to perform a 'look' again
to scan the environment for newspapers.
```

The agent repeatedly expresses uncertainty using phrases such as might, possibly, and it's possible that. Despite receiving no useful information from its previous action, it repeats the same action instead of making progress.

**Example 4: Hesitant – Self-Contradiction Loop.**

```
 put a hot mug in sidetable
· Action go to countertop 1
· env=24 step=5

Current Situation Analysis:

I am currently at cabinet 1, which is open. Inside,
I see a bowl 1 and a dishsponge 1. However, I have not yet
found a mug. The task is to "put a hot mug in sidetable."

From the strategy items provided:
- First, go to a surface (e.g., countertop 1)
  where a mug is likely to be found, such as countertop 1,
  and take the mug if present.
- Then, go to the microwave 1 and heat the mug
  using the microwave.
- Finally, go to sidetable 1 and place the hot mug there.

I need to locate a mug first. The mug is not visible
in cabinet 1. The most logical place to look for a mug
would be on a countertop, as mugs are commonly found on
countertops, especially near a coffee machine
or kitchen appliances.

I have already visited countertop 1 in previous steps
and saw only a peppershaker and a spatula. I have not seen
a mug there. However, since the strategy recommends going
to a surface where a mug is likely to be found
(e.g., countertop 1),I should go there now to check
if a mug is present.

I do not see any mug in cabinet 1, and I have no information
about a mug being in the fridge, drawer, or other cabinets.
Therefore, the next logical step is to go to countertop 1 to
look for the mug.

I will not use the microwave or sidetable yet because
I have not found the mug. The task requires a hot mug,
so heating is only relevant after the mug is found.

Action to take: go to countertop 1 to look for the mug.
```

The agent acknowledges that it has already visited countertop 1 but still decides to return there. Despite repeatedly noting that it has not found the target, it fails to consider a different approach and continues repeating the same action. Together, these four examples illustrate the contrasting reasoning patterns that the LDA classifier learns to distinguish.

Table 4: Statistical metrics computed on the full-fit LDA projection of the 493-sample CoT corpus. Each metric characterizes a complementary aspect of the learned representation space: class distribution (mean projection), intra-class compactness (standard deviation), inter-class separation (Cohen's $d$), and linear discriminability (Accuracy and AUC).

| Metric | Confidence | Hesitant | Overall |
|---|---|---|---|
| **Number of samples** | 289 | 204 | 493 |
| **Mean LDA projection** | -20.527 | 21.063 | — |
| **Std. deviation** | 14.058 | 14.180 | — |
| **Cohen's $d$** | — | — | 2.95 |
| **In-sample accuracy** | — | — | 93.91% |
| **In-sample AUC** | — | — | 0.983 |

### Interpretation of LDA Metrics

To quantitatively evaluate the separability between Confident and Hesitant representations, we report several statistical metrics based on the full-fit LDA projection. Each metric characterizes a different aspect of the learned representation space, including class distribution, intra-class compactness, inter-class separation, and linear discriminability.

**Mean LDA projection.** The mean LDA projection represents the average position of samples from each category along the learned discriminative direction. Since LDA explicitly maximizes the ratio between inter-class variance and intra-class variance, the sign and magnitude of the projected mean indicate the relative location of each class in the discriminative space. In our results, confident samples have an average projection of $-20.527$, while Hesitant samples have an average projection of 21.063. The opposite signs indicate that the two behavioral states occupy different sides of the learned discriminative axis, with a center distance of 41.590, demonstrating substantial directional separation between the two categories.

**Standard deviation of LDA projection.** The standard deviation measures the intra-class dispersion after projection, reflecting how compactly samples within each category are distributed. A smaller standard deviation indicates that samples from the same category exhibit less variation along the discriminative direction. The Confident and Hesitant groups exhibit comparable standard deviations of 14.058 and 14.180, respectively, suggesting similar degrees of within-class dispersion. Relative to the distance between their projected means, these values indicate substantial separation between the two categories, although some overlap may remain in the projected representation space.

**Cohen's $d$.** Cohen's $d$ quantifies the standardized effect size between two distributions by measuring the distance between their means relative to their pooled standard deviation:

$$d = \frac{|\mu_c - \mu_h|}{\sigma_{\text{pooled}}} \tag{14}$$

where $\mu_c$ and $\mu_h$ denote the mean LDA projections of Confident and Hesitant samples, respectively, and $\sigma_{\text{pooled}}$ denotes their pooled standard deviation. Unlike raw mean differences, Cohen's $d$ accounts for the scale of within-class variation and therefore provides a normalized measurement of separability. The obtained value of 2.95 indicates a large standardized difference between the two categories, suggesting that the learned discriminative direction captures substantial differences between confident and hesitant reasoning patterns in the original 2560-dimensional representation space.

**In-sample Accuracy.** In-sample accuracy measures the percentage of correctly classified training samples using the LDA decision boundary. It evaluates whether a simple linear classifier can recover the category labels from the learned representations. The achieved accuracy of 93.91% indicates that the learned discriminative direction correctly classifies a substantial majority of the training

samples, while a non-negligible proportion remains misclassified. This result suggests that the hidden representations contain discriminative information that can be captured by a lightweight linear probe, although the two behavioral categories are not perfectly separable.

**In-sample AUC.** The Area Under the Receiver Operating Characteristic Curve (AUC) evaluates the ranking ability of the LDA classifier independently of a specific classification threshold. The obtained AUC of 0.983 indicates that the classifier generally assigns higher hesitation scores to Hesitant samples than to Confident samples. This result provides further evidence that the learned discriminative direction captures meaningful differences between the two behavioral patterns, while also indicating that the projected representations are not perfectly separable.

Overall, these metrics provide complementary evidence: the mean projection difference demonstrates substantial inter-class displacement, the standard deviation characterizes within-class dispersion, Cohen's $d$ quantifies the standardized separation, and Accuracy/AUC evaluate the classification and ranking capabilities of the learned linear probe. Together, the results suggest that the 2560-dimensional hidden representations contain discriminative information associated with confident and hesitant reasoning patterns, which can be captured by a lightweight linear classifier. Although the two categories are not perfectly separable, the observed discriminability provides empirical support for using the learned direction as a hesitation-related signal for adaptive reflection triggering.

### Cross-Validation and Classifier Selection

To rule out the possibility that the observed separability is an artifact of a particular classifier, we adopt a stratified $K$-fold cross-validation protocol with $K = 10$. The 493-sample CoT corpus, comprising 289 confident samples and 204 hesitant samples, is partitioned into ten non-overlapping folds while preserving the class distribution in each fold. PCA is fitted on the training folds only in each round to avoid data leakage. In each round, nine folds are used for training and the remaining fold for evaluation; we report the mean and standard deviation of AUC, Accuracy, F1, Precision, and Recall across all ten rounds.

Table 5: Classifier comparison under 10-fold cross-validation on the 493-sample CoT corpus. Three representative models—LDA, Logistic Regression, and a lightweight Multi-Layer Perceptron (MLP)—are evaluated. All three classifiers achieve highly comparable AUC values within 0.963–0.971, confirming that the observed Confident/Hesitant separation is not an artifact of a specific classifier choice.

| Method | Test Accuracy ↑ | AUC ↑ | Model Complexity ↓ | Inference Cost ↓ | Interpretability |
|---|---|---|---|---|---|
| **LDA (Ours)** | **0.921** | 0.963 | **31 KB (2560-d weight + bias)** | < 1 **ms** | **Single discriminative direction** |
| **Logistic Regression** | **0.941** | 0.968 | ~30 KB (2560-d weight + bias) | < 1 ms | Linear decision boundary |
| **MLP** | 0.929 | **0.971** | ~41 KB | 0.08 ms | Low (nonlinear mapping) |

**Classifier comparison.** To verify that the observed separation is not an artifact of a specific classifier choice, we compare LDA with two representative baselines, including Logistic Regression and a lightweight Multi-Layer Perceptron (MLP), under 10-fold cross-validation on the 493-sample CoT corpus. As shown in Table 5, all classifiers achieve highly comparable performance, with AUC values consistently within the range of 0.963–0.971. The performance gap between the simplest linear model (LDA) and the nonlinear MLP is less than 0.01 AUC, indicating that additional nonlinear capacity provides limited benefit on this near-linearly-separable representation space.

**Reasons for choosing LDA.** Although MLP achieves marginally higher AUC, this improvement does not translate into meaningful practical advantages. In contrast, LDA provides a compact and interpretable decision mechanism by projecting representations onto a single discriminative direction.

Moreover, LDA is well suited to our limited-data setting, where only 493 annotated CoT samples are available. By reducing the hidden-state dimension from 2560 to 128 through PCA, LDA can learn a discriminative direction in a lower-dimensional feature space with relatively modest data requirements. In comparison, MLP involves nonlinear parameter learning and typically requires more training data to achieve reliable generalization, making it more susceptible to overfitting under limited-data conditions. For online inference, the PCA and LDA projections can be combined into one 2560-dimensional direction and a scalar offset, requiring only one dot product on the raw hidden state without explicitly running PCA at every step. Therefore, LDA offers a favorable trade-off among discriminative capability, sample efficiency, interpretability, and computational efficiency, and is selected as the default metacognitive monitor for the Hesitation Score $S_{\text{hes}}^{(t)}$.

**Comparison of Reflection Triggering Mechanisms.** We further compare the LDA trigger with random, periodic, MLP-based, and entropy-based triggering on ALFWorld and WebShop. For ALFWorld, the memory bank is constructed from training tasks and evaluation is performed on the seen test split. We report the average success rate (Avg. SR) on ALFWorld, the task score on WebShop, and average reflections per task (ART) on both benchmarks. The random trigger invokes reflection independently at each step with probability 0.5; a periodic trigger invokes reflection every $T$ steps. The MLP trigger is trained on the same 493 CoT samples used for the LDA probe, while the entropy-based trigger uses the model's predictive uncertainty after completing its CoT reasoning.

Table 6: Comparison of reflection triggering mechanisms on ALFWorld and WebShop. For ALFWorld, the memory bank is built from training tasks and evaluated on seen test tasks. ART denotes the average number of reflection triggers per task. Higher Avg. SR and Score are better; lower ART indicates fewer online reflections.

| Method | ALFWorld | | WebShop | |
|---|---|---|---|---|
| | Avg.SR ↑ | ART ↓ | Score ↑ | ART ↓ |
| Random Trigger | 43.6 | 22 | 66.8 | 26 |
| Periodic Trigger (T = 1) | 54.3 | 39 | 73.2 | 42 |
| Periodic Trigger (T = 3) | 47.1 | 13 | 65.9 | 12 |
| Periodic Trigger (T = 5) | 47.1 | 8 | 64.6 | 10 |
| MLP Trigger | 45.7 | 20 | 64.3 | 26 |
| Entropy-Based Trigger | 52.1 | 2 | 66.0 | 2 |
| **LDA Trigger (Ours)** | **59.3** | 13 | **75.7** | 16 |

**Comparison of Reflection Triggering Mechanisms.** As shown in Table 6, the LDA trigger achieves the highest task performance on both benchmarks, reaching 59.3% Avg. SR on ALFWorld and a score of 75.7 on WebShop, with 13 and 16 average reflections per task (ART), respectively. These results demonstrate the importance of identifying when reflection is necessary, rather than simply increasing its frequency.

**Random and Periodic Triggering.** Random and periodic triggers operate independently of the agent's current reasoning state. The random trigger activates reflection with a fixed probability of 0.5, potentially refreshing an effective strategy at unnecessary steps while missing critical moments when the agent becomes hesitant. Similarly, periodic triggering follows a predefined schedule without considering whether the current strategy remains appropriate for the evolving task state. When reflection is invoked at every step ($T = 1$), performance reaches 54.3% on ALFWorld and 73.2 on WebShop, but requires 39 and 42 reflections per task, respectively. Reducing the triggering frequency to $T = 3$ or $T = 5$ substantially decreases reflection overhead, but also leads to lower task performance, as necessary strategy updates may be delayed or missed. In contrast, LDA achieves higher task performance than the $T = 1$ configuration while reducing reflection calls from 39 to 13 on ALFWorld and from 42 to 16 on WebShop. This comparison shows that more frequent reflection does not necessarily improve task execution; the timing of reflection is equally important for effective adaptation.

**MLP-Based Triggering.** The MLP trigger achieves 45.7% Avg. SR on ALFWorld and a score of 64.3 on WebShop, with 20 and 26 reflections per task, respectively. Despite achieving a comparable cross-validation AUC on the CoT classification benchmark, its downstream performance remains substantially below that of LDA. One possible explanation is the limited size of the training corpus, which contains only 493 labeled CoT samples. Compared with LDA, which learns a single discriminative direction, the nonlinear MLP has greater model capacity and may require more training examples to learn reasoning patterns that generalize reliably to unseen interaction states. Consequently, comparable classification performance on held-out CoT samples does not necessarily translate into equally effective reflection decisions during online task execution. While additional training data may improve the MLP trigger, the present results show that LDA achieves stronger downstream performance under the same limited supervision setting.

**Entropy-Based Triggering.** The entropy-based trigger achieves the lowest reflection frequency, invoking reflection only twice per task on average across both benchmarks. However, its task performance remains below that of LDA, reaching 52.1% on ALFWorld and 66.0 on WebShop. This behavior may be explained by the timing of uncertainty estimation. In our framework, the trigger is evaluated after the agent completes its CoT reasoning, when the model has typically formed a preference for its next action. Consequently, the predictive distribution may exhibit low entropy even when the underlying reasoning is based on an outdated strategy or an incorrect assumption. For example, an agent may confidently decide to revisit a previously explored location despite making no progress. In such cases, low predictive entropy reflects confidence in the selected action rather than the correctness of the reasoning process. Entropy-based triggering may therefore overlook reasoning failures that are not accompanied by high uncertainty in the next-token distribution. By contrast, LDA examines hidden representations associated with confident and hesitant reasoning patterns, allowing it to identify potential reasoning difficulties that are not directly reflected in output entropy.

**Overall Analysis.** Across both benchmarks, LDA achieves the highest task performance while maintaining a moderate reflection frequency. Unlike random and periodic schedules, it conditions reflection on the agent's current reasoning state, avoiding unnecessary strategy updates while responding to potential reasoning difficulties. Compared with the MLP trigger, LDA achieves stronger downstream performance using the same 493 labeled CoT samples, suggesting that its compact linear representation is well suited to the limited supervision setting. Its single discriminative direction also enables lightweight online inference without introducing additional language model calls for trigger evaluation. Together, these results support the effectiveness of LDA-based triggering in balancing task performance and reflection overhead while requiring only a small labeled CoT corpus.

**Clarification of Confident and Hesitant Reasoning Patterns.** In our framework, *confident* and *hesitant* are operational labels derived from reasoning patterns in successful and failed trajectories, rather than direct measures of subjective uncertainty or decision correctness. Specifically, hesitant reasoning refers to patterns associated with potential failures, such as strategy inertia, repeated actions, and fixation on commonsense assumptions. An agent may confidently pursue an incorrect strategy and still exhibit these patterns, whereas uncertainty during reasonable exploration does not necessarily indicate a reasoning failure. Thus, the LDA probe is designed to identify potential failure-related reasoning patterns that may warrant reflection, rather than simply distinguishing between confident and uncertain language.

**Memory-Free Evaluation of Experience Internalization.** To examine whether OPSD internalizes acquired experience into the student's parameters, we compare the base model, SFT, offline distillation, standard OPD, and TMA-OPSD on ALFWorld. During evaluation, all models operate without the memory bank, global strategy, or online reflection. We report task success rate (SR) and the average number of environment steps required by successful episodes. As shown in Table 7, TMA-OPSD achieves 68.6% SR with 9.62 steps, compared with 43.6% and 15.28 steps for the untrained base model. Its higher success rate and shorter successful trajectories under memory-free inference provide direct evidence that experience acquired during training is reflected in the updated student parameters, rather than requiring external memory access at evaluation time.

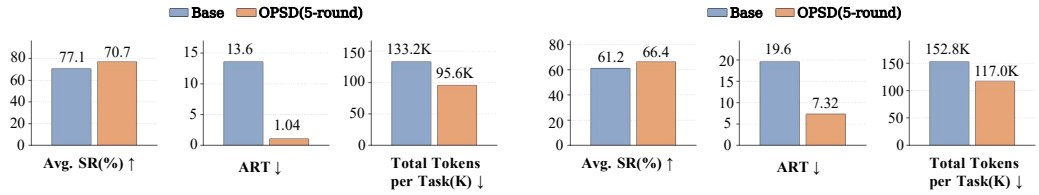
SFT and offline distillation reach 61.4% and 57.9% SR, requiring 12.03 and 12.80 steps, respectively. Their lower performance and longer successful trajectories are consistent with a mismatch between fixed training demonstrations or prefixes and the student's own inference-time behavior, although

Table 7: Memory-free evaluation of experience internalization on ALFWorld. All models are evaluated without the memory bank, global strategy, or online reflection. SR denotes task success rate; Steps denotes the average number of environment steps over successful episodes. Higher SR and lower Steps are better.

| Method | ALFWorld SR ↑ | Steps ↓ |
|---|---|---|
| Base Model | 43.6 | 15.28 |
| SFT | 61.4 | 12.03 |
| Offline Distillation | 57.9 | 12.80 |
| Standard OPD | 64.3 | 11.30 |
| **TMA-OPSD (Ours)** | **68.6** | **9.62** |

these results alone do not isolate that mismatch as the cause. Standard OPD, which distills task execution without the additional reflection objective, achieves 64.3% SR with 11.30 steps. TMA-OPSD improves on both measures, supporting the benefit of learning task execution and reflection capabilities within the same internalization framework. Overall, the memory-free results show that the distilled student completes more tasks in fewer steps even when the online adaptation components are disabled.

**Generalization of OPSD-Internalized Experience.** To further examine whether OPSD enables experience reuse beyond previously encountered tasks, we compare the base model with the student after five OPSD rounds on both seen and unseen ALFWorld tasks. The seen tasks are repeatedly encountered during OPSD training, while the unseen tasks are held out from all distillation rounds and used exclusively for final evaluation. Both models are evaluated under the same TMA inference configuration. We report the average success rate (Avg. SR), average reflections per task (ART), and total inference tokens per task to evaluate task performance and inference efficiency.




(a) Seen Tasks (b) Unseen Tasks

Figure 5: Performance comparison between the base model and the student after five OPSD rounds on seen and unseen ALFWorld tasks. Avg. SR denotes average success rate, ART denotes average reflections per task, and Total Tokens per Task measures average inference token consumption. Higher Avg. SR and lower ART and token consumption indicate better task performance and inference efficiency.

**Experience Internalization on Seen Tasks.** As shown in Fig. 5(a), after five OPSD rounds, the average success rate increases from 70.7% to 77.1%, while ART decreases substantially from 13.6 to 1.04. Meanwhile, total inference tokens per task decrease from 133.2K to 95.6K. These results show that the student becomes more effective at solving previously encountered tasks while relying less on costly online reflection. However, since these tasks are repeatedly encountered during OPSD training, the improvements may partly reflect task-specific experience internalization.

**Generalization to Unseen Tasks.** To distinguish repeated-task adaptation from cross-task generalization, we further evaluate the same distilled student on unseen tasks that are excluded from all OPSD training rounds. As shown in Fig. 5(b), the average success rate increases from 61.2% to 66.4%, while ART decreases from 19.6 to 7.32. Total inference tokens per task also decrease from 152.8K to 117.0K.

Unlike the seen-task evaluation, these improvements cannot be attributed to repeatedly training on the specific evaluation tasks. The distilled student achieves higher task success while requiring fewer reflections and fewer inference tokens on previously unseen tasks, supporting the transferability of the capabilities acquired through OPSD. Although the reduction in reflection frequency is less pronounced than on seen tasks, the consistent improvements across all three metrics suggest that the learned capabilities extend beyond task-specific experience reuse.

Together, these results show that OPSD improves both repeated-task adaptation and generalization to unseen tasks, enabling more effective task execution with lower online deliberation overhead.

# C Agent Prompts

This section lists the prompt templates used by the TMA agent at the three distinct stages of operation. Each prompt is shown verbatim, followed by a short description of its purpose. Placeholders of the form `{name}` are filled at runtime with task-specific content (e.g., the current task description, retrieved trajectories, or the latest hidden state). The templates follow a ReAct-style protocol in which the model first emits a chain-of-thought rationale and then commits to a single admissible action, except where noted.

## C.1 Basic Per-Step Prompt

```
You are an agent solving the following task in an interactive
text environment. Always ground your decisions
in the current state observation and follow the
high-level strategy distilled from past experience.

Task: {task_description}

The following global strategy has been distilled from p
ast experience  and applies to this task. Use it
as a high-level guide, but verify every step
against the current state observation.

Global Strategy:
{z_0}

Active Reflection (only present when the hesitation trigger fired
at this step, otherwise empty):
{f_t}

Current state observation:
{observation}

History of (state, action) pairs so far:
{history}

Think step by step about what to do next,
justify the choice briefly, and then output exactly
one admissible action from the current state.
End your response with a line of the form:

Action: <your_action>
```

## C.2 Initial Global Strategy Generation Prompt

```
You are an expert planner. Given the task below
and a small set of similar past trajectories retrieved
from memory, distill a concise global strategy that
will guide an agent to solve the task from its current state.
```

```
Task: {task_description}

Current initial state observation:
{observation}

Top-{k} retrieved past trajectories (most
relevant first). Each trajectory is a complete
(state, action) sequence together with its
final outcome:
{retrieved_trajectories}

Write a short, natural-language strategy that:
1. Decomposes the high-level goal into
ordered sub-goals.
2. Highlights reusable heuristics, common
pitfalls, and critical preconditions
(e.g., "open containers before placing
items inside", "heat an item before moving
it to a hot-target location").
3. Recommends when the agent should pause,
re-ground in the environment, or re-plan.

Output the strategy as a single short
paragraph written in imperative voice. Do not
output any action.
```

### C.3 Slow-Thinking Reflection Prompt

```
You are performing a deep reflection because
the agent is detected as hesitant or stuck.
Produce a short, actionable local reflection
that explains the immediate bottleneck and
proposes a corrective plan. Do NOT emit any
action---only diagnose and propose.

Task: {task_description}

Current state observation:
{observation}

Recent action history (last few steps):
{recent_history}

Past reflections triggered for this task so
far:
{past_reflections}

Retrieved trajectories that are most similar
to the agent's current state (trajectories
containing both successful and failed
outcomes):
{retrieved_trajectories}

Write a concise reflection that:
1. Diagnoses why the agent is hesitating or
repeating failed actions (e.g., visiting
the same receptacle without opening it,
ignoring an admissible action in the
observation).
2. Identifies the most informative next
sub-goal that would break the current
impasse.
3. Proposes a concrete correction, such as
```

```
re-grounding in a specific object,
revisiting a previously explored location
with a different intent, or explicitly
abandoning an unproductive branch.

Output the reflection as one short paragraph.
Do not output any action.
```

# D Case Study: How LDA Reflection Rescues a Failing Agent

To illustrate how the hesitation-aware reflection trigger works, we examine an ALFWorld episode that would have failed without adaptive reflection. The task is to `put a hot potato in the fridge`. The potato is not initially visible in the agent's starting location, so the agent must systematically search the kitchen, locate the potato, heat it, and then place it in the fridge. Without adaptive reflection, vanilla ReAct-style agents fall into one of two failure modes: (i) a false-intuition shortcut that sends the agent to the stove because the word "hot" triggers a semantic association, or (ii) a non-terminating loop where the agent repeatedly revisits the same countable surface (e.g., countertop 3) without expanding the search to new cabinets. The six annotated steps below show how our LDA-derived hesitation score intercepts both failure modes and recalibrates the agent's behavior in real time.

## D.1 Environment Setup

The episode takes place in a virtual ALFWorld kitchen containing 19 cabinets, 4 countertops, 4 stove burners, a microwave, a fridge, and several drawers. At each step, the agent can choose from 43 possible actions, most of which involve navigation. The potato is initially hidden in a closed cabinet that the agent must open during the task. We examine six key steps (**33**) where the Hesitation Score $S$ becomes positive and triggers reflection. For each step, we present the environment observation, the agent's initial reasoning and intended action without reflection, the reflection generated by TMA, and the action taken after reflection. We also explain how reflection helps TMA correct its decisions and overcome difficulties that vanilla ReAct fails to resolve.

### D.1.1 Step 7 — Early Exploration Deadlock (**LDA**= 12.81, Refresh=True)

**Task**: `put a hot potato in fridge`

**Environment Observation**:

```
You arrive at countertop 3. On the countertop 3,
you see a bowl 2, a cup 1, a soapbottle 2, and a tomato 1.
```

**Agent Thinking (CoT)**:

```
The current situation is that I am at countertop 3,
and the visible items on the countertop are:
a bowl 2, a cup 1, a soapbottle 2, and a tomato 1. There is
no mention of a hot potato in any of these observations.
The task is to "put a hot potato in fridge", which implies that
the hot potato must exist somewhere in the environment
and must be retrieved.

So far, I have:
- Explored countertop 3 (where the hot potato is not visible),
- Attempted to go to cabinet 1 (closed, no potato visible),
- Checked the microwave (closed),
- And returned to countertop 3 multiple
  times.

Given that the task requires a hot potato, and it is not currently
visible or accessible in the current state, I must consider whether
the potato might be in a place I haven't checked yet -
such as a stove burner, where food is often cooked or heated.
```

```
The stoveburners (1-4) are admissible actions.
Since the potato is "hot," it is likely that it was recently cooked
and placed on a stove burner.
```

**Agent Initial Action (before reflection)**: `go to stoveburner 1`

**LDA-Triggered Reflection (top-2 failed trajectories retrieved)**:

```
Pitfall 1: Assuming the potato is on countertop 3 without
verification - failed trajectories show this leads to endless
loops, as the potato is neverfound.

Pitfall 2: Skipping location checks and immediately heating
a non-existent or cold potato.

Concrete next steps:
1. Go to countertop 3 to confirm the potato's presence directly,
as it's the only location where it's ever been observed.
2. If not found, systematically check other cabinets or drawers
where it might be stored.
3. Only proceed to heat the potato after confirming its presence
and that it is in hand.

Current action: go to countertop 3.
```

**Agent Final Action (after reflection)**: `examine countertop 3`

**Architectural advantage.** The agent's initial impulse is to go to `stoveburner 1`, a semantic shortcut triggered by the word *hot*. The reflection directly warns "verify existence first", so the agent instead examines the current surface. This single correction prevents the agent from prematurely entering the heat phase before the potato has even been located — a failure mode that would otherwise consume dozens of steps.

#### D.1.2 Step 13 — Continued Exploration, Loop-Breaking (**LDA**=21.54, Refresh=True)

**Task**: `put a hot potato in fridge`

**Environment Observation**:

```
On the countertop 3, you see a bowl 2, a cup 1,
a soapbottle 2, and a tomato 1.
```

**Agent Thinking (CoT)**:

```
The current observation shows that on countertop 3,
there is a bowl 2, a cup 1, a soapbottle 2, and a tomato 1.
None of these items are a hot potato.
The task is to "put a hot potato in fridge",
which implies that a hot potato must be located
somewhere in the environment and then placed in the fridge.

So far, in the previous steps, we have:
- Visited countertop 3 (where the hot potato is not visible)
- Checked cabinets 1, 2, 3 - all closed or empty
- Checked microwave, fridge, sink, stove, etc.,
but no mention of a potato.

Since the hot potato is not visible on the countertop
or in any of the cabinets we've checked,
and since the task requires placing a hot potato
in the fridge, it is likely that the potato was either:
- Already in a location we haven't explored
(e.g., a drawer, stove, or microwave)
- Or, we may have missed it due to lack of inspection
```

**Agent Initial Action (before reflection)**: `go to microwave 1`

**LDA-Triggered Reflection (top-2 failed trajectories retrieved)**:

```
Pitfall: Repeatedly assuming the potato is on countertop 3
without verification, leading to loops and wasted steps
(as seen in failed trajectories). Failed agents heat
cold potatoes or ignore the task entirely.

Next steps:
1. Examine countertop 3 directly to confirm potato presence
(current observation shows no potato).
2. Check other cabinets (e.g., cabinet 4) systematically,
as potatoes may be stored there.
3. If no potato found, verify fridge contents -
a hot potato may already be inside, requiring only placement.
```

**Agent Final Action (after reflection)**: `go to cabinet 4`

**Architectural advantage.** The reflection explicitly tags "repeatedly checking countertop 3" as a failure pattern and pushes the search to `cabinet 4`. This is the pivotal moment at which the agent transitions from a local loop to systematic enumeration — a transition that is hard for non-LDA baselines to make because their policies either commit too early (Semantic shortcut) or recurse too late (Random walk).

#### D.1.3 Step 19 — Injecting Decision Philosophy (**LDA**=23.07, Refresh=True)

**Task**: `put a hot potato in fridge`

**Environment Observation**:

```
You arrive at cabinet 7. The cabinet 7 is closed.
```

**Agent Thinking (CoT)**:

```
We are at cabinet 7, which is currently closed.
Our goal is to put a hot potato in the fridge. So far,
we have not seen a hot potato in any of the observations.
The most recent items we've seen on the countertop
are a bowl 2, a cup 1, a soapbottle 2, and a tomato 1.
We have opened several cabinets, but only found a bowl 1
and a pan 1 in cabinet 6.

Since the task is to "put a hot potato in the fridge",
we must first locate the hot potato. It is not
on countertop 3 (as observed), and we have not seen it
in any cabinet or drawer.

Given that we are currently at cabinet 7, and the only
admissible action that allows us to inspect its contents
is to "open cabinet 7", this is a logical next step.
Opening it may reveal the hot potato or at least provide
information about whether it is present.
```

**Agent Initial Action (before reflection)**: `open cabinet 7`

**LDA-Triggered Reflection (top-2 failed trajectories retrieved)**:

```
Pitfall 1: Blindly assuming the potato is on
countertop 1 or 3 without verification
- failed trajectories repeatedly start here,
leading to loops.

Pitfall 2: Skipping systematic cabinet searches
and heating attempts prematurely.
```

```
Current fix: Stop looping; verify existence first.

Next steps:
1. Go to cabinet 7 (next in sequence after 6) to
check for potato.
2. If no potato found, examine fridge 1
(hot potato may already be inside).
3. Only heat if potato is found and cold.
```

**Agent Final Action (after reflection)**: `open cabinet 7`

**Architectural advantage.** Here the reflection does not rewrite the action (`initial = final = open cabinet 7`), but it actively injects the guiding principle of "systematic enumeration". This highlights TMA's robustness: reflection does not only change actions, it can also reshape the decision philosophy. Vanilla ReAct is constrained to its current policy and cannot refactor its search strategy at run-time.

#### D.1.4 Step 26 — Reinforcing the Search Path (**LDA**=22.56, Refresh=True)

**Task**: `put a hot potato in fridge`

**Environment Observation**:

```
You arrive at cabinet 8. The cabinet 8 is closed.
```

**Agent Thinking (CoT)**:

```
We are at step27, and the current observation is:
"You arrive at cabinet 8. The cabinet 8 is closed."

Admissible actions include: examine cabinet 8,
go to various cabinets, countertops, appliances, etc.,
open cabinet 8.

Our task is: put a hot potato in the fridge.

So far, we have not seen a hot potato anywhere
in the environment. The only items we have observed on
countertops or in cabinets are:
- bowl 2, cup 1, soapbottle 2, tomato 1 (countertop 3)
- bowl 1, pan 1 (in cabinet 6)

Therefore, the hot potato must either:
1. Be in a closed cabinet or drawer that
we haven't explored yet.
2. Be on a different surface we haven't examined.
```

**Agent Initial Action (before reflection)**: `open cabinet 8`

**LDA-Triggered Reflection (top-2 failed trajectories retrieved)**:

```
Pitfall 1: Repeatedly examining the fridge without
finding the potato, leading to infinite loops.
Failed trajectories show agents heat a cold potato
or waste steps without verifying existence.

Pitfall 2: Assuming the potato is on countertop 3
or in early cabinets without systematic expansion.

Concrete next steps:
1. Go to cabinet 8 (next in sequence after 7)
to expand search systematically.
2. If no potato found, go to countertop 1 or 2
to check alternative locations.
3. Only heat the potato after confirming its
```

```
presence and location.
```

**Agent Final Action (after reflection)**: `open cabinet 8`

**Architectural advantage.** The reflection reinforces the specific strategy "cabinet 8 takes priority" — in the very next step (**step 27**) the agent moves to `countertop 1`, where it ultimately finds `potato 1`. This is direct evidence that the reflection text is not merely explanatory; it actively reshapes the agent's subsequent exploration path, three steps ahead of the trigger.

#### D.1.5 Step 33 — Avoiding the Heat Loop (**LDA**=23.48, Refresh=True)

**Task**: `put a hot potato in fridge`

**Environment Observation**:

```
You arrive at microwave 1. The microwave 1 is open.
In it, you see a apple 1, and a egg 1.
```

**Agent Thinking (CoT)**:

```
I am currently at microwave 1, which is open.
Inside the microwave, I see an apple 1 and an egg 1.
I have picked up the potato 1 earlier from countertop 1,
and I now need to put the hot potato in the fridge.
However, the potato is not currently in the microwave
or any other location that I can see.

The task is to "put a hot potato in fridge."
This implies that the potato must be heated (to be "hot")
before being placed in the fridge.
Since I have the potato in my inventory
(as I picked it up from countertop 1),
and I am currently at the microwave,
I can use the microwave to heat it.

The admissible actions list includes
"heat potato 1 with microwave 1" - this is exactly the step
needed to make the potato hot.

Therefore, the correct next action is to heat the potato
in the microwave.
```

**Agent Initial Action (before reflection)**: `heat potato 1 with microwave 1`

**LDA-Triggered Reflection (top-2 failed trajectories retrieved)**:

```
Pitfall 1: Repeatedly heating the potato without confirming
it's in the microwave or that it's hot - failed trajectories
show endless loops of "heat → nothing happens → repeat."

Pitfall 2: Assuming the potato is on a surface when
it's actually in inventory; failed agents ignored inventory
and moved it incorrectly.

Next steps:
1. Confirm potato is in inventory via inventory before heating.
2. If in inventory, go to microwave and use heat potato 1
with microwave 1 only after opening it.
3. Avoid re-examining fridge or countertops
without a clear purpose.

Current action: inventory to verify possession.
```

**Agent Final Action (after reflection)**: `inventory`

**Architectural advantage.** This is the most pivotal turning point of the trajectory. The agent has already obtained the potato and is one step away from completion, but its initial impulse is to commit to `heat` without re-asserting that the potato is in its inventory. The reflection leaves the immediate action decision intact (`initial = heat`, `final = inventory`) but adds a critical verification step that prevents a heat loop where the agent repeatedly issues `heat` when the potato is no longer in hand. This is the most direct evidence that TMA's reflection is not merely a re-ranker of plausible actions — it can substitute a semantically safer action even when the agent's instinctive output is technically valid.

### D.2 Summary

Across the five traced steps, the LDA-derived hesitation score intercepts every failure mode that vanilla ReAct exhibits on this task: false intuition (**Step 7**), repetition traps (**Steps 13, 19**), search-path injection (**Steps 19, 26**) and post-acquisition verification (**Step 33**). In every case, the reflection text generated by Stage 2 is grounded in trajectories stored in the factual memory $\mathcal{M}$, and the resulting $f_t$ is overwritten after each invocation. Without the LDA trigger, the agent would require either human-engineered rules for each failure mode or a much larger LLM to recover from these stuck states empirically.

## E Limitations of LLM-Prompted Strategy Adaptation

We present a failure case in which an LLM agent becomes anchored to historical experiences and repeatedly revisits previously explored locations. The task is `heat some egg and put it in garbagecan`. Although the egg is absent from both the countertop and the fridge, the agent continues searching these locations until reaching the maximum limit of 50 steps.

```
## Historical Experience

The egg was found on countertop 1 in a previously successful trajectory.

## Step 0: Initial Exploration

Thought:
The historical experience suggests that the egg is on countertop 1.

Action: go to countertop 1

## Step 3: Repeated Search

Observation:
No egg is found on countertop 1 or in fridge 1.

Thought:
The historical experience suggests that the egg should be on countertop 1.
I should check there again.

Action: go to countertop 1

## Steps 20-40: Action-Repetition Loop

Observation:
No new environmental information is obtained.

Thought:
Since eggs are commonly kept on fidge,
maybe I haven't explored the environment thoroughly enough.
I should check the fridge again, as they might have been moved there.

I decide not to refresh the strategy.

Action: go to fridge 1
```

```
(Repeated 16 consecutive times)

## Step 50: Task Failure

Observation: Maximum steps reached.

Success: False
Progress: 0.0
```

**Failure Analysis.** Despite repeated negative observations, the agent remains anchored to historical experience and commonsense assumptions, revisiting the countertop and fridge instead of reconsidering its search strategy. This eventually leads to an action-repetition loop and task failure.